2021

Decrypting legal dilemmas

BOSTON UNIVERSITY

GRADUATE SCHOOL OF ARTS AND SCIENCES

Dissertation

DECRYPTING LEGAL DILEMMAS

by

SARAH ANN SCHEFFLER

B.S., Harvey Mudd College, 2015

Submitted in partial fulfillment of the

requirements for the degree of

Doctor of Philosophy

2021

Approved by

First Reader

Ran Canetti, Ph.D.
Professor of Computer Science

Second Reader

Mayank Varia, Ph.D.
Research Associate Professor of Computer Science

Third Reader

Joan Feigenbaum, Ph.D.
Grace Murray Hopper Professor of Computer Science
Yale University

Fourth Reader

Adam D. Smith, Ph.D.
Professor of Computer Science

How can you expect to handle the future if you can't even handle the present?
Daniel Suarez, *Daemon* (2006)

Acknowledgments

First, I thank my committee, Profs. Ran Canetti, Mayank Varia, Joan Feigenbaum, and Adam Smith, for the intellectual work they have put in to help me improve this dissertation.

Second, I thank my advisor, Prof. Mayank Varia. Mayank is not only a superb advisor and mentor, but I also consider him a friend. I am incredibly grateful for the time he's spent helping me to improve over the years, and also the fun discussions we have had about privacy, governance, technology, and many other topics.

Third, I thank my parents, Paul and Stephanie Scheffler. From both of them, I learned to care. Among other things, ever since I started studying cryptography, my dad Paul has sent me enciphered letters for my birthday every year. I owe him my scientific curiosity, my knowledge in subjects from particle physics to environmental regulations, and my speed at monoalphabetic substitution ciphers. My mom Stephanie has supported me when I was up, supported me when I was down, and talked me through so many difficult transition periods throughout my graduate school career. She's inspired me to take care of myself both physically and mentally. I hope I've made you both proud. I wish Mutti and Grandpa were here to see this day as well, I know they would burst with happiness.

Fourth, I thank my fiancé Andrew Fishberg. It's rare indeed to find someone who cares about doing the right thing so deeply, and still remembers to have fun and laugh at yourself. I know I've learned a lot and honed my moral compass through discussions with you, and I know you've done the same. I know I can – and do – trust you with anything up to and including advice on this dissertation. It's still a miracle we found each other, given the number of things that needed to go right in each of our lives for us to meet each other under the right circumstances. I love you dearly and I can't wait to see where our path will take us.

Fifth, I thank my best non-fiancé friend Devon Stork. You are the source of lifetimes of exploration of both fantastical and real worlds. I am so inspired by the attention you give to your own research and your own life choices. I am happy I can rely on you to drive me to do better, and that you will take my own suggestions seriously as well. I hope to be lifelong friends, and I am excited to see where we will go.

Sixth, I thank my friend and coauthor Aloni Cohen. I have eagerly watched Aloni's career path and used his own experiences to guide my own. We have thought through difficult problems together, using each other as foils to test our hypotheses. Along the way we became friends as well, and I am grateful for his intellectual contributions and his support. I wish you the best of luck in your future professorial endeavors!

Finally, I must additionally thank several categories of people who helped me:

I thank both of my immediate families: Paul and Stephanie Scheffler; Andrew, Carol, Dave, and Erica Fishberg; and Abe Perez.

I have had the opportunity to work with several past and future coauthors from whom I have learned much, especially Yaron Gvili, Eran Tromer, and Kobbi Nissim. I thank them for their experience and both intentional and unintentional guidance.

I thank Jonathan Mayer, my upcoming faculty host at Princeton's Center for Information Technology Policy. I look forward to our future research.

I thank several people from other institutions whom I look up to and from whom I have received some helpful mentorship and advice: Ben Fuller, Micah Altman, and Joan Feigenbaum, who also served on my committee. I thank Ron Rivest for fostering some of my initial interest and expertise in cryptography from afar. I also thank Whit Diffie – we had only one conversation, but it was an impactful one.

I thank the professors in the Computer Science Department and School of Engineering at BU from whom I have learned much and received much advice. I especially thank Sharon Goldberg for her advisorship at the beginning of my PhD journey. Additionally I thank Ari Trachtenberg, Leo Reyzin, Mark Bun, Abraham Matta, Mark Bun, Steve Homer, and Alley Stoughton. And of course Ran Canetti and Adam Smith, who also served on my committee, and Mayank Varia, who served as my advisor and on my committee.

I also thank several professors in the Law School at BU for their classes and academic advice. I also owe them my interest in law: Stacey Dogan, Andy Sellars, Julissa Milligan, and Ahmed Ghappour. I also thank the entire CyberAlliance and its organizers for being interested in the same sorts of questions as I am, and doing the work to organize some cross-pollination and collaboration.

I thank my friend-colleagues: Rawane Issa, Nicolas Haddad, Kinan Dak Albab, Ariel Hamlin, Leo Alcock, Jacob Ostling, and Hoda Maleki.

For good times and being helpful role models, I thank the PhD students and postdocs ahead of me at Boston University (some of whom have since attained professorship) that I interacted with especially much: Ethan Heilman, Sophia Yakoubov, Aanchal Malhotra, Oxana Poburinnaya, Ran Cohen, Xiao Wang, and Gabe Kaptchuk.

I also thank the PhD students that will follow me at BU that I have enjoyed talking to over the last few years: Pratik Sarkar, Luowen Qian, Gavin Brown, Marika Swanberg, Palak Jain, Islam Faisal, and Meghna Sengupta.

I also thank my mostly non-academic friends Karishma Chadha, Nicole Francisco, Anjali Kayal, Patrick Cable, and Andy Russell for their great support and friendship over the years. I also thank my slightly more academic friends Govind Ramayanan, Adam Suhl, Ben Kaiser, and Max Comstock for assorted acts of social and professional support.

I thank the undergraduate students who I had the pleasure to work with in the last year of my PhD, Julie Ha and Ziling Yang (and Xinyuan Zhang, who I had less time with since she graduated earlier). Working with them and mentoring them has been an extremely rewarding experience and I wish them the best success in their future careers.

I thank the various outreach teams I was a part of: the Titanoboa team, the Code Creative team, and the RACECAR team I have gotten to assist through Andrew.

I thank Erik Demaine and the Algorithmic Lower Bounds group for some of the most unapologetically fun research I have had the opportunity to take part in.

I thank several undergraduate professors for encouraging my passion for cryptography and also maintaining, as the saying goes, "an understanding of the impact of my work on society": Colleen Lewis, Ran Libeskind-Hadas, and Ghassan Sarkis.

I thank Erica Seymourian, Katherine D'Angelo, Kori MacDonald, and Chris Devits for putting up with my constant needs for scheduling, helping me figure out how to host talks, reserve conference rooms, order food, and meet my official academic requirements, and other assorted pieces of administrivia that turned out to be much more challenging than I expected.

I thank Bill Jemas and Jane Milrod for their support even before I became a part of their friend-family network and of course continued afterward, their advice to me and Andrew, and their help in Princeton-related questions.

I thank the Clare Booth Luce Foundation for supporting my PhD research in the initial portion of my graduate school career. I thank the Google PhD Fellowship and especially Nina Taft for going out on a limb to support the latter part of my graduate research.

I thank Devon Stork (again) and his partner Amanda Lemire for being great friends for many years. I also thank Korleki Atiki, Andrew Kane, and Mary Morrison for the great times we've had playing tabletop RPGs in person, and Amy Greaney, Danielle Marquis, and James Hobin for an excellent virtual game.

Many people have supported my journey in the past, and many others will continue supporting them in the future. I appreciate you all so much, and I hope I have made you all proud.

Sarah Scheffler
Ph.D. Student
CS Department

DECRYPTING LEGAL DILEMMAS

SARAH ANN SCHEFFLER

Boston University, Graduate School of Arts and Sciences, 2021

Major Professor: Mayank Varia, PhD
Research Associate Professor of Computer Science

ABSTRACT

It has become a truism that the speed of technological progress leaves law and policy scrambling to keep up. But in addition to creating new challenges, technological advances also enable new improvements to issues at the intersection of law and technology. In this thesis, I develop new cryptographic tools for informing and improving our law and policy, including specific technical innovations and analysis of the limits of possible interventions. First, I present a cryptographic analysis of a legal question concerning the limits of the Fifth Amendment: can courts legally compel people to decrypt their devices? Our cryptographic analysis is useful not only for answering this specific question about encrypted devices, but also for analyzing questions about the wider legal doctrine. The second part of this thesis turns to algorithmic fairness. With the rise of automated decision-making, greater attention has been paid to statistical notions of fairness and equity. In this part of the work, I demonstrate technical limits of those notions and examine a relaxation of those notions; these analyses should inform legal or policy interventions. Finally, the third section of this thesis describes several methods for improving zero-knowledge proofs of knowledge, which allow a prover to convince a verifier of some property without revealing anything beyond the fact of the prover's knowledge. The methods in this work yield a concrete proof size reduction of two plausibly post-quantum styles of

proof with transparent setup that can be made non-interactive via the Fiat-Shamir transform: “MPC-in-the-head,” which is a linear-size proof that is fast, low-memory, and has few assumptions, and “Ligero,” a sublinear-size proof achieving a balance between proof size and prover runtime. We will describe areas where zero-knowledge proofs in general can provide new, currently-untapped functionalities for resolving legal disputes, proving adherence to a policy, executing contracts, and enabling the sale of information without giving it away.

Contents

List of Tables

List of Figures

List of Abbreviations

5A	Fifth Amendment
AP	Accuracy Profile
FC	Foregone Conclusion
FNR	False Negative Rate
FPR	False Positive Rate
IKOS	Ishai, Kushilevitz, Ostrovsky, and Sahai
ITM	Interactive Turing Machine
MPC	Multi Party Computation
NIST	National Institute of Standards and Technology
NPV	Negative Predictive Value
PPV	Positive Predictive Value
RAM	Random Access Memory
U.S.	United States
ZK	Zero Knowledge
ZKAoK	Zero Knowledge Argument of Knowledge
ZKPoK	Zero Knowledge Proof of Knowledge

Chapter 1

Introduction

Like all previous years, 2021 is the year of the greatest technological change in American history, and I have every reason to expect that the pace will continue accelerating in the future. Each new technology brings with it a host of new social questions. Broadly, we can ask how the technology can improve the lives of individuals in a society, as well as the society itself? What aspects of the technology have problematic aspects that must be discouraged or prohibited, by individuals, companies and organizations, or governments? These broad questions can be broken down into more specific targeted puzzles: How can law enforcement use the technology constitutionally, or corporations use it statutorily? What statistical properties do we want from the new technology, in terms of usage, performance, or fairness? Can we make any targeted changes to existing technologies to make them more efficient, easier to use, or better quality? The converse of the questions is often meaningful as well: What legal or societal conditions could this technology improve or worsen? Can we design tools that meet a specific societal niche?

These are questions one could ask of any technology, but they become essential challenges in technologies that are very powerful, widely adopted, or both. In the last decade, *cryptography* has become ubiquitous on the Internet. Cryptography adds both enforceability and anonymity to electronic transactions. Consumer devices now offer options for encryption, in many cases by default. Encrypted communication is not only the norm for government employees, but also increasingly for average people in the U.S. More advanced forms of cryptography have been adopted by the U.S. military to secure communications and operations in the field, and by government agencies to store and compute on sensitive information.

This growth in the adoption, scale, and power of cryptography has forced more and more legal and societal questions about how to deal with it. Cryptography is involved not only in laws involving privacy, but also those that govern identity, accountability, or the enforcement of requirements. Even in laws that seem technology-agnostic on their face, cryptography often introduces new wrinkles and previously untested edge cases. Specific laws and policies involving fundamental internet infrastructure, surveillance, census, and commerce, are all increasingly intertwined with cryptography. New norms brought about by cryptography and anonymity are leading to increased scrutiny on what general people, governments, and corporations, should be allowed – or required – to do with cryptography.

The problems posed by these legal dilemmas compound over time if they are not dealt with. We are only just now beginning to inspect the legal dilemmas posed by the new technologies of cryptocurrencies and blockchain. At the same time we are also finally addressing the legal nuances of full disk encryption, a technology that has been mature for several decades, but has only seen mass adoption within the last several years. By the time a technology becomes ubiquitous, regulating it becomes a challenge because the business interests of the technology's owner become a factor when weighing the societal benefits and costs – and the legality – of limiting or requiring its use. Changes become much harder to make.

Deciding when to address these problems is challenging. Far-reaching questions are often challenged by the legal community because they are only hypotheticals. And it is difficult to predict in advance which predicted consequences of a new system will be the most important. But if we always wait for those hypotheticals to become reality, we will have ensured that we will always be playing a perpetual game of catch-up.

A good example is how algorithms have brought to the forefront a new examination of statistical “fairness” in our decision-making processes. Algorithmic fairness seeks to analyze decision-making methods under various formal definitions of fairness. There is nothing special about analyzing an algorithm with these methods – human decision-makers are subject to the same constraints. Indeed, in most algorithmic fairness research the algorithm is treated as a black-box, and there is literally no difference between measuring the fairness of an algorithm, and of a human judge. We could have formulated these concepts decades, or even centuries earlier. In fact, as we will discuss further in §3.1.1, we did formulate some of the concepts decades earlier in the 1960s and 1970s, in the context of standardized testing, however, these methods did not stand the test of time [87]. These mathematical ideas of “fairness” did not rise to public prominence because technology had not yet forced the issue. Unlike human decision-makers who can each make only so many decisions per day, the rise of computing allowed a single algorithm to determine the fates of millions of people. This called attention to the underlying problem that had always been there.

Extending this example, we believe the general principle is clear: research that examines the intersection of technology and law can unearth important questions that would otherwise go unquestioned, and in the best cases, can even begin to answer those challenging questions. There is nothing particularly special about this moment in time with regard to this principle. But the object-level topics change from year to year. This work seeks to conduct interdisciplinary research in cryptography, algorithmic fairness, law, and policy that is relevant in 2021. Without succumbing to techno-solutionism, we believe that our interdisciplinary work provides useful tools for policymakers in three different areas.

1.1 Protecting information and computation in 2021

The specific technology questions we face in the U.S. today are largely to do with the consequences of companies holding many large datasets about individuals, and the appropriate limitations on the government's power to collect, store, and analyze information. These questions are intertwined with both old and new advancements in cryptography and machine learning that are now coming into general use. These advancements are somewhat new, but not cutting-edge – it is only their wide adoption that is now forcing the legal issues.

Questions of data collection were amplified first in the mid 1990s-2000s with the advent of the consumer Internet, and amplified further by the proliferation of cell phones and then smartphones. Computing was just starting to become cheap enough that large companies and government entities had the resources to analyze the new "Big Data" that was being generated automatically from so many sensors and monitors. Now, two decades later, even small companies have access to a wealth of data about their users, and some challenges to government use of this information have arisen, in part in response to first the PATRIOT Act and later the Snowden revelations.

Now, we are pushing hard to understand the tools available to protect information and computations on this data. This protection can take the form of a legal restriction, such as Constitutional or statutory restrictions on data use or the collection of information. We will discuss a Fifth Amendment Constitutional limitation to government information collection in §2. The protection might also take the form of a fairness requirement, an obligation to ensure that data analysis treats different demographic groups equally, especially by requiring the equalization of the analysis' accuracy across demographic groups. In §3 we will examine the kind of protection that is available for this analysis, and propose new methods for such protection. Finally, the protection might be a simple privacy requirement – though the implementation of these requirements is never simple. A key invention that allows computing with sensitive data without compromising privacy is the zero-knowledge proof. Zero-knowledge proofs have risen from theoretical proposal to practicality within only a few decades, and in §4 we will present two concrete improvements for zero-knowledge proofs to further enhance this form of data protection.

In the remainder of this introduction, we will describe our results in each of these three areas, which will be presented in greater detail in later chapters.

1.2 Our results

This thesis contains three main results that advance socio-technical interdisciplinary research. In the context of the "crypto wars" and compelled decryption, we conduct a cryptographic analysis of the *foregone conclusion doctrine* as an exception to the Fifth Amendment privilege against self-incrimination. We then investigate a problem that arises when seeking fair decision-making: how can the result of a calibrated nonbinary classifier be post-processed into a binary decision that meets a statistical fairness requirement? And last, in more traditional cryptography work, we improve the tradeoff between privacy and functionality in zero-knowledge arguments.

1.2.1 Cryptography and the Fifth Amendment privilege against self-incrimination

Two fundamental human rights in free and democratic societies are the right to remain silent and the right to avoid self-incrimination. More than 100 countries around the world have enshrined some version of these rights [284], which collectively protect people from being forced by their own governments to provide the evidence needed to convict themselves of a crime. Jewish law included a right against self-incrimination at least as far back as the fifth century [219] and the privilege appears in Islamic law as well [231].

In the United States, these rights stem from the Fifth Amendment to the U.S. Constitution, which states in part that "[n]o person ... shall be compelled [by the government] in any criminal case to be a witness against himself" [300]. The rise of ubiquitous, strong cryptography has forced courts to consider how all aspects of the law apply to cryptography, including the right to silence. To date, the most prominent question surrounding cryptography and the right to silence is the following: if the government seeks as evidence a computer file that is encrypted using a key derived from a password, *can the government compel the device's owner to use her password in order to decrypt the file*?

Taken at face value, it seems that the answer to this question should be "no": the device's owner can invoke her rights in order to refuse the government's request. However, the answer to this question is more subtle because the rights to silence and to avoid self-incrimination are not absolute: they only protect actions that depend non-trivially on the contents of one's mind. For instance, the U.S. Supreme Court has held that the government can compel people to state their own name [165], provide a handwriting exemplar [139], or provide a blood sample [263] despite the right to avoid self-incrimination.

The question then arises: how significantly does decryption depend on the contents of one's mind? Both the court system (e.g., [175, 302] among U.S. circuit courts and several cases at the state and U.S. district levels) and scholars with expertise in law and technology [85, 193, 198, 222, 256, 282, 317] have divided on this question, and they all provide different non-technical arguments about how to extend existing norms and principles surrounding the right to silence so that they apply to cryptography. In §2, we provide a technical framework for the relevant legal doctrine, which we then use to reason that the answer to the compelled decryption question should often be "no."

We believe compelled decryption is an especially fruitful target for interdisciplinary cryptographic and law research for two reasons: First, existing legal research on compelled decryption almost exclusively relies on analogies to various physical objects like safes or shredders. Although analogies are helpful in many other areas of the law, in the compelled decryption context the different choices of analogy lead to different legal outcomes. Being able to reason about a cryptosystem directly allows us to avoid negotiating this edge case between analogies. Second, as we will discuss further, the underlying principles of the particular relevant legal doctrine happen to align quite nicely with the cryptographic conception of knowledge.

We stress that this question about compelled assistance is different than the more prominent part of the crypto wars, in which governments wish to mandate use of cryptosystems that they can decrypt on their own. The main thrust of the crypto wars has to do with different areas of the law. It especially involves the Fourth Amendment, such as whether encryption provides a reasonable expectation of privacy [74,206,297], and the First Amendment, e.g. whether free speech extends to the right to develop encryption software or conduct encryption research [40,119].

When dealing with the Fifth Amendment, things are different: the Fifth Amendment describes what the government may order *you* to do, in contrast with what it may *find out about you* without your involvement. In §2, we will show that it is possible to design encryption schemes that are "strong" in the sense that they preclude governments from decrypting files on their own, but are nevertheless vulnerable to the government compelling you to decrypt files for them.

Toward a rigorous definition of the *foregone conclusion doctrine*

Rather than simply viewing cryptography as a technology that introduces new legal questions, in this work we leverage the ideas of cryptography to codify legal principles and then formally prove whether they apply to any given cryptosystem. Concretely, this work examines a small yet crucial part of the right to silence called the *foregone conclusion doctrine* that is the source of all government cases involving compelled decryption in the United States (we will describe it in detail in §2.2).

Interestingly enough, this doctrine has a style that is similar to the way that cryptographic definitions are typically written: informally, it states that the government may compel a specific “implicit” type of self-incriminating testimony if (and only if) it already knows the testimony involved. This may seem tautological, but its purpose becomes clearer in the context of compelling production of documents. The documents themselves may not be protected under the Fifth Amendment privilege against self-incrimination, but the act of producing those documents might be; the foregone conclusion doctrine originated with this setting in mind. We formalize the doctrine under a cryptographic lens, providing a rigorous simulation-based cryptographic definition and formally proving whether various cryptographic protocols are susceptible to it.

At a high level, the goal of our definition is intuitive: the government can only compel a query if it can be answered without relying heavily on the contents of your mind, and that is the case if the government can simulate the response to the query based upon its prior evidence about the case and access to everything in the world *except* the contents of your mind. We also prove that the definition satisfies sequential composition, which means that compelling one action cannot change the status about whether any other action is compellable.

To justify our definition, we demonstrate that it correctly adjudicates all non-encryption-related foregone conclusion cases argued in the U.S. Supreme Court since the modern interpretation of the Fifth Amendment arose in 1976 [123] and the five most important cases at the circuit court level (i.e., the next level of the court hierarchy) as identified by legal scholars [96, 193, 198, 317]. We purposely ignore cases involving encryption since the Supreme Court has never ruled on them to date and lower courts have split on them, leaving no reliable benchmark to use.

Determining if crypto can be compelled

We reason about the government's ability to compel disclosure of cryptographic secrets. To answer the question raised above: we prove that under our definition, decryption under a password-derived key is typically *not* compellable. However, if the encryption scheme is extended with certain features (including those that are often used to bolster security overall) then it may become compellable. Additionally, we show that compelled disclosure composes with other parts of the crypto wars in a debilitating way: if there exists a reliable method for the government to decrypt data without you (even one that is somewhat costly, e.g. [319], though not one that would require superpolynomial expected time), then the government can compel you to perform the decryption instead. This makes various proposals for "exceptional" law enforcement access to devices worrisome for Fifth Amendment reasons in addition to Fourth Amendment concerns.

We also consider the government's ability to compel a person to reveal preimages to one-way functions, open messages protected within cryptographic commitments, and prove statements in zero-knowledge. We find that the government is unable to compel preimages, cannot compel the opening of commitments, and may only compel a respondent to prove a statement already known to be true to the government, and even then, it must be proven in zero knowledge. On the other hand, the government generally *can* compel a respondent to encrypt or commit to a secret under fresh randomness. While we are unaware of any court challenges to date that compel use of these cryptographic primitives, they may come someday, and our definition enables us to be forward-looking to determine whether cryptosystems can withstand these threats.

Bolstering cryptosystems against compelled disclosure

We consider how voluntary use of cryptographic systems exposes parties to higher risk of compelled actions in the future. We find that secure multi-party computation (MPC) is vulnerable to this threat: engaging in MPC protocols may increase the compellability of a party's sensitive input data. Then, we design and implement countermeasures that provably render secure computation protocols resilient to compelled requests. Our countermeasures apply to 2-party computation via Yao's garbled circuits [325], with extensions to malicious security via cut-and-choose [213, 238] or authenticated garbling [313]. We implement the latter and show that it adds a small additive factor to the runtime that is independent of the circuit size. We also show how to extend the construction to a multi-party protocol where several parties receive output while maintaining resilience against compelled requests, by incorporating techniques from differential privacy.

1.2.2 Fairly post-processing calibrated non-binary classifiers to binary decisions

The second main portion of this work concerns algorithmic fairness. Algorithmic fairness is roughly divided into works that write statistical definitions of "fairness" – a challenging task to begin with – and works that apply those definitions to decision-making systems in society. Its broad goal is to ensure fairness, equity, and accountability in algorithmic decision-making, especially high-impact decision making such as loan offerings, education and testing, employment, incarceration, and health care.

Many ideas in algorithmic fairness predate machine learning and had been previously discussed in the context of standardized testing, as we will discuss further in §3.1.1. The increasing use of machine learning and algorithmic decision-making in contexts where data points correspond to individual people caused a renewed interest in the analysis of these systems for bias, especially along legally-relevant demographic characteristics such as race and gender.

Most recent algorithmic fairness research falls under the *group fairness* paradigm. In this paradigm, inputs to the classifier belong to one or more *protected groups* (often, individuals are assumed to belong to exactly one protected group, as will be true in our work as well). For the most part, the goal in the group fairness paradigm is to equalize some error metric between the output of a classifier when run on individuals from the different groups. For a binary classifier, this could mean equalizing the False Positive Rate (FPR), False Negative Rate (FNR), Positive Predictive Value (PPV), or Negative Predictive Value (NPV), or a combination thereof. For nonbinary classifiers, this generally involves groupwise calibration (an extension of PPV and NPV to the non-binary setting), or "balance" of errors (an extension of FPR and FNR). Two key works [81, 200] found slightly different variants of the same finding: equalizing all of these metrics simultaneously is impossible unless one of two conditions is met: either the groups have equal underlying base rates (an equal fraction of members in each group are in the positive class as opposed to the negative class) or the classifier is perfectly correct (and therefore $\mathsf{PPV} = \mathsf{NPV} = 1$ for both groups, and $\mathsf{FPR} = \mathsf{FNR} = 0$ for both groups).

This impossibility result has shaped the tone of the modern field of algorithmic fairness: researchers are aware that there will be no mathematical panacea that will cure our social ills. Some parts of the field attempted to set guidelines for when each fairness metric should be chosen, and what value each of those encoded. This runs into problems quickly, as we often wish to meet multiple criteria simultaneously and feel uncomfortable with the tradeoff between them. The example of the COMPAS recidivism prediction algorithm is well-known in the algorithmic fairness community. Much modern interest in algorithmic fairness was spurred by a ProPublica article decrying COMPAS for having unequal false positive rates and false negative rates between Caucasian and African-American defendants (African-Americans were more likely to be labeled higher risk but not re-offend, an unequal false positive rate; Caucasians were more likely to be labeled lower risk yet did re-offend, an unequal false negative rate) [11]. Yet COMPAS satisfied a different metric of (approximate) groupwise calibration and, with thresholding, equalized positive predictive value [81]. Neither metric is obviously superior to the other: The existing system which equalizes PPV but leaves FPR and FNR unequal feels intuitively unfair – if an African-American and a Caucasian defendant were each scored, the chance of an incorrect result would be higher for the African-American than for the Caucasian. But a system which equalized FPR and FNR at the cost of PPV would essentially mean that a score of "high risk" would represent a different probability of re-offense for a Caucasian and an African-American defendant. Lest this seem less catastrophic, notice that a judge who learned only the "high-risk" label choosing whether to release the defendant or not, and whose goal is to release those below a certain probabilistic threshold of re-offense, would have to treat the two differently (see more in [229]). This also feels unfair. Unfortunately, the underlying logical constraints force us to choose between the above two scenarios, a perfect classifier, or equalizing the measured base rates of recidivism between the two groups.

One final note on this topic: Although the base rates of the input dataset were unequal between races, there are any number of biased processes that could have led to this input state. The data itself may have been collected in a biased fashion. Reducing unfair discrimination in all parts of this process is important. This work focuses on the classifier only: we assume our data are the most accurate available (the WYSIWYG assumption of [327]), and focus on the question of how to do the best we can with what we have.

The task of fairly post-processing groupwise-calibrated classifiers

In §3, we concentrate on the task of post-processing a *groupwise-calibrated, non-binary* classifier under group fairness constraints. We suppose that individuals belong to one of two or more disjoint *protected groups.* Our overall task is to decide whether a given individual has some hidden binary property B in a way that ensures "fair balancing of errors" across the groups.

For that purpose, we consider the following two-stage mechanism. The first stage consists of constructing a classifier $\hat{S}$ that outputs for each individual x a "soft" nonbinary score between 0 and 1, $s \in [0, 1]$, representing the classifier's guess at the probability that x has property B. The only requirement we make of $\hat{S}$ is *groupwise calibration*: within each group, and for each $s \in [0, 1]$, the fraction of individuals in the group that get score s and have the property, out of all individuals in the group that get score s, is s. The second stage takes as input the output $s = \hat{S}(x)$ of the first stage and the group to which x belongs, and outputs a "hard" binary decision: its best guess at whether x has property B.

An attractive aspect of this two-stage mechanism is that each stage can be viewed as aimed at a different goal: The first stage is aimed at gathering information and providing the best accuracy possible, with only minimal regard to fairness (i.e. only groupwise calibration). The second stage is aimed to extract a decision from the information collected in the first stage, while making sure that the errors are distributed "fairly."

To further focus our study, we take the first stage as a given and concentrate on the second. That is, we consider the problem of *post-processing* the scores given by the calibrated soft classifier $\hat{S}$ into binary predictions. A representative example is a judge making a bail decision based on a score provided by a software package. Following [81, 162] we consider the following four performance measures for the resulting binary classifier: the *positive predictive value (PPV)*, namely the fraction of individuals that have the property among all individuals that the classifier predicted to have the property; The *false positive rate (FPR)*, namely the fraction of individuals that were predicted to have the property among all individuals that don't have the property; The *negative predictive value (NPV)* and *false negative rate (FNR)*, which are defined analogously. Ideally, we would like to equalize each one of the four measures across the groups, i.e. the measure will have the same value when restricted to samples from each group. Unfortunately, however, we know that this is impossible in general [81, 200]. This leads us to a broad question that motivates our work:

> Under what conditions can we post-process a calibrated soft classifier's outputs so that the resulting hard classifier equates a subset of $\{\mathsf{PPV}, \mathsf{NPV}, \mathsf{FNR}, \mathsf{FPR}\}$ across a set of protected groups? How can we balance these conflicting goals?

Post-Processing with thresholds

In a first set of results we consider the properties obtained by post-processing via a "threshold" mechanism. Naively, a threshold post-processing mechanism would return 1 for individual x whenever the calibrated score $s = \hat{S}(x)$ is above some fixed threshold, and return 0 otherwise. We somewhat extend this mechanism by allowing the post-processor to "fine-tune" its decision by choosing the output probabilistically whenever the result of the soft classifier is exactly the threshold.

We first observe that the popular and natural post-processing method of using a single threshold across all groups has an inherent deficiency: No such mechanism can in general guarantee equality of either PPV or NPV across the protected groups.

We then show that, when using different thresholds for the different groups, one can equalize *either* PPV or NPV (but not both) across the two groups, assuming the profile of $\hat{S}$ has some non-degeneracy property.

The combination of the impossibility of single threshold and the possibility of per-group threshold also stands in contrast to the belief that a soft classifier that is calibrated across both groups allows “ignoring” group-membership information in any post-processing decision [229]. Indeed, the conversion to a binary decision “loses information” in different ways for the two groups, and so group membership becomes relevant again after post-processing.

***Deferrals* as a post-processing tool**

For the second set of results we consider post-processing strategies that do not always output a decision. Rather, with some probability the output is $\perp$, or “I don’t know,” which means that the decision is deferred to another (hopefully higher quality, even if more expensive) process. Let us first present our technical results and then discuss potential interpretations and context.

The first strategy is a natural extension of the per-group threshold: we use *two* thresholds per group, returning 1 above the right threshold, 0 below the left threshold, and $\perp$ between the thresholds. We show that there always exists a way to choose the thresholds such that, conditioned on the decision not being $\perp$, both the PPV and NPV are equal across groups.

Next we show a family of post-processing strategies where, conditioned on the decision not being $\perp$, *all four quantities* (PPV, NPV, FPR, FNR) are equal across groups.

All strategies in this family have the following structure: Given an individual x, the strategy first makes a randomized decision whether to defer on x, where the probability depends on $\hat{S}(x)$ and the group membership of x. If not deferred, then the decision is made via another post-processing technique.

One method for determining the probabilities of deferrals is to make sure that, the distribution of scores returned by the calibrated soft classifier *conditioned on not deferring,* is equal for the two groups (That is, let $p_{s,g}$ denote the probability, restricted to group g, that an element gets score s conditioned on not deferring. Then for any s, we choose deferral probabilities so that $p_{s,g_1} = p_{s,g_2}$.) The resulting classifier can then be post-processed in *any* group blind way (say, via a single threshold mechanism as described above).

Of course, the fact that all four quantities are equalized conditioned on not deferring does not, in and of itself, provide any guarantees regarding the fairness properties of the overall decision process — which includes also the downstream decision mechanism. For one, it would be naive to simply assume that fairness "composes" [112]. Furthermore, the impossibility of [81, 200] says that the overall decision-making process cannot possibly equalize all four measures.

However, in some cases one can provide alternative (non-statistical) justification for the fairness of the overall process: For instance, if the downstream decision process never errs, the overall process might be considered "procedurally fair." We present more detailed reflections on our deferral-based approach in Section 3.7.

We note that deferring was considered in machine learning in a number of contexts, including the context of fairness-preservation [217]. In these works, the classifier typically punts only when its confidence regarding some decision is low. By contrast, we use deferrals in order to "equalize" the probability mass functions of the soft classifier over the two groups, which may involve deferring on individuals for whom there is higher confidence. Indeed, deferring on some higher-confidence individuals seems inherent to our goal of equalizing PPV, NPV, FPR, and FNR while keeping the deferral rate low. Furthermore, our framework allows for a wide range of deferral strategies which might be used to promote additional goals.

Experimental results

We test our methodology on the Broward county dataset with COMPAS scores made public by ProPublica [11] in order to better understand its strengths and limitations. Indeed, it has been shown that the COMPAS scoring mechanism is an approximately calibrated soft classifier [81,124]. We first ran our two-threshold post-processing mechanism and obtained a binary decision algorithm which equalizes both PPV and NPV across Caucasians and African-Americans. In addition to minimizing PPV and NPV, this method also resulted in the desirable property of deferring when the classifier's output is close to uniformly random, i.e. the classifier did not have much confidence in its result.

We then ran our post-processing mechanism with deferrals to equalize all four of PPV, NPV, FPR, FNR across the two groups, with three different methods for deciding how to defer: In the first method, decisions are deferred only for Caucasians; in the second, decisions are deferred only for African Americans; in the third method, decisions are deferred for an equal fraction of Caucasians and of African Americans. This fraction is precisely equal to the statistical (total variation) distance between the distributions of scores produced by the soft classifier on the two groups. For all of these methods, any group-blind classifier would achieve all fairness guarantees on the un-deferred outputs. The shape of the deferrals themselves was very different across these three methods. Due to the shapes of the input distributions, the last method resulted in deferring only on low-risk Caucasians and high-risk African-Americans, which creates a challenge for designing an appropriate deferral mechanism. More details about the results along with figures are given in Section 3.6.

Algorithmic fairness is inherently interdisciplinary

Algorithmic fairness shares several qualities with cryptography: much of its purpose is to accomplish some goal (run a classifier, transmit a message) while ensuring some additional property on that goal is met (statistical fairness notions, ensuring the message is not read or tampered with). It shares a need to capture a wide set of "adversaries" and limit their abilities – although this notion is more widespread in cryptography than in fairness, fairness has enough overlap with data analysis that many fairness papers adopt similar practices of treating the "analyst" as an "adversary" who, instead of trying to learn as much as possible about the individuals in a dataset, will try to justify a biased decision on the output of a classifier. It additionally shares a need to model actual humans' behavior in a formalized way that can be mathematically analyzed, and both occasionally draw on relevant concepts from economics or game theory.

And, like cryptography, a portion of the field tests existing systems for flaws, for example analyzing deployed algorithms for bias or finding ways to "remove" bias from a decision at the cost of destroying information. Recently, there has been a positive push toward using algorithmic fairness methods to detect discrepancies in training data rather than using it to post-process a classifier (e.g. [75, 166, 257]). For example, if facial recognition algorithms have significantly lower accuracy on one group than another, this tells the designer of that algorithm that they must train their algorithm on more faces in that group [66]. (Other works in the space object to facial recognition for purely normative, ethical, or governance reasons unrelated to the accuracy discrepancy.)

Algorithmic fairness is a field that by design interacts heavily with law, policy, and ethics. Governments providing anti-discrimination laws, or companies justifying their actions, are in great need of this research to inform the limits of what is possible, and how to approach those limits as closely as possible. As mentioned before, although the statistical analyses of algorithmic fairness are not limited to algorithms, the increasingly widespread use of homogeneous algorithms brings new urgency to the need for formalized analysis of these questions, which interdisciplinary research is in a good position to provide.

1.2.3 Improvements to zero-knowledge argument systems

Chapter §4 of this thesis is not inherently interdisciplinary work; instead it describes improvements to a standard cryptographic primitive (zero-knowledge proofs) by a standard cryptographic performance metric (proof size). At a high level, zero-knowledge proofs allow a prover to convince a verifier of the truth of a statement without revealing the witness that shows why that statement is true.

Interest in zero-knowledge proofs has grown in the legal sphere, between cryptocurrency, smart contracts, and deployments in commercial settings. They have been used in areas spanning from digital watermarking [2] to nuclear armament verification [141]. We describe these areas of legal interest in much greater detail in §4.1.

Our work in §4 lessens the tradeoff between privacy and utility: by making these proofs better, we improve their chance to be used in legal and societal settings.

TurboIKOS: improved MPC-in-the-head

The focus of the work in §4.2-§4.6 is when both public verifiability and low RAM utilization are required and a linear proof size is acceptable. In this setting, the best available constructions are based on the "MPC-in-the-head" paradigm developed by Ishai et al. [177]. These proofs are constructed by executing a secure multiparty computation (MPC) protocol, which only requires fast symmetric key crypto operations and is amenable to the Fiat-Shamir transform [120]. As a result, proofs in the

MPC-in-the-head paradigm form the basis of the Picnic digital signature scheme that is currently an "alternate candidate" in round 3 of the NIST post-quantum crypto competition [5, 76, 188, 224].

These sections contribute a new zero knowledge proof in the MPC-in-the-head paradigm that provides *concretely smaller proof sizes* than prior work for some parameters settings. Our construction, called TurboIKOS, retains the benefits of all constructions in the MPC-in-the-head paradigm: low RAM utilization, public verifiability, avoiding structured setup, prover and verifier runtime that are linear in the circuit size $|C|$, and the ability to make the proof non-interactive via the Fiat-Shamir transform.

We describe two variants of TurboIKOS, both of which operate over an NP relation encoded as an arithmetic circuit C over a large field $\mathbb{F}$. The first version is an improvement over Baum-Nof [25] that reduces the number of field elements sent per gate from 4 to 3, and is intended for circuits with large field size (Section 4.4.3). The second version further reduces the number of field elements sent per MUL gate from 3 to 2, and uses a modified batched consistency check that allows the technique to be used in smaller fields (Section 4.4.4).

BooLigero: improved sublinear zero-knowledge proofs for Boolean circuits

In §4.7-§4.11, we present BooLigero, an improvement to the sublinear (but not log-size) ZK proof Ligero [10] tailored for Boolean circuits. Our method allows us to utilize the "full" field element and store $\log |\mathbb{F}|$ bits of the witness per element, rather than storing only a single bit per (larger) field element and enforcing an additional constraint as is required in Ligero. We can utilize the full field for XOR and NOT operations; for AND we can use $\sqrt{\log |\mathbb{F}|}$ bits of the field element. This buys us an improvement in the proof size between $O((\log |\mathbb{F}|)^{1/4})$ and $O((\log |\mathbb{F}|)^{1/2})$ compared to original Ligero, depending on the proportion of ANDs in the circuit. The prover and verifier runtime should not change much compared to original Ligero. We do this while maintaining Ligero's properties of being public coin, perfect honest-verifier zero knowledge, amenability to the Fiat-Shamir heuristic, being plausibly post-quantum secure in the standard model, and requiring no trusted setup.

Our primary tool is efficient zero-checking and tests for constraints such as if the bits follow a certain well-defined pattern. In Ligero, the witness is encoded, and constraints are checked by ensuring that the prover's claims are consistent with parts of the encoded witness that were randomly chosen by the verifier. We add the ability to reveal masked elements of the witness directly, in such a way that the verifier may check properties on the masked elements that will enable them to test properties of other hidden witness elements. Tests with a certain kind of linearity are extremely efficient, requiring only a constant overhead in the number of witness elements to test arbitrarily many instances of the property on existing variables. This enables us to test properties that would normally be difficult to test while representing many bits per word, such as testing whether certain bits are zero, or testing bit "patterns" such as masking and shifting. We can also use these to build range tests. These tests may be helpful in frameworks outside BooLigero as well.

We evaluate our performance on the hash functions SHA-3 and SHA-2, which are common benchmarks and have particular appeal to the cryptocurrency community. We achieve a 1.7-2.8× improvement over Ligero for Merkle trees of SHA-3 from 2^1 to 2^{15} leaves. Our circuit for SHA-3 utilizes one of our specialized tests to perform the bit-rotation step of the SHA-3 main loop. For SHA-2, we achieve a 1.1-1.6× improvement over Ligero for Merkle trees from 2^1 to 2^{15} leaves. Note that this is in spite of the fact that SHA-2 uses some addition modulo 2^{32} operations, which Ligero supports directly and BooLigero does not.

1.3 Organization

§2 contains a cryptographic and legal analysis of compelled decryption under the Fifth Amendment and the privilege against self-incrimination. Our modeling allows analysis of a variety of cases under the foregone conclusion doctrine, not only encryption cases. We additionally provide a new security definition for cryptographers who are seeking to create protocols that do not leave one vulnerable to the threat of being compelled by the government to reveal something about the protocol.

§3 is an algorithmic fairness analysis of a common problem in which a non-binary classifier must be post-processed into a binary decision. Our analysis has two major impacts on policy. First, we present impossibility results that show when certain goals are inherently in conflict with each other and cannot be achieved together. Second, we present a "deferral" mechanism that, while it does not (indeed, cannot) achieve our statistical fairness criterion on the entire set of inputs, it allows achieving it on a subset of inputs and uses a different societally-acceptable method to decide on the remaining inputs.

In §4, we turn to a more traditional-cryptographic presentation of two improvements to zero-knowledge proofs. In §4.1, we motivate the improvement of zero-knowledge proofs by discussing legal tasks that zero-knowledge proofs are well-suited for, and in §4 we dive into the improvements themselves. Both improvements are focused on reducing the concrete proof size, an important consideration if long-standing records are to be kept. One improvement focuses on MPC-in-the-head style proofs [177], one on Ligero [10].

Finally, we conclude in §5 by providing background and motivation for the growing field of interdisciplinary law-computer science (especially law-cryptography) research in general, and by describing some areas of cryptography-law research we believe will be especially fruitful in the coming years.

Chapter 2

Cryptography and the Fifth Amendment privilege against self-incrimination

This chapter is based on joint work with Mayank Varia [261].

2.1 Introduction

As we described in §1.2, this work uses cryptography to address the legal question of what actions the government may compel a respondent to perform while abiding by the Fifth Amendment of the U.S. Constitution. The Fifth Amendment enshrines the right to avoid self-incrimination in the U.S.; it states in part that "[n]o person . . . shall be compelled [by the government] in any criminal case to be a witness against himself" [300]. Over the last decade, a number of cases have risen through the courts which ask the following question: if the government seeks as evidence a computer file that is encrypted using a key derived from a password, *can the government compel the device's owner to use her password in order to decrypt the file?*

If the analysis ended there, it would seem that the answer should be "no": asking a respondent to decrypt the file would force the respondent to incriminate herself, which the respondent could refuse to do under the Fifth Amendment privilege. However, over the years a number of limitations have been applied to the Fifth Amendment privilege. Most importantly, the Fifth Amendment only applies to *testimonial* actions, which depend non-trivially on the contents of the respondent's mind. For example, writing one's own name [165], providing a handwriting exemplar [139], or providing a blood sample [263] are all examples of actions which may be compelled as an exception to the Fifth Amendment privilege.

Courts and legal scholars are split as to how to treat compelled requests to enter or state passwords under the Fifth Amendment (e.g. [85, 175, 193, 198, 222, 256, 282, 302, 317]). In this work, we cryptographically model the relevant legal doctrine, which we then use to reason that the answer to the compelled decryption question should often be "no." We test our model by ensuring that it has the same outcome on all non-encryption-related cases regarding this legal doctrine in the U.S. Supreme Court, and five key Circuit Court cases. Additionally, we write a security definition capturing the conditions under which the secrets used in a cryptographic protocol avoid the threat of compelled disclosure in our modeling, and we construct and implement a 2-party protocol secure under this definition based on Yao's garbled circuits [325] using authenticated garbling [313] or cut-and-choose [213, 238] to achieve malicious security.

2.1.1 Related work

Here, we focus on law-focused related work, encryption schemes that enable governments to execute search warrants in the presence of encryption, and cryptography to prevent government overreach outside the specific context of the crypto wars. We will discuss work on other forms of legal-cryptographic modeling in §5.3.

Legal analyses of compelled decryption. To our knowledge, Cohen and Park [85] is the only legal analysis of the foregone conclusion doctrine by authors with cryptographic expertise; their expository work describes several legal concepts and how they fare against technological advances such as widespread use of deniable encryption or hardware kill switches. Additionally, there exist several normative works by law scholars whose reasonings (which often analogize encryption to other security mechanisms like safes or shredders) lead to very different conclusions. Winkler [317] argues that the right against self-incrimination prevents the government from compelling a respondent to use her passwords in any way. Kerr [193], McGregor [222], and Terzian [282] make distinct yet related arguments that the government should have the power to compel decryption in order to restore balance between government powers and civil liberties in light of modern encryption's strong confidentiality guarantees. Kiok [198] and Sacharoff [256] settle somewhere in the middle, only allowing the government to compel decryption if they already know certain aspects of the targeted files with "reasonable particularity." Unlike all of these works, our definition is rigorous, composable, applies directly to encryption rather than using an analogy, and is easier for the scientific community to analyze when evaluating the security of a new system.

Existing case law. Analyses of compelled decryption are timely because courts in the United States are currently divided on the issue. Some courts say people can be compelled to disclose passwords themselves (e.g. [273]), some say they can be compelled only to enter the password but not reveal it (e.g. [90, 91]), and others say that only specific files already known to the government can be compelled (e.g. [266]). See §2.4.1 for a summary of these rulings. Also in that section, we provide a description of previous cases the U.S. Supreme Court has rejected concerning compelled decryption via the foregone conclusion doctrine, and our prediction for future such cases.

Cryptography for search warrants. Several prior works consider using cryptography to enable governments to execute search warrants where encryption is involved. Smith et al. [271] and Feigenbaum et al. [117] discuss broad principles for this topic. Specific encryption schemes with key escrow have been proposed since the 1990s [107], and more recent proposals combine cryptography with trusted hardware so that a device manufacturer can assist law enforcement in decryption [62,258,278]. There also exist MPC-based constructions that provide more fine-grained functionalities like auditable threshold decryption [77,204] and private set intersection across private companies [265]. Bellovin et al. [29,30] sidestep cryptography altogether and look to lawful hacking as a resolution to the Crypto Wars.

Cryptography to prevent government overreach. Conversely, several prior works use cryptography to limit government overreach technologically. Tyagi et al. [291] provide "self-revocable" encryption in which a user can temporarily revoke her own ability to access her secret data for the purpose of defending against temporary compelled decryption threats such as border crossings. Traffic unlinkability tools like Tor [110] protect against traffic analysis by governments, and encrypted search techniques can be used to limit collection of metadata stored at rest [186,320]. There are works that protect against subversion by the government (or anyone else) for encryption schemes [170], digital signatures [15], and hash functions [20]. None of these works consider compelling the respondent to perform the decryption in a court setting, and as we show in §2.5.5 security against that threat is reliant upon the government's inability to get the data some other way.

2.1.2 Remarks

We hope this work provides worthwhile designs of cryptosystems that withstand government compelled requests, inspires the community to include this threat when designing secure systems, and casts new light on the value of passwords as a useful protection against this threat. That having been said, we make several remarks to clarify the context of this work.

First, it is difficult to judge the accuracy of any legal definition in a common law system. We show the best possible evidence: that our definition is consistent with established precedents by the Supreme Court and the appellate courts (§2.4) and that it adheres to the core principles of the Fifth Amendment (§2.3.3). Nevertheless, subsequent decisions by the courts might strengthen or restrict government power in a way that renders our definition moot. Even if this should happen, we believe that our paper provides enduring value by showing a methodology to reverse-engineer a formal definition from common law.

Second, this work only captures a subset of legal cases, albeit a subset that we believe is useful. We presume that the government tells the truth when interacting with the court system, although we do not presume that the government tells the *whole* truth. This work only considers the right to silence as interpreted in the United States. Additionally, this work considers self-composition of compelled requests (§2.3.5) and reasons how compelled requests compose with the government's own decryption capabilities (§2.5.5), but we do not consider how the Fifth Amendment itself composes with other aspects of the law. We acknowledge that this gap can introduce two-sided error: actions that are permissible under the Fifth Amendment might be refuted on other grounds, and conversely, information protected under the right to silence might be accessible to the government via other means.

Third, we stress that this work only focuses on security against one specific threat: that of compelled action by the government. Therefore, the threat model in this work is necessarily incomplete and potentially counterproductive if protections against government compelled requests conflict with protections for other threats. For this reason, we prove the constructions in this work secure under their traditional definitions in addition to analyzing their resilience to foregone conclusion requests (§2.6). We hope that this work inspires the information security community to consider government compelled actions within scope in their threat models.

2.1.3 Organization

This section is structured as follows. In §2.2, we describe the body of law known as the foregone conclusion doctrine, which is the deciding factor in most decryption cases. In §2.3, we provide our formal definition of a foregone conclusion and analyze its properties, including sequential composition. In §2.4, we compare our approach to legal scholarship and justify our definition by showing that it comes to the same outcome as U.S. Supreme Court and Circuit Court decisions. In §2.5, we analyze the compellability of common cryptographic primitives under the foregone conclusion doctrine. In §2.6, we explore the extent to which voluntarily participating in a cryptographic protocol leaves one more vulnerable to future compelled requests; we call a protocol *resilient* if any compellable action after running the protocol was already compellable before running the protocol.

2.2 Overview of the foregone conclusion doctrine

In this section, we provide a brief overview of the Fifth Amendment to the United States Constitution (abbreviated “5A”). We emphasize one aspect of 5A law called the foregone conclusion (FC) doctrine, which is the crux of all compelled decryption cases.

The right to silence as an interactive protocol. The right to silence in the United States involves three parties: a government actor G such as a prosecutor or law enforcement officer, an individual *respondent* R of the compelled request, and a neutral court. We use the term respondent rather than “suspect” or “defendant” because people can be compelled to perform government actions even without being accused of a crime, and we consider individuals because companies do not have Fifth Amendment rights. Also, we stress that this work focuses on G’s compelled requests to R, not G's powers or restrictions to search for information on its own.

Compelled requests follow a 3-round interactive protocol: first G issues a subpoena asking R to respond to a query, then R responds by asserting her right to silence, and finally G requests that a court compels R to answer the query anyway. For the court to approve the government's request to "override" the respondent's right to silence, the burden of proof falls on the government to demonstrate that the compelled request is not covered under the respondent's rights [172].

The Fifth Amendment's protections are broad but not absolute: they only apply to government requests that are compelled, incriminating, and testimonial [123]. The first two properties are relatively simple to describe. First, 5A cannot retroactively protect statements that R has previously provided voluntarily to the government, the statement must have been compelled by the government in the first place. Second, 5A can only be invoked if the compelled statement would "furnish a link in the chain of evidence needed to prosecute the claimant for a federal crime" [168]. Many cases involve compelled actions that are non-incriminating because the government has granted R immunity from prosecution [292]; 5A does not apply in these cases since the respondent cannot be prosecuted for the crime at hand. Some other specific actions, like being compelled to state one's own name, have also been decided to be non-incriminating [165]. Because these two criteria are usually simple to verify, throughout this work we assume that all parties agree that G's request is compelled and potentially incriminating.

Testimony. We focus in this work on the final requirement: the government is only restricted from compelling people to perform acts that are *testimonial*, meaning that they "disclose the contents of [the respondent's] own mind" [98]. Based on this principle, speaking your password to the government is testimonial [296], but providing a blood sample [263] does not rely on any mental state of R, so it is non-testimonial and therefore compellable under 5A.

The law protects *pure testimony* in which the written or spoken output of a compelled request directly reveals information about R's mind, and *implicit testimony* in which the government can infer something within R's mind by "relying on the truthtelling" [123] of the respondent when performing an action C and producing the result. In implicit testimony, the *act of production* is the testimonial object in question, not the *contents* produced. For instance, G cannot compel R to provide written documents (whose contents are *not* 5A-protected) if G must rely upon "the respondent's truthful reply [to receive] the incriminating documents" [295]. If R provides the documents upon request, then R's act of producing them testifies to (at least) the existence of the documents, as well as R's possession of them and her belief that they are authentic [123]. Pure testimony is always forbidden within compelled requests, although implicit testimony need not be; for this reason, we focus on implicit testimony in this work.

We emphasize that only the testimonial aspects of a compelled request C are covered under 5A. The output of C might reveal more or less information than the implicit testimony implied by it, but only the latter is protected. For example, suppose that G compels R to provide all documents sitting in plain sight within her locked office. Whether the documents themselves are incriminating is irrelevant; R only has the right to withhold from G the implicit testimony revealed by executing C, i.e., the knowledge in R's mind implied by her truthful response. In this example, the only implicit testimony from the compelled action is that R has the ability to access her own office. There is no ambiguity as to the choice of documents themselves, and thus no testimonial aspect – the government *could* have sent someone to break into her office and collect the documents themselves, without relying on R. So the only testimonial aspect of this compelled request is R's ability to access her office. If

G already has evidence that R knows the location of her office key, then executing C would not reveal any *new* implicit testimony to G. This begs the question: does it violate R's rights for G to compel R to implicitly testify to a statement that G already knows to be true?

The foregone conclusion doctrine. The U.S. Supreme Court case *Fisher v. United States* answers the above question in the negative, thereby providing a power to the government that can counter R's invocation of the right to silence. The *Fisher* case says that the courts can compel R to execute an action C if its implicit testimony is a *foregone conclusion* to the government, in the sense that it "adds little or nothing to the sum total of the Government's information" [123]. Concretely, the law enumerates several blacklisted predicates: if G would learn about the existence, location, or authenticity of any new evidence from its interaction with the respondent, then the compelled action is *not* a foregone conclusion.

This work starts from the premise that simulation-based cryptographic definitions can dovetail with the concepts within the foregone conclusion doctrine for 3 reasons. First, simulatability formalizes the concept of "not learning new evidence" [146, 212]. Second, simulation sidesteps entirely the task of enumerating sources of implicit testimony; instead, it holistically determines whether all implicit testimony present in a compelled action C is a foregone conclusion. Third, whereas predicate blacklist-based definitions often allow a series of individual requests that might be deemed to be invasive in totality, we will demonstrate security under composition.

2.3 Rigorous definition for a foregone conclusion

The crux of the foregone conclusion question is how to know when the government is "relying" on the contents of the respondent's mind, when compelling her to perform an action? This work uses the cryptographic concept of *simulation* to codify the idea that running a foregone compelled action "reveals nothing" to the government about the respondent's mind, above and beyond what the government can learn from the rest of the world.

In this section, we provide both informal and then rigorous descriptions of our security game that encapsulates the foregone conclusion doctrine. Additionally, we prove that our definition remains secure under sequential composition.

2.3.1 Informal walkthrough

In this section, we provide an informal description of our game-based definition of the foregone conclusion doctrine. Our game proceeds interactively between the government and respondent to determine whether an action is (or is not) a foregone conclusion. We abstractly represent all of the information in the rest of the world (outside of the respondent's mind) as "Nature." We also assume as a pre-condition that the government and the respondent have already agreed on the evidence E of the case.

The government acts first in our game. It declares a compelled action C that it wants the respondent to perform; this action may make use of both the respondent's mind and Nature. (For interactive protocols, the government must also output a second machine G codifying the government's response to each message from C.) We stress that C represents the *act of production*, i.e., the process of obtaining the result rather than the result itself. The government has the burden of proof to demonstrate that its compelled request is a foregone conclusion, as required by the courts [193]. The government submits this proof in the form of a simulator S that tries to output the same result as C without access to the respondent's mind but with the significant power to view anything else in the world.

Second, the respondent has an opportunity to demonstrate that the compelled action depends non-trivially on her own mind. To do this, she must equivocate: specify a "world," comprising Nature N and the contents of her own mind R, where the simulation disagrees to match the compelled action with non-negligible probability. This world must be consistent with the evidence, or else the respondent loses our game.

Third, we run a thought experiment to test whether the government's uncertainty about the state of the world is too high for the compelled action to be deemed a foregone conclusion. Concretely, we run the compelled action and the simulation, and we ask a distinguisher to attempt to tell the two results apart. If the results are indistinguishable, then we declare that any implicit testimony in the compelled action is a foregone conclusion on top of the existing knowledge already available to the simulator (i.e., everything in the rest of the world). If the results are different, then we declare that the government is relying too much on the truthtelling of the respondent for the compelled action to be deemed a foregone conclusion. The government could try again with a different compelled action, an improved simulation strategy, or more evidence; any of these options would cause the game to begin anew.

2.3.2 Formal definition

In this section, we formally define a foregone conclusion. We emphasize that the evidence should be sufficient so that a *single* government simulator S can simulate the response of *any* respondent R that acts consistently with the evidence; that is, the choice of S cannot depend on the contents of any specific respondent's mind.

Summary of participants. We describe below several components in our model: a string representing nature, and several interactive Turing machines (ITMs) in the manner formalized by Canetti [69].

The runtime and behavior of all machines are affected by a parameter λ. Usually this parameter refers to "the amount of evidence"; we will explain this further both in this section and in §2.3.4. Machine M has a time bound of $t_M(\lambda)$ (for example, the simulator S has a time bound of t_S). For machines that can query N, they have a query bound of $q_M(\lambda)$.

In a typical court setting, defining the machines' time and query bounds on only the singular setting of λ that is relevant to the case at hand (i.e. using concrete bounds) is sufficient. For many sequential court rulings that must compose with each other, or for security researchers attempting to design FC-resilient systems, an asymptotic treatment of these bounds (where the machines are poly-time and have a polynomial number of queries in λ) is likely to be more useful. For further discussion of how to set these bounds, see §2.3.4.

We will additionally augment this model with a few special symbols and random tapes that are specific to this work.

First, we list the machines and strings we will use:

- Nature N represents the entire world except the contents of the respondent's mind. It is a string that is exponentially long in λ. with characters from alphabet Σ, where most of the time we expect $\Sigma = \{0, 1\}$.

- Respondent R represents the contents of the respondent's mind. It is called by the compelled action C via methods. (For example, the evidence might enforce the existence of method $R.pw$, and the output of this method will be used in the execution of the compelled action.) R also has a special method called $R.\texttt{Equivocate}$ that can make at most $q_R(\lambda)$ changes to Nature at the beginning of the security game. It has a time bound of $t_R(\lambda)$.

- Evidence $E^N(R)$ restricts R's allowed changes to Nature. E always returns either **true** or **false** within $t_E(\lambda)$ time. In effect, the respondent will automatically lose the security game if the Evidence returns **false** with greater than probability α (which will be 0 in a typical case). The Evidence is the main power the government has to limit the degree to which R may equivocate. E will return **false** unless N and R meet specific properties.

For example, if the evidence is that R has a valid driver's license in her house, E would contain an authentication function for the driver's license, and would return **true** if the portion of N corresponding to R's house contains a chunk that the authentication function recognizes as a valid driver's license. (Note that although naively we might expect E to need to be unbounded in order to check all of N, in reality it only needs to have enough time to check R's `Equivocate` method.)

We recommend a typical setting of λ to be the sum of the queries E makes to N plus the size of the circuit(s) it uses to check R. In the event that λ is set separately, we cap the time bound of E at $t_E(\lambda)$ and the query bound of E to $q_E(\lambda)$. For more on the typical setting of λ as well as other settings, see §2.3.4.

- Compelled request $C^{N,R}$ is the action the government wishes the respondent to perform. C has oracle access to both nature and the respondent (via method calls), and for some compelled actions it is interactive with an additional machine G^N which has access only to Nature. C has time bound $t_C(\lambda)$ and query bound $q_C(\lambda)$, and G has time bound $t_G(\lambda)$ and query bound $q_G(\lambda)$. We denote the transcript of the interaction between C and G as $t(G^N, C^{N,R})$.

- Simulator S^N attempts to reconstruct a transcript t' that is indistinguishable from the real interaction. It has oracle access to N, but it cannot access the respondent's mind R. It has time and query bounds $t_S(\lambda)$ and $q_S(\lambda)$ respectively. These bounds are arguably the most important settings in the entire definition; we discuss proper settings in §2.3.4.

- A distinguisher $\mathcal{D}^N$ receives either the "real" execution $t(G^N, C^{N,R})$ or the "ideal" execution S^N, and attempts to distinguish between the two. If no $\mathcal{D}$ can distinguish between these, the result is a foregone conclusion. $\mathcal{D}$ has time bound $t_{\mathcal{D}}(\lambda)$ and query bound $q_{\mathcal{D}}(\lambda)$. The advantage (the probability with which the distinguisher must succeed past an even chance) is $\epsilon(\lambda)$, ideally a negligible function as described in §2.3.4.

We discuss how to set time and query bounds in §2.3.4.

In the remainder of this section, we describe the general properties of all participating Turing machines and then any extra properties of each individual machine in more detail.

General Turing machine model. For our purposes, all interactive Turing machines contain the following:

- Oracle access to one or two oracles (C, G, S, E, and R during the reading phase all have access to N, which is queried on indices, and C also has oracle access to R).
- A read-only random tape (except for E, which is deterministic).
- An internal read/write working tape.

Respondent. R is a Turing machine that represents the subpoena respondent's actions. It has time bound t_R and may access Nature q_R times. R has an internal working tape, a write-only output tape, and a read-only randomness tape. Because R often acts as an oracle, we presume that it has methods whose existence and interface may be specified in the evidence. It receives these method calls on a read-only oracle-input tape and writes the results to a write-only oracle-output tape. R always has one special method called `Equivocate`, which is used near the beginning of the security game. When R.`Equivocate` is called, R outputs a set of locations and values Δ; N

will be modified at these locations to be these values; i.e. setting $N[i] = x$ for each pair in Δ. E may mandate the existence of additional methods, or verify that the code of specific methods acts in a certain way (e.g. that an internal variable was generated from the correct distribution using the randomness tape).

Evidence. E is a Turing machine that restricts the ways in which R can modify N. It has oracle access to the modified N and takes the source code of R as input. It may require R to place certain values within N: for example, if the government knows that a certain document is in R's house, it requires R to place that document in a location in N corresponding to R's house. Furthermore, it may require R to have methods with certain behavior, for example requiring the existence of a method $R.\texttt{SafeCombo}$ that outputs the combination that will open a safe in N. E might require only the existence of the method, or it might require the method to have certain behavior. As such, E has oracle access to N, and may inspect the code of R to check the method behavior. E is poly-time bounded. Although that prevents detecting elements "anywhere" in Nature, it still allows E to check any *simulatable* existence claim, since the simulator itself is bounded. It is deterministic so that the chance that R and N are "allowed" by the evidence (defined formally in Definition 2.3.2.1) depends only on R and N themselves, and not on E's coins. E can also force a concrete time bound T on R: It can automatically return **false** if R has not completed within time T. We will require two properties of the evidence, as described further in Def. 2.3.2.2: First, the evidence must be *satisfiable* (contrary to the usual computer science meaning of this term, we simply mean that there must exist at least one "allowed" respondent for which E returns true with sufficient probability). This prevents an action from being vacuously foregone. Second, E must be *non-censoring* – E must not prohibit R from modifying additional locations in N it has not checked. (For example, the evidence may check that the documents exist in R's house, but it may not check that nothing *else* exists in her house, except in extreme cases and when all parties in the court agree to this far-reaching evidence.)

Compelled request and government responder. C is the interactive Turing machine representing the action compelled by the subpoena, in the form of an interaction with a government agent G. The compelled request C has oracle access to N and R, whereas G only has oracle access to N. While a simple request could be done in a single round in which C outputs a message to G, this formalism also permits more complex requests which take multiple rounds.

Both C and G have the following tapes:

- A read-only identity tape
- A one-bit activation tape (determining whether it is currently in the process of "executing"
- A read-only (but externally-writable) input communication tape on which it receives messages
- A write-only output communication tape on which it sends messages

We denote the transcript between G^N and $C^{N,R}$ as $t(G^N, C^{N,R})$.

At all times, the input tape of C should match the output tape of G, and vice versa, and their switch bits should always be opposed. When finishing sending a message, the machine always sends a special end of line character "//". We refer to $t(G^N, C^{N,R})$ as the *transcript* of G^N and $C^{N,R}$, consisting of a "log" of the messages sent between the two machines. This is formatted as an alternating string of messages between each machine's output tape, alternating to the other machine at each // character. This transcript is formatted as a concatenation of "id:$\mathsf{message}$//" for each alternating message where id is either C or G, $\mathsf{message}$ is the message written on the output tape by id (received on the input tape of the opposite machine), and // is a reserved end-of-line character. At the end of the transcript (sent by whichever machine sends the last message) is a special $\square$ symbol, reserved at the beginning of the computation. After receiving or sending this symbol, both machines halt. (A computation that consists of two separate computations concatenated together may use a different $\blacksquare$ symbol to denote the end of the separate computations.) As an example, if G sends a and then C sends b, completing the computation, the transcript t is "G:a// C:b// $\square$".

Finally, one should consider individual instances of C and G to be completely unrelated; in other words, they fully securely erase all of their state after the computation.

Government simulator. S is the government simulator. Its goal is to generate a result that is indistinguishable from the exchange between C and G. It runs in time t_S and may access N as an oracle q_S times (and has no ability to query R). S has a single output tape in which it will attempt to simulate the transcript of a paired set of machines, but is not itself interactive. S is prohibited from returning $\perp$ to this output tape.

Composing Turing machines. In §2.3.5 we will also consider machines composed with each other. Let M_1 and M_2 be Turing machines that, upon completion, use the computation-end symbol $\blacksquare$. Then let $M_1 \| M_2$ denote a machine with computation-end symbol $\square$ that begins by running M_1, exchanging messages as normal and ending with $\blacksquare$. Then it starts computing M_2, exchanging messages as normal and also ending with $\blacksquare$. Then, since its computation has ended, it outputs $\square$. In short, $M_1 \| M_2$ denotes the sequential composition of M_1 and M_2.

Allowed respondents. When checking for a foregone conclusion, the respondent is automatically "caught" if it does something that violates the evidence E. We say that R is *allowed* by the evidence if $E\ (R)$ returns **true** with overwhelming probability over the initial random initialization of N.

Definition 2.3.2.1 (Allowed respondents)**.** R is α-*allowed* if

$$\Pr_{N,R}\left[\begin{array}{c} N \leftarrow_{\$} \Sigma^{\exp(1^{\lambda})} \\ \Delta \leftarrow R.\texttt{Equivocate} \\ \textbf{for } (i,x) \in \Delta : N[i] = x \end{array} \quad ; \quad E^N(R) = \textbf{true}\right] > 1 - \alpha.$$

Because all allowed Turing machines *could* represent the real state of the respondent's mind as far as the government is aware, our foregone conclusion definition will require that the government can simulate all allowed R. Conversely, the simulator is only required to succeed on allowed respondents. To avoid degeneracy, the definition will require the existence of at least one allowed respondent.

Security game

```
$\mathsf{Game}_{E,C,G,S,R}(\lambda)$
$N \leftarrow_{\$} \Sigma^{2^{\lambda}}$    // initialize N randomly
$\Delta \leftarrow R.\texttt{Equivocate}$    // $\Delta$ is a set of index-value pairs
// $\Delta$ is a set of changes to $N$
$\textbf{for } (i, x) \textbf{ in } \Delta : N[i] = x$
// check evidence and return $\perp$ if false
$\textbf{if } E^N(R) = \textbf{false} : \textbf{return } \perp$
// return either real or simulated transcript
$\textbf{return } N, \boxed{t(G^N, C^{N,R})}, \boxed{S^N}$
```

Figure 2·1: *Real (solid) and ideal (dashed) foregone conclusion games. Steps without a box are common to both games.*

We specify the real and ideal versions of our security game in Figure 2·1. In the real game, the government interacts (possibly over multiple rounds) with the respondent who executes the compelled algorithm C. In the ideal game, the government's simulator S forges a transcript using only its access to Nature (which has previously been prepared by the respondent). The two games are identical except for the final step. In the last step, the real game (solid box) returns the transcript $t(G^N, C^{N,R})$ of all communications between the government and respondent, whereas the ideal game (dashed box) returns the simulated transcript S^N. Both games also offer oracle access to N.

Next, we provide our formal definition of the foregone conclusion principle in Def. 2.3.2.2. It requires that the government's simulator S faithfully emulates real-world transcripts. Moreover, it limits the respondent R's ability to equivocate and the evidence E's ability to censor R's use of nature.

Definition 2.3.2.2 (Foregone conclusion (FC_λ))**.** Let λ be a security parameter. The exchange between G and C is a *foregone conclusion* with respect to E and S if the following four conditions are met:

1. *Efficiency:* C has time bound $t_C(\lambda)$ and query bound $q_C(\lambda)$, G has time bound $t_G(\lambda)$ and query bound $t_G(\lambda)$, and S has has time bound $t_S(\lambda)$ and query bound $q_S(\lambda)$.

2. *Simulatability:* $\forall$ allowed R that run in $t_R(\lambda)$, $\forall$ $\mathcal{D}$ that run in $t_{\mathcal{D}}(\lambda)$ and make $q_{\mathcal{D}}(\lambda)$ queries,

   $$\left| \Pr[\mathcal{D}^N\big(t(G^N, C^{N,R})\big) = 1] - \Pr[\mathcal{D}^N\big(S^N\big) = 1] \right| < \epsilon(\lambda)$$

   where N, $t(G^N, C^{N,R})$, and S^N are the results of the real and ideal security games defined in Fig. 2·1, and ϵ is the advantage of the distinguisher.

3. *Satisfiability of evidence:* There exists at least one allowed R. Hence, simulatability cannot be vacuously true.

4. *Non-censorship of evidence:* For any allowed R where $R.\texttt{Equivocate} \to \Delta$, all R' where $R'.\texttt{Equivocate} \to \Delta'$ such that $\Delta \subseteq \Delta'$ are also allowed. That is, E does not prevent R from making additional changes to N beyond the locations it checks.

Notice that the probability in the satisfiability requirement is taken over the randomness of R and the random choice of N (modified by R), whereas the probability in the simulatability requirement is taken over R, N, $\mathcal{D}$, C, and S.

Remarks. First, notice that the definition puts the burden of proof on the government as is true in the legal regime [193] by requiring that it construct the simulator S rather than merely asserting that one exists, and by requiring that S is chosen before the respondent R chooses its equivocation strategy. Second, observe that due to the order of quantifiers and R's equivocation ability, if the compelled action C is deterministic then the simulator must match this action exactly. This is explained further in the proof of Lemma 2.5.1.1.

We make several more observations which are expanded upon in §2.3.6. First, the code of R represents the respondent's current actions and limitations in the present (based upon the government's evidence) even if this doesn't correspond to the exact code that the respondent originally executed in the past. Second, because the simulator S can access nature, it doesn't need to forge the *contents* of any documents; rather, it must only forge the *process* of producing them. And third, we presume that the government tells the truth about its evidence.

For more information on how to choose time and query bounds, and how to set λ, see §2.3.4.

2.3.3 Adherence to Fifth Amendment principles

The other parts of this work aim to model what the foregone conclusion doctrine *is*. In this subsection, I describe why this model adheres to what the foregone conclusion doctrine *should be* by the principles of the Fifth Amendment in historical context.

The general attitude against self-incrimination was brought to America from the English common law. The English common law itself developed that attitude as a response to other courts: First, especially early on, the development of English legal norms stood in contrast to the courts of the Inquisition governed by the canon law of the Roman Catholic Church in continental Europe [210, p. 23]. And later, the attitude about self-incrimination further evolved in the late 1500s due to the struggle between Anglican and Calvinist Protestants, both in English ecclesiastical courts and in the Privy Council and Star Chamber which dealt with matters of high political importance to the Crown [191, p. 247]. Inquisitorial courts used a certain oath known as the oath *ex officio*, which required the suspect to respond truthfully to all interrogation. The ecclesiastical court and Council also used similar oaths [210, p. 23]. In the late 1500s and early 1600s, the oath was wielded against Puritan (Calvinist) clergy in England to force them to state their religious conviction to Anglicanism or else face imprisonment, fine, or capital punishment [191, p. 263]. In response, a concentrated campaign against the oath *ex officio* was launched by an alliance of Puritans, common law judges (who resented the expansion of ecclesiastical courts into more secular territories), and members of the House of Commons (especially Sir Edward Coke, who opposed it not on religious grounds but as a matter of governance believed that such an oath could only be used if approved by Parliament) [191, 210]. In 1641, the first formal version of the right against self-incrimination entered English law [191, p. 285], and in 1677 this was imported into law in the colony of Virginia [210, p. 406], from which it was eventually smithed into the Fifth Amendment to the U.S. Constitution.

In all three courts that used some version of the oath, a central common thread is that the oath was used to compel people to share their private thoughts, especially religious beliefs. Puritans were interrogated about their adherence to Anglican doctrine [191, p. 263], and heretics of various kinds were interrogated about their secret writings or secret meetings with other potential heretics in a way that incriminated them both [210, p. 34-35]. Especially in Inquisitorial courts in earlier years, a high requirement of proof (intended to help the innocent) created a perverse incentive to obtain the suspect's full confession at any cost: since most heresies would not have multiple witnesses or documentary evidence, the only acceptable evidence in many cases was confession [210, p. 26-27]. The debates at the time centered around whether direct interrogation for self-incriminating testimony was morally permissible. Debates about the more subtle "indirect" self-incriminating testimony resulting from subpoena for compelled documents or actions did not arise until this first principle was firmly established in both England and America.

In modern American interpretations, we think of the Fourth and Fifth Amendments as acting separately. As discussed earlier in this work, the Fourth Amendment governs what the government can *do to you*, and the Fifth governs what the government can *make you do.* This interpretation was not always the case, and there was in fact a large period of American case law in which the opposite assumption was held: the Fourth and Fifth Amendments were meant to be understood in conjunction with each other, defending all of a person's "private papers" from subpoena *or* government search and seizure [59, 203]. This interpretation formally began with *Boyd v. U.S.* [59] in 1886. This protection was eroded in subsequent cases for corporations and organizations [159, 299], and for papers which the respondent was legally required to hold [269]. The biggest turning point occurred in 1966 in *Schmerber v. California*, in which the Fifth Amendment's coverage was restricted to "testimonial or communicative" information [263, l. 761], allowing the collection of a blood sample under the Fifth Amendment. This set the stage for *Fisher* [123] in 1976, which found that acts of production had "implicit" testimonial aspects, but which allowed compelling that production anyway if that implicit testimony was already known to the government and "the question was not of testimony but of surrender" [123, 220]. Thus, the foregone conclusion doctrine was born, and Boyd became essentially defunct [123, 203].

With this context in hand, what is the "right" interpretation of the foregone conclusion doctrine? Consider a choice between three main alternatives: First, a strict *Boyd*-like interpretation that a defendant may be compelled to bring nothing at all, not even documents that the government knows the defendant must possess in a known location, and can authenticate. Second, the existing foregone conclusion doctrine which we believe is accurately modeled by our simulation-based definition: the Government must demonstrate that it knows all testimonial information implicit in the act of production – that is, the act "adds little or nothing to the sum total of the Government's information" [123, l. 411] – and if this is the case, then the respondent may be compelled to perform the action. Third, a weaker interpretation of the respondent's protections under the foregone conclusion doctrine that focuses on the "verifiability" of the action: if the government does not "rely on the truthtelling" [123, l. 411] of the respondent in performing the action, then it may be compelled. (This last interpretation, unlike our simulation-based definition, might for instance allow a compelling a hash preimage, since the veracity of a computationally-bounded respondent could be tested using the hash function. For more discussion on the differences between these two models, see [86].)

The first option (the respondent cannot be compelled to any action) has been soundly rejected by the cases that over time rejected the *Boyd* interpretation. As the courts in these cases noted [123,263,269], the principles of the Fifth Amendment, once extricated from those of the Fourth Amendment, protect *testimony*, not every single thing about the person. This reasoning is in keeping with the objections to the oath *ex officio* and the other arguments against self-incrimination even back into the English common law: the problem is "confession" of beliefs or facts that the government would not otherwise know.

At the same time, though it has not been much discussed in the literature (old or new), the third option (verification rather than simulation) also goes against those core principles. In interrogatories for secret meetings, or compelled confessions to locations of secret texts, there were surely cases in which the government could check the veracity of the respondent's claims – but only after the respondent made those claims. The first-principles approach to the Fifth Amendment would seem to reject this approach for implicit testimony as well as explicit testimony, for this approach forces the respondent to provide information that the government could not get any other way. This remains true even if the government can independently authenticate the truth of the respondent's response after it is given.

As an aside, we furthermore reject a fourth option in which only explicit testimony (e.g. oral or written) is covered by the Fifth Amendment and any implicit testimony can be compelled. We expect this is uncontroversial – although the question does not arise in the early period of the Fifth Amendment's formation, we expect that legal scholars of both ages would agree that the distinction between "explicit" and "implicit" testimony would be meaningless if any implicit testimony could be compelled. An order to "raise your hand if you committed the crime" is implicit but clearly may as well be explicit, and in the same way, an order to "produce any incriminating papers if you have them" puts the respondent at the same dilemma of perjury or self-incrimination as an oral question. The foregone conclusion doctrine avoids this problem purely because the implicit testimonial information is already known to the government; the respondent does not provide any new power or evidence that the government did not demonstrate that it already could have had.

With this in mind, the foregone conclusion doctrine is best seen as a necessary concession to the reality that the government is not all-powerful. An all-powerful government that could costlessly retrieve any papers from the respondent would have no reason to subpoena the respondent to produce them herself. As described, the compelled action of producing the papers adds nothing to the sum total of the government's information. The question "is not of testimony but of surrender" [123,220]. One could object that forcing the respondent to bring those known papers anyway is an act against the respondent's dignity, even if the act is non-testimonial. To any degree which this is true, this does not even approach the loss of dignity of being forced into a confession without which the government would have had no case. And since the testimonial aspects are not at issue, the foregone conclusion doctrine remains adherent to the principles of the Fifth Amendment. We concede that the need to have a moderately functional justice system supercedes the minuscule loss of dignity that the respondent faces by responding to subpoenas, while the Fifth Amendment remains intact to prevent the much larger loss of dignity that would from giving up *testimonial* information.

In this view, the fact that a whole class of compelled decryption cases seem to hinge on the foregone conclusion doctrine being interpreted in a manner that is less beneficial to the respondent should be a red flag for that interpretation. In contrast, our simulation-based definition forces the government to demonstrate its knowledge of any testimony implicit in the action before the action can be compelled, thus ensuring that "no Constitutional rights are touched" [220] while making the concession that the government is not all-powerful and should be able to compel the respondent to take actions as long as those rights are not touched. It finds that, in the absence of other evidence, there is testimonial information being given up by the respondent in compelled decryption (see §2.5), since the government is not capable of simulating that action even given access to everything in the entire world except the respondent's mind. We preserve the government's ability to compel actions when they can be simulated using only non-testimonial information outside the respondent's mind, while ensuring that no testimonial information is compelled.

We conclude that our simulation-based definition of the foregone conclusion doctrine adheres to the Fifth Amendment principle which holds testimony in the highest regard and prevents defendants from “confessing” new testimonial information that would incriminate themselves. We reject alternative formulations for not adhering to that principle, or for taking it too far and rejecting non-testimonial information as well. Happily, it seems that the definition that we believe is the one in place happens to also be the definition that is most in keeping with Fifth Amendment principles.

2.3.4 Choice of parameters

In this section we describe how one might choose the time and query bounds t and q for all the components of the definition.

Choosing the parameter λ

We allow courts to pick their own choice of λ. This should be the value by which all machines (especially S, but also $\mathcal{D}$, C, G, E, and R) scale their time and query bounds. As a recommendation, we set our λ as a measure of the amount of evidence for the remainder of this paper. That is, λ is a sum of the amount of bits of N queried by E, plus the size of the circuits it uses to test R.

Consider the following example: $R.key()$ contains the respondent’s 128-bit key. $R.message()$ reveals a 1000-bit message, and $N[1]$ stores a 1000-bit ciphertext which is the encryption of $R.message()$ under $R.key()$ using some encryption scheme. First, observe that if λ is set based on the amount of evidence as we suggest, then λ is at least as big as the security parameter of the encryption scheme (since the key is at least as big as the security parameter of the scheme). Second, note that until we approach a 2^{128}-bit message size, λ will not artificially allow brute forcing of the scheme.

We expect that this setting will work for the vast majority of cases. However, there are three scenarios which prevent us from declaring this to always be the case; for these reasons we allow courts to set their own settings of λ.

First, if there is only a small amount of evidence, we may encounter pathologies in which the runtimes of other machines (especially R and S) are artificially short. Thus we additionally set a minimum for λ, say $\lambda \geq 80$, to ensure that all parties still get a reasonable amount of computing power even with minimal evidence. (80 is a typical medium-low setting of a security parameter for a symmetric encryption scheme.)

Second, in some cases an extremely high amount of evidence may be a valid model. For example, a security researcher may wish to model a nation-state adversary as having $\lambda = 2^{128}$ pieces of evidence, a number so high that $t_S(\lambda) = t_S(2^{128})$ would allow time for brute-force decrypting 128-bit cryptosystems. We believe this is unrealistic, but we wish to allow this alternate model as an option. Note that with this setting of λ, since Nature is exponentially long in λ, it will be doubly exponential in whatever parameter λ itself is exponential in. This has the effect that cryptography can be brute-forced, but arbitrary elements of Nature still cannot be.

The third reason we allow alternate choices of λ is that setting this parameter as "the amount of evidence" encourages the government to present an inflated E machine with many useless queries, since this will allow it to use a more powerful simulator. This is another way in which our definition is brittle against maliciously-crafted evidence. All parties should be in agreement on the Evidence machine E before getting to the point of running this definition.

As a final note, the setting of λ does not matter as much in the concrete setting, where the time and query bounds of each machine can be set individually for a single arbitrary value of λ.

Choosing a simulator bound

Arguably the most important choice of parameter in this system is the choice of the time and query bound on the simulator (t_S and q_S respectively). The general effect of setting t_S and q_S is to quantify the work the government must do *without* the contents of the respondent's mind in order to replicate an action that the respondent can do *with* the contents of their mind.

To use a toy example, in order to determine whether the respondent's glove fits her hand, the government likely only must query two locations in Nature (the glove and the hand), and perform the computational work of checking whether they are the same size. Perhaps if the respondent is particularly coy about putting on the glove, the government at worst has to do some extra work to make a plaster cast of her hand and fit the glove to that. All this is nontrivial work, but it seems reaonably attainable within the budget of a police department – a simulator should be fairly easy to build for this action. If the evidence contains information about 1000 hands, each of which is paired to a potential glove (λ is at least 2000: the sum of the 1000 hands in N and the 1000 gloves in N) then this may take roughly 1000x more work than checking one glove – both q_S and t_S are linear functions in λ (assuming λ is chosen as the amount of evidence as described in the previous section). If it is not known which glove corresponds to which hand, then we may have a matching problem between the hands and the gloves: λ and q_S are unchanged at 2000, but t_S is now about 1000^2: each of 1000 gloves must be matched against 1000 hands, a quadratic function in λ.

Increase the complexity of this example a little more and it is not clear that the government could plausibly achieve the cost in question. A poly-time runtime and query bound plausibly represents a stronger government than the real one. In many cases a court might want to pick a more fine-grained option.

Furthermore, observe that all of this does not even make a scratch in the problem of the government needing to break or brute force cryptography, generally a problem that requires *exponential* work in the cryptosystem's security parameter. (Note that as discussed in the choice of λ this parameter should always be at least the size of the security parameter of a cryptosystem used where a key is stored in either R's mind or N.)

An unbounded simulator may be a good choice in an extreme setting where (for example) a very paranoid security researcher wishes to create a scheme that is FC-resilient even if the government is exceedingly powerful and has no bound on its simulator. However, we expect most courts (and indeed most security researchers) to be satisfied with the poly-time and poly-query bounded simulator, which as we said is still giving the government more power than it likely actually possesses.

Choosing a distinguisher bound and advantage

The distinguisher bound may again be either concrete, asymptotic (fine-grained or just "polynomial") or unbounded. For the main section of this paper we focus on computational indistinguishability, that is, the distinguisher time bound $t_{\mathcal{D}}(1^n)$ is polynomial.

The reasoning here is similar to the discussion of the simulator time bound. The choice of computational indistinguishability is a concession to the fact that the government is not all-powerful, as we discussed in 2.3.3. It is hard to see how one would operationalize an unbounded distinguisher, since in real life no party would be able to serve as the distinguisher. A more limited distinguisher is also possible, but we find that the standard poly-time bound is useful for analyzing cryptosystems. We see the choice of an asymptotic polynomial time and query bound as the most useful setting since it enables easier analysis while still being fairly realistic.

We also primarily choose to set the maximum allowed advantage of the distinguisher, $\epsilon(\lambda)$, to a (positive) negligible function, This means that for any polynomial

$p(\lambda)$, there exists some number n such that $\forall\lambda > n$, $\epsilon(\lambda) < \frac{1}{p(\lambda)}$. One could choose to set ϵ to a constant function instead of a negligible one, e.g. the distinguisher must be able to discern the simulated from the real version with at most a 1% advantage to be a foregone conclusion. This would favor the respondent; many more things would be foregone than with a negligible advantage. However, this does not scale with the security parameter, and generally makes analysis more difficult. We see the negligible setting of ϵ as the natural choice.

When to pick asymptotic versus concrete bounds

As described in the previous section, for the most part courts may wish to set a concrete bound for each machine based on the realistic resources that could be applied to the case.

On the other hand, asymptotic time and query bounds make analysing the composition of several actions easier without making the simulator too unreasonably powerful (though still more powerful than it would likely be in real life). As we will discuss further in §2.3.5, we make the normative claim that foregone compositions should compose: one should not be able to "split up" one large non-foregone action into multiple actions which are each individually foregone, and one should not be able to combine non-foregone actions into one larger foregone action. We show in §2.3.5 that this holds in the asymptotic setting. In the concrete setting, one can still consider composition, but one must take careful account of the *total* time and queries used so far, rather than considering each action individually. In our opinion, the asymptotic definition is easier to analyze for composition and it does not lose much, and so we will use it for the remainder of the paper.

One other side effect of choosing a concrete or a fine-grained bound rather than the recommended poly-time bound is that compelling *fine-grained cryptography* may become not foregone. Fine-grained cryptography aims to provide security only up to a specific polynomial runtime adversary (e.g. quadratic), rather than *all* polynomial-time adversaries, e.g. [106, 115, 207, 223]. Compelling decryption with a fine-grained cryptosystem would likely be foregone under the standard poly-time simulator, but not under a more limited simulator.

2.3.5 Sequential composition

In this section, we prove that Definition 2.3.2.2 remains secure under sequential composition. Essentially, our theorem states that the information disclosed by a government compelled action cannot immediately open up *new* actions that the government can subsequently compel.

Theorem 2.3.5.1 (Sequential composition)**.** *Suppose C_1, G_1 is a foregone conclusion with respect to E and S_1. Then C_2, G_2 is a foregone conclusion with respect to E and S_2 if and only if there exists a simulator $S_{1\|2}$ such that $(C_1\|C_2), (G_1\|G_2)$ is a foregone conclusion with respect to E and $S_{1\|2}$.*

While composition generally follows naturally in simulation-based definitions, the proof in our setting is somewhat non-standard. For instance, proving composition for zero-knowledge proofs requires an auxiliary input so that later instances store the results of simulated versions of earlier instances, but our definition doesn't have a direct concept of auxiliary input. We proceed in the other direction: we can proactively store simulated versions of later instances in nature in order to test the limits of whether earlier instances are truly foregone conclusions.

We prove the two directions of Theorem 2.3.5.1 separately. Beforehand though, we find it useful to make a simple observation: the transcript of a composed machine is equivalent to the composition of the transcripts of the individual machines.

Lemma 2.3.5.2 (Concatenation of composed transcripts)**.** *If M_{1a} and M_{1b} are paired interactive Turing machines with identities a and b respectively, and $M_{2a'}$ and $M_{2b'}$*

are paired interactive Turing machines with identities a' and b', then

$$t(M_{1a}, M_{1b})\|t(M_{2a'}, M_{2b'})\square = t(M_{1a}\|M_{2a'}, M_{1b}\|M_{2b'}).$$

Proof. This is a straightforward consequence of the sequential (i.e., non-parallelized) nature of a Turing machine and the way we defined Turing machine composition in §2.3.2. □

The first direction of the theorem states that compelling two foregone conclusions in a row is still a foregone conclusion.

Lemma 2.3.5.3. *If C_1, G_1 is a foregone conclusion with respect to E, S_1, and C_2, G_2 is a foregone conclusion with respect to E, S_2, then $C_1\|C_2$ is a foregone conclusion with respect to $E, S_{1\|2}$, where $S_{1\|2}$ first runs S_1 and then S_2.*

Proof. Since C_1, G_1 is a foregone conclusion with respect to S_1, E, we know that $t(G_1^N, C_1^{N,R}) \approx_c S_1^N\ \forall R, \forall \mathcal{D}^N$ even with oracle access to N. This must include R that sent $\mathsf{NEnc}(t(G_2^N, C_2^{N,R}))$. ($R$ can generate this value by running the code of G_2 and C_2, reading from N and responding from its own code when appropriate.) Thus, $t(G_1^N, C_1^{N,R})\|t(G_2^N, C_2^{N,R}) \approx_c S_1^N\|t(G_2^N, C_2^{N,R})\forall R, \forall \mathcal{D}^N$. By Lemma 2.3.5.2 we therefore also have

$$t((G_1\|G_2)^N, (C_1\|C_2)^{N,R}) \approx_c S_1^N\|t(G_2^N, C_2^{N,R}) \tag{2.1}$$

$\forall R, \forall \mathcal{D}^N$.

Separately, we know that since C_2, G_2 is a foregone conclusion with respect to S_2, E we know that $t(G_2^N, C_2^{N,R}) \approx_c S_2^N\ \forall R, \forall \mathcal{D}$ even with oracle access to N. Since $\mathcal{D}$ can run the code of S_1, it must also be true that

$$S_1^N\|t(G_2^N, C_2^{N,R}) \approx_c S_{1\|2}^N \tag{2.2}$$

$\forall R, \forall \mathcal{D}$.

Thus, combining equations 2.1 and 2.2, we have

$$t((G_1\|G_2)^N, (C_1\|C_2)^{N,R}) \approx_c S_1^N\|t(G_2^N, C_2^{N,R}) \approx_c S_{1\|2}^N$$

Thus, $t((G_1\|G_2)^N, (C_1\|C_2)^{N,R}) \approx_c S_{1\|2}^N$, fulfilling the simulatability property of a foregone conclusion. In order to show that the composed version is a foregone conclusion, we must also show efficiency and satisfiability of evidence.

Clearly, if all of the component machines are efficient (guaranteed by the efficiency of machines in foregone conclusions), their concatenation must also be efficient. The evidence is the same for the composed and non-composed versions, so it maintains the satisfiability and non-censorship properties. This completes the argument. □

The second direction of the theorem states that if the composed request is a foregone conclusion, then each individual request must also be a foregone conclusion.

Lemma 2.3.5.4. *Suppose C_1, G_1 is a foregone conclusion with respect to E, S_1. Suppose also $C_1 \| C_2$, $G_1 \| G_2$ is a foregone conclusion with regard to $E, S_{1\|2}$, where $S_{1\|2}$ is S_1 concatenated with some S_2. Then C_2, G_2 is a foregone conclusion relative to E, S_2.*

Proof. Suppose by way of contradiction $\exists R, \mathcal{D}$ such that $\mathcal{D}$ is capable of distinguishing $t(G_2^N, C_2^{N,R})$ from S_2^N.

Then we can easily construct $\mathcal{D}'$ that can distinguish $t((G_1\|G_2)^N, (C_1\|C_2)^{N,R})$ from $S_{1\|2}^N$ for the same R. $\mathcal{D}'$ simply removes the first set of messages (before the ■ symbol of the interaction between G_1 and C_1) as well as the final □ symbol ending the composed interaction. The result is simply either $t(G_2^N, C_2^{N,R})$ or S_2^N. $\mathcal{D}'$ then calls $\mathcal{D}$ on this input and returns the same result. It is easy to see that $\mathcal{D}'$ can distinguish the composed computation with the same advantage as $\mathcal{D}$ can distinguish only the second interaction. (Notice also that by renaming the arguments, this proof also applies to the first compelled argument.) Thus, we have a contradiction; such a distinguisher cannot exist.

The other properties of a foregone conclusion also continue to hold: If the composed machines are efficient, then the decomposed machines must also be efficient. Furthermore, the same evidence is used in all the iterations, so the satisfiability and non-censorship properties of the evidence remain unaltered.

This completes the proof. If $C_1 \| C_2$, $G_1 \| G_2$ is a foregone conclusion with regard to $E, S_{1\|2}$, then it must also be true that C_2, G_2 is a foregone conclusion relative to E, S_2. □

Combining Lemmas 2.3.5.3 and 2.3.5.4 proves Theorem 2.3.5.1 and demonstrates that the government gains no advantage by waiting for the result of one foregone conclusion request before beginning the next.

This theorem demonstrates that our foregone conclusion doctrine satisfies two intuitively-appealing goals. First, if a compelled action C would *not* be a foregone conclusion given the government's existing evidence, then it should not be possible to split C into smaller actions (compelled in sequence) that collectively perform C and that are each individually deemed foregone conclusions. Second, there should not be a way for the government to compel beforehand a different foregone conclusion C' in order to change the status of C into a foregone conclusion.

We emphasize that the composition theorem only applies to government requests made in sequence without changes to Nature or the Evidence in between; it *is* possible that the government could compel an action, use the response to guide its police investigation to gather more evidence, and then compel a second action based on this additional evidence.

2.3.6 Additional comments on the Foregone Conclusion Definition

This section describes some additional detail to the foregone conclusion definition.

Universal simulator and constructive definition

Recall that our definition (informally) states that the interaction between C and G is a foregone conclusion with respect to S if $\forall R$, the exchange between G^N and $C^{N,R}$ is indistinguishable from S^N.

First, notice that our definition uses a universal simulator: it is defined first, before the other party (in this case, before the respondent's equivocation strategy). This is similar to black box zero knowledge, where the simulator is defined before the cheating verifier. The reason we require a universal simulator can be seen by imagining the opposite: If a different simulator was required to respond to different equivocation strategies, then this amounts to the respondent "testifying" as to which world state (Nature and the Respondent's mind) is true. Instead, for an action to be a foregone conclusion, we insist that the government be able to simulate the action for all equivocations that are consistent with the evidence.

Furthermore, this definition is constructive – it forces the government to submit a construction of S as an input, rather than relying on the mere existence of such a simulator. This puts the burden of proof on the government to present a simulator that demonstrates that something is a foregone conclusion. This is consistent with the legal literature in that the burden of proof is on the government [193]. Although the caselaw has not definitely determined how high the government's burden of proof is [193], we argue in §2.3.3 that our high-but-achievable standard of simulation is appropriate.

The simulator represents a process, not the contents

The foregone conclusion literature is clear that only the *act of production* of most subpoenas may be protected as testimonial under the 5th Amendment, not the contents of the documents themselves [123, l. 410]. (This is often because the contents of the documents were not themselves "compelled," but the act of production of those documents is compelled.) Our model is in keeping with this principle – although the items returned by the interaction or simulator do correspond to "contents" (often elements of N), the object that the government must present in order to demonstrate a foregone conclusion is S: a simulator for the *process* of getting those contents. Our abstract encoding of "Nature" is abstract specifically to allow the government to demonstrate its ability to simulate the *process* of retrieving documents from it. If the government can simulate this process without access to the contents of the respondent's mind (R), then that act of production was a foregone conclusion.

Weak to malicious evidence

One weakness of our definition is that it is brittle to a maliciously-chosen E. This can occur in multiple ways – the most obvious is for the evidence to be chosen such that there is only one computable result for either S or the real interaction, but it could also restrict R's computation time to 1, preventing it from doing its own computation.

However, we point out that in the legal system, the standards for existing evidence are high. In particular, the "due process of law" guaranteed to all persons by the Fourteenth Amendment [301] demands that State prosecutors may not knowingly present false evidence or perjured testimony [228], or allow evidence or testimony they know to be false to stand [7]. Thus, we expect that all parties in a court will agree on the truthfulness of existing evidence before reaching the question of whether a compelled action is a foregone conclusion; the main issue we are trying to prevent is that the compelled action may overreach in a "fishing expedition" (as it did in, for example, *U.S. v. Hubbell* [295, l. 32]).

Respondent's previously-rolled randomness cannot necessarily be compelled

A subtlety arises when compelling the result of random coins that the respondent rolled in the past (e.g. an encryption key). Generally, this type of action is *not* a foregone conclusion without additional information. However, the government *could* compel a respondent to make a fresh sample from the same original distribution (i.e. randomly pick a new encryption key).

The following example is illustrative: Suppose the compelled action is to encrypt a fixed message using the same key the respondent uses to encrypt her hard drive (or any key that was randomly rolled in the past, but is now fixed and known only to the respondent at the time of the compelled action) and return the ciphertext. This is not deterministic since presumably the encryption scheme is randomized. Yet as we shall see, there exists a distinguisher which will always distinguish the real compelled action from the simulated compelled action with high probability. Because the respondent essentially has the key hardcoded, there exists a distinguisher that knows the same key. (Equivalently, the respondent can communicate the key to the distinguisher via the covert channel described in §2.3.6.) That distinguisher can always successfully decrypt the real ciphertext to recover the fixed message. On the other hand, the simulator attempting to replicate this ciphertext does not know the key in the respondent's mind, and so has a minuscule chance of providing a ciphertext that successfully decrypts under the same key to that message. This provides a reliable means of distinguishing.

On the other hand, compelling the respondent to create a fresh key and then encrypt the message under that key is a foregone conclusion – the respondent does not execute the compelled action, and therefore does not roll the key, until after it has already set Nature, and there no longer exists a single distinguisher that can reliably reproduce the freshly-random key. So compelled encryption under a fresh key is compellable.

As a side effect, it may be the case that the "true" way in which the respondent acted in the past no longer corresponds to an α-allowed R. For example, if the distribution of R's symmetric key is known, but so is the ciphertext and nonce for an unknown plaintext, this effectively "collapses" R's randomness by conditioning on the result, and forces R to sample twice in a way that would lead to the same result.

In effect, we consider this a reasonable way to describe the choices R may have made in the past, but is now committed to. Perhaps R could have generated any ciphertext, but in the end, we know exactly which ciphertext it generated, and so any future interaction with R must take this into account.

If for some reason a random distribution must be described in R rather than in the compelled action C, this is also possible under this framework. Since the evidence has the power to inspect R's source code, it has the ability to verify that R chooses a random variable correctly according to a known distribution, using randomness from its randomness tape.

These distinctions are also relevant to the compelled encryption scenario described formally in Theorem 2.5.5.1.

Covert channel between R and $\mathcal{D}$

The adversarial respondent will equivocate and try to provide the distinguisher with information that will allow it to distinguish the real execution from the simulated execution. It can do this by setting a value in N to something that will help the distinguisher realize which is correct. We define a useful functionality:

Definition 2.3.6.1 (Encoding within nature (NEnc_ℓ, NDec_ℓ)). (NEnc, NDec) is an *encoding scheme within nature* for R and $\mathcal{D}$ over messages of length capped by ℓ defined by the syntax:

- $\Delta' \leftarrow R.\mathsf{NEnc}(z)$, such that $\Delta' \subseteq R.\texttt{Equivocate}$ and $z \in \Sigma^\ell$
- z' = NDec^N, making queries only to indices in Δ'

where $z' = z \forall z \in \Sigma^\ell$.

This functionality represents the ability of R to hide information in N that the distinguisher may use. There are many possible instantiations of this functionality. As an example for messages z of length bounded by ℓ, consider R, $\mathcal{D}$ with a hardcoded index i and mask $m \in \Sigma^\ell$:

$$\begin{aligned}
\mathsf{NEnc}_{i,m,\ell}(z) &= \{(j, z[j] \oplus m[j])\}_{j=i..(i+\ell)} \\
\mathsf{NDec}^N_{i,m,\ell} &= N[i] \oplus m[0] \| N[i+1] \\
&\qquad \oplus m[1] \| \cdots \| N[i+(\ell-1)] \oplus m[\ell-1]
\end{aligned}$$

This works because the "$\forall \mathcal{D}$" is defined after the "$\forall R$." This means that some R will have such a code, and some $\mathcal{D}$ for that R will know of it. This also depends crucially on the non-censorship property of the evidence – if E had the ability to prohibit R from modifying cells that it did not inspect, this functionality would not work.

This functionality is most relevant in the case of samples from a known distribution. Suppose R has a secret s sampled from a known distribution $\mathcal{S}$ that is specified in the evidence. The simulator will be able to sample a fresh s' from the same distribution $\mathcal{S}$. If C compels R to reveal the *specific* s it sampled, however, R can use $\mathsf{NEnc}(s)$ and $\mathcal{D}$ may run NDec^N to recover s, allowing it to distinguish the real output of the interaction between C and R (which will always return s) from the simulator (which will return a fresh sample from $\mathcal{S}$).

However, notice that no randomized action during the actual C can be recorded in N in this way, only actions that depend on the randomness of R alone.

As a consequence, the only way for S to simulate a deterministic compelled request is for the output of S to exactly equal the transcript of C's interaction with G.

2.4 Legal analysis

In this section, we justify Def. 2.3.2.2 by demonstrating its consistency with prior court cases that involve the foregone conclusion doctrine. We begin by comparing our definition with existing legal scholarship in §2.4.1. Then, we apply our definition to all relevant U.S. Supreme Court cases in §2.4.2 and all circuit court cases in §2.4.3 that were identified by the legal scholarship, and we demonstrate how our definition reaches the same conclusions as the courts. We discuss encryption-related cases [88–91, 176, 264, 266, 273, 274, 296, 298, 303, 304, 306] in §2.4.4 however we do not use them in our analysis of our definition. We believe these cases are actually less illustrative than their non-encryption counterparts because these rulings are quite varied and subject to being overturned by higher courts. We end the section with a discussion of prior and potential future Supreme Court cases involving compelled decryption and the foregone conclusion doctrine.

2.4.1 Legal scholarship context

This section provides a thorough description about how our approach compares to prior legal scholarship on the foregone conclusion doctrine.

Other legal analyses of compelled decryption [85, 193, 198, 222, 256, 282, 317] rely upon analogies between encryption and physical security mechanisms like safes or shredders. Kerr recently stated "whether [the Fifth Amendment] privilege bars compelled entry of the password. . . depends on a choice of analogy" [195]. These analogies are further muddled by ambiguous language in court cases: In a now-infamous dissent, Justice Stevens said that he "do[es] not believe [a defendant] can be compelled to reveal the combination to his wall safe – by word or deed" [111]. Does "reveal by deed" mean to be forced to enter the combination without the government seeing it? We assume so, but this is not the only interpretation; for example, Orin Kerr interprets the hypothetical to mean that a person cannot be compelled to *reveal* their combination to the government by opening the safe in plain sight of an investigator [193].

We wrote this paper to move the compelled decryption debate beyond the choice of analogy. We recognize the prevalence and value of analogies in the development of common law, but because their use leads to such differing results in this case, we believe this situation warrants rejecting analogies. Under our model, we can reason directly about the principle that for a compelled action to be a foregone conclusion, it should not "rely on the contents of the mind." This also suggests a change to the three-prong test of existence, location/possession, and authenticity for determining whether an action is a foregone conclusion. Rather than reasoning only about these (which happened to be the implicit testimony in *Fisher*, as we will show in §2.4.2) we can reason in a thought experiment about the government's ability to recreate the act of production without using the contents of the respondent's mind.

Because we can avoid the use of analogies, our reasoning is different than all prior work. The closest legal landmark to our model is Sacharoff's authentication-based interpretation of "reasonable particularity" [256], but there are some important differences between the two approaches. Sacharoff's envisioned test, like our method, is based on the idea that information entered into evidence from non-respondent sources can be used to demonstrate a non-reliance on the contents of the respondent's mind. Indeed, one could argue that the simulator in our scheme must produce "reasonably similar" output to that of the true compelled action. However, the methods are not the same. First, addressing an issue brought up by Kerr [193], our method applies to any compelled action even if there are no produced documents at the end that could be described with "reasonable particularity." Second, and more importantly, our method highlights the fact that the action taken, not the objects produced, contains the implicit testimony. For better or worse, the reasonable particularity method makes it harder to distinguish between the "door-opening" and the "treasure," as Kerr would put it [193]. Our model makes it clear that the government must not learn the new implicit testimony involved in the process of complying with the request (as opposed to the results).

Our interpretation is very different from other prior work. As mentioned, Kerr [193] distinguishes between "door-opening" and "treasure." This analogy, reasonably, tries to separate the act of production from the contents produced. In the same paper, Kerr proceeds to claim that "'I know the password' is the only assertion implicit in unlocking the device" [193, p. 779] We disagree; we described the "reliance" on the respondent's mind in §2.5.5. Our objection is solely in the compelled action, not the contents revealed.

Kiok [198] bemoans the fact that the cryptography analogies have, thus far, "missed the metaphor." McGregor [222] also notes that the choice of analogy greatly impacts the outcome, and proposes the analogy of piecing together shredded papers without knowing which order they go in. This analogy is an improvement over the safe/combination dichotomy, but we believe our approach avoids the issue entirely.

In his discussion on foregone-conclusion-based compelled decryption, Terzian [283] describes a split between courts that compel decryption of an entire device and decryption of specific files, and places the burden of proof on those who argue for specific files. Our analysis does not fit neatly into either of these categories, but it is closer to the files interpretation. We do not require the government to specify "every scrap of paper" that must be produced, but we do require the government to avoid compelling files for which the contents of the mind are demonstrably necessary to access (since they did not demonstrate an alternative method of production).

Finally, our conclusion does not go as far as Winkler [317], who claims that the foregone conclusion doctrine does not apply to non-physical evidence and thus compelled decryption is never a foregone conclusion.

2.4.2 U.S. Supreme Court cases

This section contains our analysis of all federal Supreme Court cases involving the foregone conclusion doctrine. In §2.4.3, we also analyze several key foregone conclusion Circuit Court cases.

The foregone conclusion doctrine dates back to *Fisher v. United States* [123]. We checked all citations of *Fisher* in Google Scholar's database of case law and found only two subsequent Supreme Court cases that deal with the foregone conclusion doctrine: *United States v. Doe (1984)* [294] and *United States v. Hubbell* [295]. In this section, we show that our definition agrees with the result of all three cases.

Evidence in *Fisher* [123]	**Translation into our framework**
The papers...	$\exists k, \mathsf{p}$ such that:
are in the possession of the taxpayer [123, line 409]	$k \in \mathsf{locations}$ (where $\mathsf{locations}$ is a small set of indices in Δ)
were prepared by the accountant [123, line 411]	implies that $\exists (k, \mathsf{p}) \in N$
are the kind usually prepared [in this situation] [123, line 411]	Δ contains code in a small known set of indices acc that creates p
can be authenticated by the accountant [123, note 13]	$\exists \mathsf{Auth} : \mathsf{Auth}^{\mathsf{acc}}(x) = 1$ iff $x = \mathsf{p}$

Table 2.1: *The evidence check E in* Fisher v. U.S.

Fisher v. U.S. [123]. The *Fisher* case examined a hypothetical in which a taxpayer R was compelled to produce an accountant's papers in R's possession (similar to the motivating example in §2.2). As a note, although hypothetical scenarios described in a court opinion typically do not contribute to the ruling, in the case of *Fisher* the entire foregone conclusion doctrine has arisen from the basis of this hypothetical.

The court in *Fisher* determined that the act of producing the papers communicates potentially testimonial and incriminating evidence to the government; "[c]ompliance with the subpoena tacitly concedes the existence of the papers demanded and their possession or control by the taxpayer. It would also indicate the taxpayer's belief that the papers are those described in the subpoena."

Recall that *Fisher* determined a three-prong test for whether or not an act of production was a foregone conclusion: the government must have knowledge of the papers' existence, location, and authenticity. Table 2.1 translates the circumstances of *Fisher* into an evidence test within our framework. The evidence includes the facts that the government knows that the papers p exist, they reside in one of a small set of possible $\mathsf{locations}$, the taxpayer R can produce them, and the papers can be authenticated using only the accountant's testimony (without the taxpayer's help).

With the evidence described above, the compelled action is simulatable using only information within nature: S can search through $\mathsf{locations}$ and use the accountant to test which papers are the desired ones. This simulation is perfect no matter how the taxpayer R equivocates, as long as R puts the papers $\mathsf{p} \in \mathsf{locations}$ as required by the evidence check E. Moreover, E does not censor R, it allows the true taxpayer code, and all procedures are efficient. Thus, the taxpayer R must produce the legitimate papers p. This analysis matches the Supreme Court ruling that compelling the papers is a foregone conclusion.

Note that in the Supreme Court's analysis, the fact that the papers "are the kind usually prepared by an accountant working on the tax returns of his client" [123, line 411] was used as evidence toward the papers' existence and their possession. However, later, in *U.S. v. Hubbell*, the Supreme Court rejected a similar argument, saying that the government cannot rely on the "overbroad argument that a businessman such as respondent will always possess general business and tax records that fall within the broad categories described in this subpoena" [295, line 45]. We have chosen to take the evidence presented by the court in each individual case; our framework remains the same whether such a statement is put into evidence (and therefore part of the government's existing knowledge) or not.

We emphasize that all facts contained within the evidence E in Table 2.1 are necessary for the simulator to succeed. The remaining two cases show how the foregone conclusion decision changes when the government cannot pin down the location of, or independently authenticate, the papers.

U.S. v. Doe [294] (1984). The *Doe* case also required the respondent R to produce documents, but unlike in *Fisher*, in this case the government did not have much prior information about the documents. As a consequence, the non-censorship requirement states that E cannot restrict where the respondent R places the documents in nature, or indeed whether she writes the documents anywhere at all. The wide variety of possible respondent equivocations defeats the simulator S from above (and indeed any other simulator), so Definition 2.3.2.2 is not satisfied. Our definition again agrees with the result of the case, in which the Court found that "nothing in the record that would indicate that the United States knows ... that each of the myriad documents demanded by the five subpoenas in fact is in the appellee's possession or subject to his control" [294, note 12] and thus the act of production is not a foregone conclusion.

U.S. v. Hubbell [295]. The *Hubbell* case is complicated by a grant of immunity that is outside of our model; we describe here a subset of the facts that remain relevant in our setting. The government compelled Hubbell to provide "documents fitting within ... 11 broadly worded subpoena categories." [295, line 42] In this case, the government not only sought the documents themselves, but also the "respondent's assistance ... to identify potential sources of information" [295, line 41] and to "testif[y] that those were all of the documents in his custody or control that were responsive to the commands in the subpoena" [295, line 31]. Our definition is unsatisfiable for compelled actions that are subject to either one of these considerations: given any simulator S, we can construct an equivocating respondent R that decides differently from S which documents are relevant. Once again, our definition aligns with the Supreme Court's decision that it was "unquestionably necessary for respondent to make extensive use of 'the contents of his own mind' in identifying the hundreds of documents responsive to the requests of the subpoena" [295, line 43].

An additional case: Doe v. United States (1988) [111]. A fourth case, *Doe v. United States (1988)* [111] is often brought up in discussions of the Fifth Amendment as relevant to compelled decryption. However, the decision in that case found that the compelled action was not testimonial at all, and thus the justices never reached the point of asking whether it was a foregone conclusion. The case itself concerned forcing

Doe to sign 12 consent forms authorizing various foreign banks to “disclos[e] any bank records” Doe had at the banks under specific account numbers [111, line 203]. The Supreme Court found that the consent directive Doe was forced to sign had been “carefully drafted not to make reference to a specific account, but only to speak in the hypothetical” and thus “neither the form, nor its execution, communicates any factual assertions, implicit or explicit, or conveys any information to the Government” [111, line 215]. I agree with the court on its specific ruling: since the papers had been written so as not to state any claims, the compelled action was not testimonial and therefore not privileged against self-incrimination. However, I find the outcome of this case distasteful for a different reason. It seems natural to me that the government should not be able to claim to external entities that you consent to an action that you do not, in fact, consent to. So although I agree that the statement was not incriminating, I believe there ought to be a law prohibiting the government form compelling you to sign a consent form against your will.

2.4.3 Circuit Court cases

In addition to Supreme Court foregone conclusion cases, there are also several important foregone conclusion cases in the circuit courts. Rather than check all approximately 1200 circuit court cases citing Fisher, we rely on prior law review articles [96,181,193,198,247,282,317,318] to identify the most relevant circuit court cases. Five circuit court cases are mentioned or cited that involve the foregone conclusion doctrine but do not involve encryption. For a brief summary of the encryption cases, see §2.4.4.

U.S. v. Greenfield (2nd circuit) [305]. *Greenfield*, a 2016 case from the 2nd Circuit Court of Appeals, is a good example of a case that came close to being a foregone conclusion, but needed slightly stronger evidence to prevent the recipient

Greenfield from equivocating. Greenfield had been accused of tax evasion and, after some back-and-forth, had been compelled to produce three categories of bank records for accounts already known to the government, some non-bank documents including “ownership records”, “professional services documents”, and “communication documents”, plus his expired passport and other documentation for trips in his passport [305, line 114]. The court found that, had the government requested the documents in 2001, production of the passport and travel documents would have been a foregone conclusion, authenticating by using the Department of State or airlines. However, while the government showed knowledge of the existence and control of the other documents, it would have relied on Greenfield’s testimony to authenticate them and therefore they were not compellable. However, in 2013, when the summons occurred, even the passport and travel documents were no longer a foregone conclusion, because the government could not show that Greenfield had retained control over them through the 12-year gap. For our purposes, the 2001 analysis is more interesting than the 2013 analysis, since the categories of documents compelled were very similar, but came to different outcomes nonetheless.

Let br or, psd, cd and td be unknown strings, and exists Auth such that $\mathsf{Auth}_{\mathsf{td}}(x) = 1$ if $x = \mathsf{td}$. Let $\mathsf{possess}$ be a small set of indices. The evidence established the information shown in Table 2.2.

We first examine what would have occurred in *Greenfield* if the production had been compelled in 2001. The 2nd Circuit determined that compelling the travel documents td was a foregone conclusion, since the existence, control, and authenticity of the documents were established. We can see that a S that reads each location in $\mathsf{possess}$ and checks its authenticity with $\mathsf{Auth}_{\mathsf{td}}(\cdot)$ will always return the same td as is returned by $R.\mathtt{M'}$.

The existence of the communication documents cd was not proven, thus, R can

Knowledge of government	**Formalization in E**
In 2001, the travel documents...	$\exists k, \mathsf{td}$ such that:
existed [305, line 123]	$(k, \mathsf{td}) \in \Delta$
were in Greenfield's possession [305, line 123]	$k \in \mathsf{possess}$ ($\mathsf{possess}$ is a small set of indices)
were under Greenfield's control [305, line 123]	$\exists\mathtt{M} : R.\mathtt{M} = \mathsf{td}$
could be authenticated [305, line 123]	$\exists\mathsf{Auth} : \mathsf{Auth}^N(x) = 1 \text{ iff } x = \mathsf{td}$
In 2001, the communication documents...	
would be in Greenfield's possession [305, line 123]	$(\exists i : N[i] = \mathsf{cd}) \to i \in \mathsf{possess}$
In 2001, the ownership records...	
existed [305, line 122]	$(j, \mathsf{or}) \in \Delta$
In 2001, the bank records...	
existed [305, line 119]	$(i, \mathsf{br}) \in \Delta$
were in Greenfield's possession [305, line 119]	$i \in \mathsf{possess}$
were under Greenfield's control [305, line 120]	$\exists\mathtt{M'} : R.\mathtt{M'} = \mathsf{br}$

Table 2.2: *Evidence shown in* Greenfield

choose not to place them in N at all, making the chance of their recovery by S^N exponentially small. Similarly, the ownership records existed somewhere in N, but since no knowledge is known about their whereabouts or control, S^N's chance of recovering them is still exponentially small.

However, the bank records br could not be authenticated. Supposing the government has some idea of what the documents should look like (i.e. their distribution, which we assume is not overwhelmingly in favor of one outcome; else authentication would be trivial), S has the power to search $\mathsf{possess}$ (a small subset of N) and return the document there most likely to be the correct record. However, R has the ability to forge an alternate br' (e.g. by resampling) and *also* put that in $\mathsf{possess}$ in addition to the true br'. In this case, C (calling $R.\mathtt{M}$ and returning the result, without interacting with G) would return the true br consistently, but S would mistake the false br' for the real br a non-negligible amount of the time. This allows $\mathcal{D}$ which has the correct br hardcoded into it to distinguish between the two possible distributions.

As an added note, the non-foregone decision for the bank records seems to rest on the inability of the government to authenticate them, and in fact some other court cases seem to be more lenient in their standards for authentication. If such an authentication mechanism were known, the bank records would have been a foregone conclusion by the same rationale as the travel documents.

In re Grand Jury Proceedings (8th circuit) [172]. In this 8th circuit case, the Bayirds were served with a subpoena for several categories of documents, primarily business documents related to their income. The circuit court found that the subpoena was insufficiently well-defined – the Bayirds' choice of which documents were covered and which were not could be testimonial [172, line 381]. In short, R could return different distributions of documents while still complying with the evidence. As long as it returns a different distribution than S (which must work for all choices of R), the results will be distinguishable and therefore not a foregone conclusion.

In re Grand Jury Subpoena (9th circuit) [173]. In this case, respondent John Doe was an employee of a corporation accused of price fixing DRAM chips. Two subpoenas were issued: the first to John Doe, compelling the production of all documents he had related to DRAM sales, including calendars, diaries, and notes; the second to the corporation [173, line 911]. The government had reasonably extensive knowledge of Doe's actions through interviews and other documents. Nonetheless, the subpoena served to Doe was overbroad and asked for categories of documents that the government did not know existed until the results of the *second* subpoena (to the corporation) was responded to. Furthermore, the government never provided a means to authenticate the personal documents (e.g. diaries) without Doe's testimony. S would have no means of simulating these categories of documents, and as such, the result is not foregone.

U.S. v. Ponds (D.C. circuit) [307]. In *Ponds*, a defense attorney Ponds was accused of tax evasion and fraud with regard to a previous case, and was asked to produce documents in several categories. For our purposes, we will consider only two of these categories: documents referencing a white Mercedes Benz (suspected by the government to have been illegally held by Ponds) and copies of correspondence between the Law Offices of Navron Ponds and courts and prosecutors having to do with the prior case. For the first category, while the government suspected Ponds had the car, the only information in E was that the car was "normally parked at Ponds' apartment and was registered to his sister." [307, line 325] For the second category, since the government was party to the correspondence [307, line 325] and Ponds was expected to have retained a copy, E may require the value in N representing the documents to be the exact string that the government saw before, allowing S to duplicate it exactly. Compelling the first category was not a foregone conclusion; compelling the second category was.

In re grand jury subpoena (2nd circuit) [174]. In this 2nd circuit case, the government subpoenaed John Doe to produce "[t]he original version of any diary or calendar for the year 1988, a copy of which has been produced to the SEC" [174, line 88], and this act of production was determined to be a foregone conclusion. Ultimately, this case is fairly straightforward in our model; the evidence demands that a copy of the documents be in Nature in a set of indices corresponding to the SEC, and the simulator could produce the documents by obtaining them from there.

2.4.4 Compelled decryption cases

As stated at the beginning of this section, we do not fully analyze prior encryption cases under our model. This is because, to our knowledge, only two encryption cases concerning the foregone conclusion doctrine have risen to the level of the circuit courts, and they were decided quite differently. The 11th Circuit found in *In re Grand Jury Subpoena Duces Tecum* [175] that compelled decryption of a hard drive with unknown contents was *not* a foregone conclusion, since the government had not shown that the drives contained any files. That is, they impose a requirement that the government must know what they will find on the encrypted drive with "reasonable particularity" [96, 173, 175, 307]. However, the 3rd Circuit has rejected this requirement [302]. They found in *U.S. v. Apple MacPro Comput.* [302] that decryption of a particular hard drive *was* a foregone conclusion, in part because they had verbal testimony from the defendant's sister as to the contents of the drives.

There are also many cases in lower courts involving encryption and passwords. Most of these courts agree that compelling the disclosure of the password (instead of compelling the defendant to enter the password into the device to unlock it) is not permissible even under the foregone conclusion exception to the fifth amendment [89, 264, 306]. Only one state supreme court found that disclosure of the password itself is allowable, stating that passwords are "of minimal testimonial value" [273]. Several states found that compelling entry of passwords (rather than disclosure) is allowed, but cite different reasons. In Massachusetts, the standard is either that the government must show that the defendant knows the password [91] or that she knows the password/key, knows that the device is encrypted, and has been shown to be the owner of the device and its contents [90]. In North Carolina, a recent case allowed compelling entry of the password, but the defendant had already admitted to using the device to store illegal material [303], leaving the alternative undecided. The U.S. district court for the northern district of California decided that since biometrics are compellable, so too passwords must be compellable [298]. On the other hand, when denying an application for a search warrant, the same court decided a year later that since biometrics often serve the same purpose as passwords, perhaps both biometrics and passwords are not compellable [221]! Finally, Indiana's state Supreme Court ruled that even entry of the password is not compellable unless the government can show that it knows the existence of specific files, and that they belong to the defendant [266]. We refer readers to [234] for additional details on several of these cases.

U.S. Supreme Court cases on compelled decryption Thus far, the federal Supreme Court has had two potential compelled decryption cases come before it, however it has rejected both. In both cases, although the reasons for denying the petition are not given by the Court, we suspect that the reasoning seems to be largely independent from the question at hand. The first case, *Davis v. Pennsylvania* [89], did not represent a true circuit split for its specific question at the time of its petition; as described by Orin Kerr [194], there is therefore no reason for the Supreme Court to get involved.

There is a split now, with *Andrews v. New Jersey* [273]. Orin Kerr describes [196] that *Andrews* was likely rejected by the Supreme Court due to the "final judgement" rule – the final judgement for Andrews has not been decided by the state yet, so the Supreme Court is likely unable to claim jurisdiction.

It is not yet clear which case the Supreme Court will take to resolve the compelled decryption debate, but we still believe it is likely to happen in the next year or two.

2.5 Compellability of cryptographic systems

In this section, we analyze whether it is a foregone conclusion for the government to compel the respondent to use some common cryptographic constructs: one way functions, commitment schemes, encryption schemes, and non-interactive zero-knowledge proofs. We show that compelling the use of these cryptographic primitives is typically *not* a foregone conclusion under our definition, although there exist fact patterns for which it is foregone.

For consistency, throughout this section we presume that the respondent contains a method $R.s$ that, if called, deterministically reveals a secret within the respondent's mind like a password, encryption key, or value inside a commitment. This models the common scenario where the government has evidence that R "knows a secret." Note that we generally place no restrictions on the behavior of $R.s$, i.e., the government does not know anything about the nature of R's secret itself aside from the fact that R knows it.

2.5.1 One way functions

Let $f : \mathcal{X} \to \mathcal{Y}$ be a one-way function. In this section, we show that compelling a preimage of $y \in \mathcal{Y}$ is typically not a foregone conclusion. Specifically, this compelled action is only foregone if the government can demonstrate that R knows exactly one preimage and the government knows an alternative method to produce the same preimage.

Lemma 2.5.1.1. *Let $E^N(R) := \exists R.s \in \mathcal{X} \wedge f(R.s) = y \wedge E'$ be the evidence that the method $R.s$ exists, it produces an element in $\mathcal{X}$ that is a preimage to y, and any additional evidence E' that the government knows. Then, the compelled action $C^{N,R} := R.s$ is a foregone conclusion with respect to evidence E if and only if this evidence suffices for the government to provide a simulator S that reliably produces $R.s$.*

Proof. This compelled action C is deterministic, so the government must simulate it perfectly to evade detection by the distinguisher that has the real $R.s$ hardcoded into it. □

Whether the government can build S depends on the additional evidence E' at its disposal. If $E' = \emptyset$ and $y \leftarrow \mathcal{Y}$ is sampled uniformly, then simulation is impossible by the one-wayness of f. However, there exists evidence that permits government simulation, such as if E shows that the respondent wrote down $R.s$ somewhere in Nature.

This question has immediate relevance to existing court cases – in the most famous example, the 3rd Circuit ruled that a device owner can be compelled to decrypt the contents if the Government can show its knowledge (via hash values) of files on the device, and that the owner is capable of accessing them [302, line 248]. Our definition would arrive at a similar conclusion but via different means. By Lemma 2.5.1.1, compelling the preimage of a hash is not a foregone conclusion on its own. Nevertheless, in the facts of the case [302], digital forensic examiners were able to identify encrypted files with specific hash values that were known to contain child pornography. We believe the Government could have shown that it was able to produce testimony or evidence that would describe the files (preimages) – the forensic examiners could likely fill such a role. This would allow the creation of a simulator that would make requesting the files (preimages) a foregone conclusion. While the court in [302] forced the decryption of the entire device (actually multiple devices), we believe that only the specific files with known preimages should have been compelled. The remaining files could not have been returned without the use of the respondent's mind.

2.5.2 Commitment schemes

Compelling a randomized functionality introduces a new wrinkle beyond the cases discussed in §2.4 and §2.5.1: now the simulator merely needs to be computationally indistinguishable from the real transcript, rather than being identical.

Concretely, we consider below a randomized commitment scheme $(\mathsf{Com}, \mathsf{Decom})$ that is computationally binding and hiding. The algorithm $\mathsf{Com}(s) = (c, r)$ produces a commitment c that is sent to the (government) receiver and a random state r that is maintained by the (respondent) committer, and $\mathsf{Decom}(c, r) = s$ uses both of these values to recover the original secret s. we show below that it is a foregone conclusion for the government to compel the respondent to commit (but not decommit!) to the secret in her mind.

Lemma 2.5.2.1. *A compelled action $C_{comm}^{N,R}$ to sample a hiding commitment $c \leftarrow \mathsf{Com}(R.s)$ is a foregone conclusion, as long as the government has evidence E that the method $R.s$ exists.*

Proof. The government can provide the trivial simulator S_{comm} that chooses a random value x and returns a commitment to it. We claim that this simulator can even fool a distinguisher that has $R.s$ hardcoded into it, because R *cannot* communicate the randomness used within the real commitment since it is only chosen later within C_{comm}. If there exists a distinguisher $\mathcal{D}$ that can distinguish a commitment to $R.s$ from a commitment to a random x without knowing the randomness used (i.e., without opening), then $\mathcal{D}$ breaks the hiding property of Com. □

Similarly, it is also foregone to compel a commitment to a value s that is not within R. This includes the settings in which C samples a secret s at random, hardcodes s, or obtains s from a known location in nature.

On the other hand, compelling the opening of a commitment to a secret value is *not* foregone unless the government already had the ability to compel the secret via other means. This lemma leverages the power of our composition theorem.

Lemma 2.5.2.2. *Let $C_{decom}^{N,R}$ be a machine that decommits to a value c provided by the government G^N. Also, let $E := \exists R.s \wedge E'$ be any evidence that includes the fact that $R.s$ exists, and let S be any simulator.*

Then, $(C_{decom}^{N,R}, G^N)$ is a foregone conclusion with respect to E and S if and only if there exists a simulator S' such that compelling the secret $R.s$ is a foregone conclusion with respect to E and S' independently of the commitment scheme.

Proof. We apply the Theorem 2.3.5.1 with the machines C_{comm} and C_{decom}. Combining the theorem with Lemma 2.5.2.1, there exists some simulator S' such that the composed machine $C := C_{\text{comm}} \| C_{\text{decom}}$ is a foregone conclusion with respect to E and S' if and only if C_{decom} is foregone with respect to E and S. Note that C simply commits to this value, provides the commitment to the government and then receives the same commitment back, and opens the commitment in a binding manner. Hence, C is equivalent to the machine C' that outputs the secret $R.s$, without any commitment scheme involved. Therefore, C_{decom} is foregone with respect to E and S if and only if C' is foregone with respect to E and S', as desired. □

2.5.3 Zero knowledge proofs

Next, we consider an interactive proof protocol Π, where R's secret equals a witness to an NP language. It turns out that compelling a ZK proof is possible but uninteresting. While most of the claims in this section require the government's evidence to contain the fact that R knows a secret with a particular structure, in this case that is already equivalent to the knowledge gained from the ZK proof itself. Sadly, this evidence is required, even for languages in P! The lemma below also applies to ZK arguments and to proofs of knowledge since it is agnostic to the knowledge soundness property.

Lemma 2.5.3.1. *Let (C, G) execute an interactive ZK proof where C acts as the Prover with witness $R.w$, and G acts as the Verifier. Given any evidence E, there exists a simulator S such that (C, G) is a foregone conclusion with respect to E and S if and only if the government's evidence suffices to show that $R.s$ is a witness to the NP statement.*

Proof. If the evidence E allows R to equivocate between a valid and invalid witness for $R.s$, then no simulator can consistently emulate both options. On the other hand, if E guarantees that $R.s$ is a witness, then the compelled action is simulatable by the algorithm S that hardcodes the circuit G and runs an execution between the ZK simulator S_{ZK} and verifier G, potentially rewinding G as usual. The only remaining equivocation available to the respondent is her choice of $R.s$ among satisfying witnesses, but this change is inconsequential by witness indistinguishability. □

Next, we consider non-interactive ZK proofs of knowledge using a common reference string (CRS) as the trusted setup, which is sampled honestly by the respondent and checked by the evidence. In this scenario, the government is in a weaker position than before: in order to compel a NIZK, the government must know a witness themselves.

Lemma 2.5.3.2. *Let C denote a non-interactive ZK proof of knowledge using the witness $R.s$. It accesses a CRS stored in Nature, where the CRS is placed by R and verified by E. If there exists E and S such that C is a foregone conclusion with respect to E and S, then there exists an extractor X that returns a witness.*

Proof. Because the S has no control over the CRS, its proofs are real. If S produces a proof with noticeable probability (over the random sampling of the CRS, among other things), then the knowledge soundness property guarantees the existence of an extractor X' that can extract a witness when executing S multiple times with on different choices of CRS. While the foregone conclusion game in Def. 2.3.2.2 only runs S once (without rewinding), we can construct the desired extractor X by running the entire game many times, since R will honestly sample the CRS independently each time. □

2.5.4 Pseudorandom functions

Next, we examine the circumstances under which the government may compel the use of a pseudorandom function family $\{F_k : \mathcal{X} \to \mathcal{Y}\}_{k \in \mathcal{K}}$. This question turns crucially on whether the key is sampled freshly and ephemerally as part of the compelled action, or if the action requires the use of a long-running key that can be used elsewhere in Nature.

Lemma 2.5.4.1. *Let $C_{prf}^{N,R}$ be the circuit that samples a random key $k \in \mathcal{K}$ and outputs $F_k(R.s)$. This compelled action is a foregone conclusion with respect to any evidence E that includes the fact that the method $R.s$ exists.*

Proof. Just as with Lemma 2.5.2.1, the government can provide the trivial simulator S that chooses a random output $y \in \mathcal{Y}$. Any algorithm $\mathcal{D}$ that can distinguish C_{prf}^{R} from S also serves to break the pseudorandomness of F_k. □

Lemma 2.5.4.2. *Let $\tilde{C}_{prf}^{N,R}$ be the circuit that computes $F_k(x)$, where the key equals the respondent's secret $k = R.s$ and the constant $x \in \mathcal{X}$ is publicly known. Given the minimal evidence $E := \exists R.s$ that R knows the key, there is no simulator S under which $\tilde{C}_{prf}$ is foregone with respect to E and S.*

Proof. This evidence permits R to equivocate between two secrets k and k' that produce different outputs $F_k(x) \neq F_{k'}(x)$, and it must be possible to efficiently sample such keys or else making a query to x would distinguish the PRF from a random function. Any simulator S must fail to output at least one of these strings with noticeable probability, and R can choose this one to evade simulation. □

Lemma 2.5.4.3. *Let $\tilde{C}_{prf}$ be defined as in the previous lemma. Suppose the government knows the value of k as evidence $E^N(R) := (R.s = k)$. Now, there exists a simulator such that $\tilde{C}_{prf}$ is a foregone conclusion.*

Proof. Simulator S^N computes $F_k(m)$ from the known values. This perfectly emulates the real transcript. □

2.5.5 Symmetric encryption

In this section, we consider the compellability of symmetric (authenticated) encryption, which is of particular importance due to its ubiquitous use within full-disk encryption systems. We show that if the respondent keeps the secret key (or a high-entropy password used to derive it) only in her mind, and there are no side channels in Nature capturing the intermediate state during encryption and decryption, then both compelled encryption and decryption are not foregone conclusions.

We focus on the Counter Mode construction of symmetric encryption from a pseudorandom function where KeyGen samples a PRF key, $\mathsf{Enc}(k, m) = (r, F_k(r) \oplus m)$ and $\mathsf{Dec}(k, (r, c)) = F_k(r) \oplus c$. We remark though that the following theorem would also hold for many other modes of operation, including ones that provide authenticity.

Theorem 2.5.5.1. *Suppose the respondent stores two secrets: a secret key k and a message m; that is, $R.s = (k, m)$. With respect to the evidence $E := \exists R.s$ that R knows the secrets,*

- *Compelled encryption of message m under an ephemeral key $k^* \leftarrow \mathcal{K}$ is a foregone conclusion using the simulator that outputs a random element of the ciphertext space.*
- *Compelled encryption of message m or decryption of a ciphertext c using the respondent's secret key k are not a foregone conclusion with any simulator.*

Proof. For the first claim, S can simply sample a random string c' in the ciphertext space. Any algorithm $\mathcal{D}$ that can distinguish $(r, F_{k^*}(r) \oplus m)$ from (r, c) also serves to break the pseudorandomness of F_k.

For the second claim, we assume without loss of generality that the distinguisher has m or c hardcoded, and thus the question reduces to simulating $F_k(r)$. For most alternative keys k' it must be the case that $F_k(r) \neq F_{k'}(r)$ by pseudorandomness, and the evidence permits R to equivocate between secrets k and k'. Any simulator S must fail to output at least one of these strings with noticeable probability, and R can choose this one to evade simulation. □

The above theorem leverages the strength of the respondent's key management within her own mind and the weakness of the government's evidence in preventing R from equivocating. If either of these two properties changes, then decryption might be compellable. Essentially: if there exists any method for the government to decrypt data without your help, then they can instead compel you to do so.

Theorem 2.5.5.2. *If the government knows evidence E and a PPT algorithm K such that $s \leftarrow K^N$ recovers R's secret key s, then there is a simulator S such that compelled decryption of a known ciphertext c is a foregone conclusion under E and S.*

Proof. Construct the simulator S^N that runs K^N, fetches the ciphertext from the known location, and uses the key to decrypt the ciphertext. This simulator is efficient and it perfectly emulates the real transcript. □

This theorem applies broadly to several categories of encryption schemes: enterprise or cloud backup systems that use an external key (e.g., one stored in a Hardware Security Module), threshold encryption with a threshold smaller than the full number of parties since the Fifth Amendment only protects against *self*-incrimination, and exceptional access systems that permit law enforcement access to encrypted devices via a key known to the vendor [62, 278], one or more courts [77, 204], law enforcement [62], or the device itself [258]. In all such cases, the existence of an alternative key bypasses the testimonial aspects of the respondent's assistance.

In the next section, we show specific constructions of secure multi-party systems that remain resilient to compelled actions; these can be used to build threshold and backup systems with stronger Fifth Amendment protections.

2.6 Resilience against compelled requests

So far, we have only considered how *past* actions impact whether or not a *current* compelled request is foregone. In this section, we ask whether a *current* protocol execution may open parties up to *future* compelled requests. If running a protocol does not open a party up to additional compelled requests, we call it *FC-resilient.* In this section, we formally define FC-resilience, design and implement a 2-party secure computation protocol that is both malicious secure and FC-resilient for one party, and leverage differential privacy to design a multi-party computation protocol that is FC-resilient for many parties.

2.6.1 Defining FC-resilience

In this section, we ask whether running protocols that are unrelated to any current legal issues will open the parties up to *future* compelled requests that would not have been possible before running the protocol. To see why this is an issue, consider the following scenario: Alice participates in a multi-party computation with several other parties, including Bob, in which she and Bob receive the same output. Later, Alice is the target of a compelled request in which the government seeks the result of the computation. Since the government could access the information without involving Alice (by compelling testimony from Bob instead), the output of the protocol is a foregone conclusion and Alice must provide it. Depending on the function computed, this may reveal information about Alice's secret inputs that was not previously compellable because it had only been stored in Alice's mind.

We provide a proactive cryptographic countermeasure against the above scenario, which we dub *FC-resilience.* Informally, we say that a protocol is FC-resilient if all compelled actions that are foregone after running the protocol were already foregone before running the protocol.

Model. Concretely, we consider an interactive protocol Π between $n+1$ parties $P_*, P_1, \ldots, P_n$ that is secure for computing function f with the $n+1$ parties' inputs up to abort and with erasures. If P_* has just as much ability to equivocate on any compelled action before running the protocol as after, then we say that Π is *FC-resilient for P_**.

We use the nomenclature that the government's evidence E *checks* a string X if it verifies that X exists in Nature at a public canonical location, returning **false** otherwise. (The evidence may still return false even if X does exist, unless some other conditions are met as well.)

In our setting, we presume the government knows that the parties have executed t timesteps of the protocol and that its evidence will check for this fact. Given a protocol Π and a timestamp t, we say that Π's modifications to nature in the $\mathcal{F}$-hybrid model with secure erasures, denoted M_t^{Π,P_*}, include the messages and local state of all protocol parties after running t steps of Π, except for P_*'s tapes for its communication with sub-module $\mathcal{F}$. (Formal modeling of the Turing machines of the parties was given in §2.3.2.)

We are now ready to define FC-resilience. Our definition requires that the execution of Π cannot subject P_* to any new compelled actions, no matter what time the government pauses Π to issue its request.

Definition 2.6.1.1 (FC-resilience for P_*)**.** Let protocol Π be a protocol among parties $P_*, P_1, \ldots, P_n$. Let E be an evidence machine. We say that Π is *FC-resilient for party P_** if the following holds true:

Suppose (C, G) is a foregone conclusion in the $\mathcal{F}$-hybrid model when addressing party P_* with respect to E_t^{Π}, S, for some $t \geq 0$, where E_t^{Π} runs machine E and also checks M_t^{Π,P_*}. Then there exist machines C_0, G_0, and S_0 such that: (1) (C_0, G_0) is a foregone conclusion with E and S_0; and (2) The two compelled disclosures have indistinguishable transcripts: $\forall\ R$, $\forall N$, $t({C_0}^{N,R}, {G_0}^{N}) \approx_c t(C^{N,R}, G^N)$

The idea is that although the government can now compel (C, G) after having witnessed up to all of the transcript of Π, it could have compelled an action (C_0, G_0) that yields an indistinguishable result, even without having seen the protocol transcript. Thus, the transcript of Π did not add any "new" possible information for which P_* can be compelled after running Π, but not before.

$\mathcal{F}$ separates P_*'s mind from local state. It would be convenient if we could keep all of P_*'s state as part of the "contents of her mind" rather than Nature. However, P_* is not likely to be storing her state or performing computations in her head. More likely, P_* will be doing these on a local computer, and she can only hold a small amount of state (e.g., a password) in her head.

To model this, we permit P_* to access an ideal sub-module which encapsulates both the small, long-term "state of the respondent's mind" as well as the limited operations that the respondent carefully performs only when she is not at risk of being compelled. Qualitatively, it is preferable to minimize the number of times $\mathcal{F}$ is invoked and the state that it stores.

The formal design of this sub-module is inspired by the treatment of tamper-proof hardware tokens in UC [187]. However, it represents something very different in this model: the occasions when the party is "currently using" the limited long-term state of the mind. The model prevents this state from entering Nature during the computation. However, any function of the output of this sub-module does become part of Nature, and it is incumbent upon P_* to choose a functionality $\mathcal{F}$ whose outputs don't trivially cause new compelled action to become foregone conclusions.

Different possibilities for the actual functionality of $\mathcal{F}$ are possible depending on how assured P_* is of a lack of sudden compelled requests. In this work, we consider the functionality $\mathcal{F}_{\text{pbkdf}}$ that computes a PBKDF of the party's password in a safe space. This functionality is described in Fig 2·2. $\mathcal{F}_{\text{pbkdf}}$ samples a long-term password from a distribution with sufficient min-entropy λ and then runs a password-based key derivation function on demand.

One might worry that using a password-derived key would subject our construction to password brute-force attacks that would not occur with a non-password-derived key K. Fortunately, as long as we store the PBKDF salt in the same manner that K would have been stored (e.g., in a trusted enclave), then our password-derived key resists brute-force attacks and retains the same cybersecurity protections as K.

```
Functionality $\mathcal{F}_{\text{pbkdf}}$
Public parameters: $\lambda$, PBKDF $f : \{0,1\}^* \to \{0,1\}^\lambda$
Setup: Upon receiving setup from $P$, do the following:
    1. If there is already a stored pw, halt
    2. Generate a random pw from a distribution with good min-entropy
    3. Generate a random salt uniformly at random with good entropy
    4. Store pw
    5. Output salt to $P$
Refresh: Upon receiving refresh from $P$, do the following:
    1. Generate a random salt uniformly at random with good entropy
    2. Output salt to $P$
Query: Upon receiving $(\mathsf{query}, \mathsf{salt}, m)$ from $P$:
    1. Check whether there is a stored pw. If there is not, halt
    2. Output $f(\mathsf{pw}, \mathsf{salt}, m)$ to $P$
```

Figure 2·2: *Ideal functionality for $\mathcal{F}_{pbkdf}$, a possible version of $\mathcal{F}$, which assumes P will not be compelled while computing a PBKDF of her password*

Government may compel at any time. Just as our foregone conclusion definition gave the government the strong power to view anything in the rest of the world, our FC-resilience definition allows the government full freedom to determine when to make its compelled request. An FC-resilient protocol must maintain protection against compelled requests made against P_* whether the protocol has completed or has been interrupted partway through (e.g., with intermediate state that has not yet been deleted). We presume that compelled requests only occur at one instant of the protocol execution; because the government is non-censoring, we presume that parties can alert each other to abort the protocol if they have been compelled to disclose information. Due to our composition theorem, it suffices to consider a single compelled request made by the government to P_*. Finally, we presume that the government is aware of the protocol execution.

2.6.2 FC-resilient two-party computation

In this section, we design and implement secure 2-party computation protocols based on Yao's garbled circuits [325] that are FC-resilient for one party in the $\mathcal{F}_{\text{pbkdf}}$-hybrid setting.

This is a non-trivial objective: While executing most MPC protocols, the parties' inputs and intermediate state are typically all foregone conclusions for the simple reason that all the (large) state is distributed throughout Nature rather than being stored within anyone's mind. This compelling adversary violates the non-collusion assumption required for secure MPC (even if the original protocol was malicious secure, or handled adaptive or mobile adversaries).

Using fully homomorphic encryption (FHE) can protect against compelled disclosure because compelled decryption is not a foregone conclusion (§2.5.5). For faster performance, we construct and implement a new secure computation protocol that is resilient to government compelled disclosure without the need for FHE. Our protocol involves careful modifications to Yao's garbled circuits at the input and output stages. It assumes secure deletion and a reliable communication channel whereby the parties can halt the secure computation if any or all of them are compelled to provide their state.

Construction of FC-resilient 2PC

We consider Yao's garbled circuits where the garbler additionally has access to the ideal module $\mathcal{F}_{\text{pbkdf}}$. For now assume that only the garbler receives output from the 2PC; we will relax this assumption later. Our method maintains malicious security against the evaluator, and it is compatible with two methods for ensuring malicious security against the garbler: cut-and-choose [213, §3.3] and authenticated garbling [313].

The main idea is that the garbler will "self-garble" tables for her input and output wires, so that even she does not know how to interpret her input or output without re-entering her password. The garbler inputs her password into the $\mathcal{F}_{\text{pbkdf}}$ module during three phases of the protocol. First, during pre-computation when preparing the garbled circuits, the garbler generates labels for the input wires uniformly at random (as normal) and augments these labels with a pseudorandom tag that is based on the PBKDF. For the outputs to the circuit, garbler appends no-op gates to the circuit where the output wire labels are again chosen pseudorandomly using the PBKDF. She then securely deletes the mapping of wire labels for her input and output bits, so that it can only be reconstructed with her own password. Second, upon receiving her own input, the garbler uses the PBKDF again and matches the resulting values with the pre-computed tags; this informs the garbler which wire labels to send to the evaluator while safeguarding the input itself. Third, at the end of the protocol, the garbler uses her PBKDF to find the outputs by using the output tables of the no-op gates. The concrete self-garbled tables for authenticated garbling are shown in Table 2.3.

Figure 2·3 depicts the full protocol Π, including a detailed description of all changes to garbled circuits compatible with the cut-and-choose approach. In total, our construction imposes an additive overhead to Yao's garbled circuits equal to a constant number of PBKDF calls per input and output wire. We emphasize that neither the password sub-module nor the garbler's password are required during circuit evaluation; they are only used at the beginning and end to provide input and read output.

Theorem 2.6.2.1 (simplified)**.** *Under the same cryptographic assumptions as*

Tag	**Self-garbled masked input** ($\hat{x}_w$)
$PBKDF(w, x_w = 0)_{1 \cdots n-1}$	$PBKDF(w, x_w = 0)_n \oplus \lambda_w$
$PBKDF(w, x_w = 1)_{1 \cdots n-1}$	$PBKDF(w, x_w = 1)_n \oplus (\lambda_w \oplus 1)$

(a) Self-garbled input tables for wire w (permutation not shown)

Info from E	**Self-garbled masked output** (x_w)
$\hat{x}_w = 0, s_w = 0$	$PBKDF(w, \hat{x}_w = 0, s_w = 0) \oplus r_w$
$\hat{x}_w = 0, s_w = 1$	$PBKDF(w, \hat{x}_w = 0, s_w = 1) \oplus (r_w \oplus 1)$
$\hat{x}_w = 1, s_w = 0$	$PBKDF(w, \hat{x}_w = 1, s_w = 0) \oplus (r_w \oplus 1)$
$\hat{x}_w = 1, s_w = 1$	$PBKDF(w, \hat{x}_w = 1, s_w = 1) \oplus r_w$

(b) Self-garbled output tables for wire w

Table 2.3: *Self-garbled tables for the garbler in the authenticated-garbling-based 2PC protocol FC-resilient for the garbler. w is the wire index, x is the true wire value, $\hat{x}$ is the masked wire value, and $\lambda = r \oplus s$ is the mask on the wire. r was held by the garbler during pre-processing but was securely deleted; s is held by the evaluator.*

malicious-secure Yao's garbled circuits, protocol Π *in Figure 2·3 is secure against malicious adversaries and is FC-resilient for the garbler.*

We prove this theorem by proving malicious security and FC-resilience individually, after proving some preliminary useful lemmas.

Definition 2.6.2.2 (Straightforwardly compellable)**.** We say a value x is *straightforwardly compellable* (or just *compellable*) under evidence E if there exists efficient C that makes no calls to R, and exists G such that $x \in t(C, G)$.

Lemma 2.6.2.3. *Let X be a public distribution. Then $x \sim X$ is straightforwardly compellable under any E.*

If X is a distribution described at a public location in N, then $x \sim X$ is straightforwardly compellable under E that checks that X is described at the proper location in N.

Proof. For the first statement, C can simply sample its own fresh $x \sim X$. For the second statement, C can query N to get X, then sample $x \sim X$. (Oftentimes, X is a point mass.) □

Lemma 2.6.2.4. *Let x be straightforwardly compellable under evidence E. Let C, G be such that $t(C^{N,R}, G^N) = x$ for all N and allowed R. Then there exist efficient C_0,*

G_0, and S, where C_0 does not query R, such that (C_0, G_0) is a foregone conclusion with respect to E and S.

Proof. Notice that the first three properties of a foregone conclusion are trivial to achieve (aside from S efficiency, which we will come back to). There exist efficient C_0 and G_0 that output x by Lemma 2.6.2.3. The evidence is the same, so the existence of an α-allowed R, and the non-censorship property of E, are unaltered.

S emulates the execution of C_0 and G_0 and outputs the result.

Thus, (C_0, G_0) is a foregone conclusion with respect to E and S. □

Lemma 2.6.2.5. *Let X be a distribution that is computationally indistinguishable from a public distribution Y. Then there exists C, G, S such that $t(C^{N,R}, G^N)$ contains a fresh sample from distribution X and S returns a fresh sample from Y. Then (C, G) is a foregone conclusion with respect to to S and any evidence E.*

Proof. We know that sampling a fresh $y \sim Y$ is straightforwardly compellable under any evidence E. Let (C, G) be a foregone conclusion with respect to E and a simulator S, where S samples $y \sim Y$ and outputs the result.

Consider probabilistic poly-time distinguisher D attempting to distinguish X from Y. Observe that if $x \sim X$ is *not* a foregone conclusion with respect to E and S, then D can distinguish X from Y. Thus, compelling a fresh sample x from C, G is a foregone conclusion with respect to E and S. □

Now that we have proven these lemmas, we can go on to prove the theorem itself. Our first goal is to ensure that this protocol retains malicious security against (separately) corrupt G and V. Informally, we will show that if this protocol does not have this property, then we can break either the PRF called by $\mathcal{F}_{\text{pbkdf}}$, or the security of the 2PC protocol described in [213].

Theorem 2.6.2.6. *Let $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$. Let Π be an instantiation of our protocol (Figure 2·3) for f. Assume that the oblivious transfer protocol is secure, that we have a perfectly-binding commitment scheme (used by the garbler) and a perfectly-hiding commitment scheme (used by the evaluator for coin-tossing), and that $\mathcal{F}_{pbkdf}$ generates a password with good min-entropy and contains a good pseudorandom function. Then, Π securely computes f.*

Proof of Theorem 2.6.2.6. We prove this via a series of hybrids that show that an execution of this protocol (using circuit C_2) is indistinguishable from an execution of the protocol of [213], which we know securely computes f under the same assumptions:

H_0 The real protocol described above

H_1 No permuted/masked tables – follow the protocol through step 6 (decommitment for check-circuits), but then skip steps 7, 8, 9a, and 9b. Instead, keep the original garbled tables and go straight to step 9c.

H_2 Same as H_1, but instead of sampling the outputs of the no-op gate $w_{c,i,b}$ from $\mathcal{F}_{\text{pbkdf}}$, G instead chooses them as normal wire labels (random values).

H_3 The ideal functionality for a 2PC computing $f(x, y)$.

It is easy to see that $\mathsf{H}_0 \equiv \mathsf{H}_1$: Observe that the messages sent in each protocol are exactly the same; the only changes are to G's local state. G clearly cannot learn anything new about V's input in H_0 that it couldn't learn in H_1, since all the extra computation happens on information G already had. Since the distributions of H_0 and H_1 are equal for all x and y, no distinguisher can tell which game we are playing.

We next show that $\mathsf{H}_1 \approx_c \mathsf{H}_2$. Let $\mathcal{A}$ be an adversary attempting to distinguish $\mathcal{F}_{\text{pbkdf}}$ from random. Suppose it has access to $\mathcal{A}'$ which can distinguish H_1 from H_2. $\mathcal{A}$ first calls $\mathcal{F}_{\text{pbkdf}}(\mathsf{setup}, 2\lambda)$. It then runs an entire execution of the protocol in Figure 2·3 and inputs the transcript into $\mathcal{A}'$. When it computes the wire labels for the output wire, it sets $w_{c,\mathsf{out},b} = \mathcal{F}_{\text{pbkdf}}(\mathsf{query}, \mathsf{salt}, (c, i, b))$ for $c \in [s]$, $i \in [n]$, and $b \in \{0, 1\}$. When $\mathcal{A}'$ outputs its guess as to which hybrid it is in, $\mathcal{A}$ guesses that it is interacting with a random function if $\mathcal{A}'$ guessed H_2, and a pseudorandom function if it guessed H_1. It is not hard to see that $\mathcal{A}$ will have the same advantage as $\mathcal{A}'$. Thus, the security of $\mathcal{F}_{\text{pbkdf}}$ ensures that H_1 is indistinguishable from H_2.

Finally, to show that $\mathsf{H}_2 \approx_c \mathsf{H}_3$, we rely on the original security proof of [213]. Observe that H_2 is the protocol of [213]. We can use C_2 as the auxiliary input circuit (called C_0 in that paper) to that protocol, since it computes $f(x, y)$, The proof of security follows directly, for both a malicious garbler and a malicious evaluator. □

Finally, we wish to show that the protocol in Figure 2·3 is FC-resilient for the garbler.

Theorem 2.6.2.7. *Assuming secure deletion for the garbler G, and assuming the existence of a channel through which the parties can halt the computation if any of them is compelled, then the protocol in Figure 2·3 is FC-resilient for G.*

Proving this theorem will require keeping track of the changes the protocol makes to Nature over time. Given an interactive protocol Π, we wish to specify formally the state of nature N after t timesteps. This models a scenario where the protocol is interrupted by a request to compel information in the middle of the protocol. (After this interruption, we assume all parties abort and do not complete the protocol.)

Following the ITM model of Canetti [69], only one machine is deemed to be active in a given timestep. The active machine can write to or read from a tape, or move a head. This action could represent one step of local computation (writing/reading a local work tape) or communication to another party (by writing on that machine's communication input tape).

At any point in time, we consider the *configuration* of an interactive Turing machine to comprise the contents of all tapes (including input and output tapes), the current state, and the location of the head in each tape. We refer readers to [69, Def. 4] for more details.

The protocol's changes to nature consist of the entire state of Π at a given time except for the contents of P_*'s sub-module $\mathcal{F}$.

Definition 2.6.2.8 (Protocol modifications to Nature)**.** Let Π be a protocol among parties $P_*, P_1, \ldots, P_n$, where P_* has a sub-module $\mathcal{F}$. Given $t \in \mathbb{N}$, let Π's *modifications to nature* M_t^{Π,P_*} be the set of all messages sent to and from all parties up through t total timesteps of computation (by all parties), the current configuration of all parties except P_* after t timesteps, and the current configuration of P_* except its input tape, its output tape, and its tapes communicating with sub-module $\mathcal{F}$ (and the tapes of $\mathcal{F}$ itself).

Now we are ready to prove Theorem 2.6.2.7.

Proof of Theorem 2.6.2.7. Let Π be an instantiation of the protocol in Figure 2·3, and suppose it runs on inputs x and y for parties garbler G and evaluator V respectively. Let C, G be a compelled request, let E be an evidence machine that checks the auxiliary inputs of Π and V's input, and let S be a simulator. Suppose the compelled request is made after $t \in \mathbb{N}$ timesteps of computation of Π, and let $M_t^{\Pi,G}$ be the additions to Nature by timestep t, as in Definition 2.6.2.8. Let E_t check $M_t^{\Pi,G}$ and then run E. Say (C, G) is a (λ, α)-foregone conclusion with evidence E_t and simulator S.

We will prove the claim by showing that all elements of $M_t^{\Pi,G}$ are straightforwardly compellable under E.

First, notice that the satisfiability and non-censoring properties are met: E checks a subset of the locations in N compared to E_t, so all α-allowed R from the post-protocol foregone conclusion are still α-allowed under E. And, by construction, E_t is non-censoring if and only if E is.

The remaining goal is to show that all elements of $M_t^{\Pi,G}$ are straightforwardly compellable. This will complete the proof, by Lemma 2.6.2.4 and by our composition theorem (Theorem 2.3.5.1).

We can lean on Lemma 2.6.2.3 to show this, and we can instead show that all values in $M_t^{\Pi,G}$ were generated from known distributions or could be generated from Nature.

Essentially, this means we must "simulate" all values added to $M_t^{\Pi,G}$ starting from only the auxiliary inputs and the other parties' inputs, and show that they could have been simulated *before* running the protocol. This will prove the claim, since this is the only additional information that the evidence E and simulator S can work with. Consider two cases, depending on t.

Case 1: The compelled request occurs before step 9. In this case, it is Pareto-optimal for the request to occur immediately before step 8. It is always better for the government for information to be added to N, and until step 8, $M_t^{\Pi,G}$ only grows as t increases. Only during step 8 is G's local state erased, so $M_t^{\Pi,G}$ is smaller for t in step 8 than it is for t immediately before step 8.

We proceed to demonstrate that we can simulate these values even before running the protocol. At t immediately before step 8, the information in $M_t^{\Pi,G}$ is:

- The auxiliary inputs for both parties and V's input. These exist before the protocol and were checked by E, and are compellable by Lemma 2.6.2.3.

- Preprocessing phase (steps 0, 1, 2a): All values generated by G during these steps are ephemeral random values with known distributions and are compellable by Lemma 2.6.2.3).
- Preprocessing phase (step 2b): The output wire labels are the output of a PRF with an ephemeral "key" (actually salt). By Lemma 2.5.4.1, compelling the wire labels is foregone.
- Evaluation Part 1 phase (steps 3-6): All new additions to $M_t^{\Pi,G}$ in this phase are either new ephemeral random values, or a function of straightforwardly compellable information.
- Preparation phase (step 7a): As discussed in §2.5, the results of the calling the PRF in step 7a straightforwardly compellable.
- Preparation phase (steps 7b, 7c): Step 7b involves only fresh random samples, and step 7c is a function of existing values.
- All information held by V. This is all a function of V's input and the messages sent to V (which already showed was compellable).

As mentioned, running step 8 will only limit the government's ability to create a foregone conclusion. Thus, for t before step 9 of Π, all values in $M_t^{\Pi,G}$ are straightforwardly compellable.

Case 2: The compelled request occurs during or after step 9.

If the compelled request occurs during or after step 9, then it is Pareto-optimal for the request to occur at the end of the protocol, for the same reasoning as in the previous case. We proceed to demonstrate how to simulate Part 2 of the evaluation. In this phase, we must also deal with G's input x, which is *not* efficiently compellable (though it also is not directly part of $M_t^{\Pi,G}$).

- Step 9: Now that the mappings of wire labels to values have been deleted, G has in effect "garbled" her own state so that simply writing m_{c,i,x_i} or w_{c,i,x_i} (or t_{c,i,x_i}) does not reveal x_i itself. These values cannot be compelled by indexing x_i (since the government does not know G's secret x_i), so the best the government can do in the existing C, G is compel one of these values indexed by an arbitrarily chosen bit b (unless E itself contains x_i). By Lemma 2.6.2.5, these values are compellable. Furthermore, all values computed in the decommitments are functions of values that were straightforwardly compellable.

- Step 10, 11a: All this is done by V, and thus is in N either way.
- Step 11b: First, note that the $w_{c,\mathsf{out},b}$ value was already compellable. Second, recall that the output tapes of $\mathcal{F}_{\text{pbkdf}}$ (and G's final output tape) are not included in $M_t^{\Pi,G}$. Thus, all the information in this step was already compellable.

So, for t after step 8 of Π, all values in $M_t^{\Pi,G}$ are compellable.

Thus, by Lemma 2.6.2.4 and the Sequential Composition Theorem (2.3.5.1), we know that if (C, G) was a foregone conclusion with respect to E_t and S and the request occurred after step 8 of the protocol, we can create machines C_0, G_0, S_0 such that $t({C_0}^{N,R}, G^N) = t(C^{N,R}, G^N)$ for all N, R and (C_0, G_0) is a foregone conclusion with respect to E and S_0. This shows that the protocol Π in Figure 2·3 is FC-resilient for the garbler and completes the proof. □

Implementation of FC-resilient 2PC

We implemented an FC-resilient two-party computation based on the authenticated garbling work of [313]. Our implementation was forked from `emp-toolkit` [314], and our source code can be found at this GitHub repository.[1]

The main part of our implementation was about 250 additional lines of code. The code contained two main changes: giving the output to the garbler rather than the evaluator, and implementing the the self-garbled tables shown in Table 2.3. The tables were created during function-dependent pre-processing, and accessed at the beginning and end of the online phase. The PBKDF used was Argon2i [43].

We emphasize that the added runtime is linear in the input/output wires, but is independent of the size of the circuit itself. To demonstrate this, we tested our implementation by running repeated iterations of SHA-256 while XORing the result with a "chaining" value as is done in computing PBKDF2 [289]. All experiments were

[1] https://github.com/sarahscheffler/password-ag2pc

performed on a Dell XPS 9370 laptop with an Intel i7-8650U processor and 16GB of RAM. The results are in Table 2.4; they show that the FC-resilience cost of running thousands of executions of a PBKDF (two per input wire, four per output wire) is costly for small circuits but quickly becomes negligible.

N		Total time	**FC-resil. parts**	Unmod. parts
1	G	1551 (15.48)	1161 (14.01)	389.9 (7.760)
	E	1406 (17.02)	1003 (15.37)	402.5 (7.270)
10	G	4758 (37.90)	1673 (15.74)	3084 (34.50)
	E	4612 (42.04)	1417 (17.98)	3195 (33.63)
100	G	35320 (1229)	2279 (175.9)	33040 (1089)
	E	35180 (1216)	1073 (134.1)	34106 (1127)

Table 2.4: *Performance times (ms) for our test implementation of FC-resilient authenticated garbling, computing N iterations of SHA-256. The average time over 10 runs is shown with the standard deviation in parentheses. Pre-processing and online times are combined.*

Constructing FC-resilient zero-knowledge proofs

ZKGC [182] is a zero knowledge proof of knowledge in which the verifier garbles a circuit and the prover evaluates the circuit using its witness as input. It follows from the Theorem 2.6.2.1 that ZKGC with self-garbled tables is FC-resilient for the verifier.

Corollary 2.6.2.9. *The ZKGC protocol combined with our self-garbled table construction is FC-resilient for the verifier.*

What about FC-resilience for the prover? Suppose Alice engages in an interactive zero-knowledge proof with Bob. Interactive zero-knowledge proofs are generally not transferable, from a cryptographic point of view. However, from a legal viewpoint, if the government wishes to investigate Alice, it can instruct Bob to disclose his interaction with her. Since Bob is not part of Alice's mind, he is part of Nature, and we presume that his testimony is truthful and can be added to the evidence E. Hence, the government can learn one bit about Alice based on testimony from Bob, so we believe that zero knowledge proofs cannot be FC-resilient for the prover. This makes it extremely challenging to create FC-resilient protocols where anyone other than P_* receives output it could not have already computed. We next describe one method for achieving this outcome.

2.6.3 FC-resilient multi-party computation

Whereas the constructions in the last section only provided results to one party, in this section we describe a technique that permits everyone to receive the output of a large n-party secure computation, using ideas from differential privacy. This construction uses the BMR multi-party garbled circuit protocol [26], and it only achieves semi-honest security.

From an FC-resilience perspective, there are two challenges that occur when multiple parties receive output. First, we require a more complicated output opening protocol that requires all n parties to use their passwords in order to read the final result. In the semi-honest setting, the self-garbled no-op gates from the previous section solve this problem: each party masks the output table with a PBKDF of their password during garbling, and then each party in sequence can de-garble the final output wire at the end of the protocol.

Second, any party must operate under the assumption that the result of the computation can be compelled by the other participants, so she must ensure that the result reveals very little about any party's input. We propose to address this issue by considering MPC applied to differentially private functions. This comes at the expense of requiring a looser distinguishing bound when defining foregone conclusions, since differential privacy does not guarantee negligible statistical distance between neighboring distributions.

More formally, we present the following lemma to show that differentially-private functionalities are foregone if we assume this looser distinguishing bound. Recall that a function f is differentially private if for all "neighboring" inputs X_1 and X_2, for all possible subsets of the image of f, $S \subseteq \mathsf{im}(f)$, we have $\Pr[f(X_1) \in S] \leq e^{\epsilon} \Pr[f(X_2) \in S]$, where ϵ is a parameter. In the context of MPC, inputs X_1 and X_2 are neighboring if they differ by the inclusion/exclusion of one party P_*'s input.

Lemma 2.6.3.1. *Let f be a differentially-private mechanism, X be a dataset in Nature, and $X' = X \cup \{R.s\}$. Consider the compelled action $C := f(X')$ with the evidence E that checks for the existence of a secret input $R.s$. Then there exists a simulator S such that C is foregone with respect to E and S.*

Proof. Construct the simulator S^N that queries Nature to recover X and then returns $y \leftarrow f(X)$. Differential privacy guarantees that the two distributions $C^{N,R}$ and S^N are e^{ϵ}-statistically close (over the coins of f), as desired. □

From a scientific perspective, widening the distinguishing bound makes actions easier to compel, which has a two-sided impact on FC-resilience: the evidence gathered from voluntarily executing a cryptographic protocol might be used to compel more functionalities, but these functionalities may also have been compellable beforehand. The question then arises: what distinguishing bound is desired by the courts? This appears to be an open question: although the burden of proof to show that an action is a foregone conclusion is on the government, "how high that burden is remains surprisingly unclear" [193] from existing case law. Possible evidentiary standards to apply in foregone conclusion cases include the "reasonable suspicion" standard [281] that almost certainly does allow for noticeable chance of error, and the "beyond a reasonable doubt" standard that may not [311]. While we have taken the approach in this work that a negligible error rate is desirable since it suffices "to protect the innocent who otherwise might be ensnared by ambiguous circumstances" [270] and it yields an appealing composition property, our definition is easily extensible to any choice that courts decide as a policy decision.

2.7 Conclusion

This work initiates a scientific study of disclosures compelled by the U.S. government under the foregone conclusion doctrine. We provide a cryptographic security definition that is grounded in the law but that can be used by security researchers without the need to understand the law. We show that existing cryptosystems can be vulnerable to this threat, yet it is possible to design countermeasures at reasonable cost.

Beyond this paper's scientific contributions, this work also has significant bearing on a potential upcoming Supreme Court case. As we discuss in §2.4.1, state Supreme Courts and lower federal courts are divided on the issue of compelled decryption under the foregone conclusion doctrine. Legal scholars believe that the U.S. Supreme Court will take one of these cases soon to resolve the issue [195]. For this case to come to a sound conclusion, the courts must analyze the foregone conclusion doctrine from many perspectives. We hope that the technical lens provided by this paper will shine new light on the doctrine that was not provided by prior legal analysis. The Supreme Court's decision will impact compelled decryption for the foreseeable future; we can only hope that the result is not already a foregone conclusion.

Input: G has input $x \in \{0,1\}^n$, V has input $y \in \{0,1\}^n$.

Auxiliary input and setup: G has access to $\mathcal{F}_{\text{pbkdf}}$ and has already run $\mathcal{F}_{\text{pbkdf}}(\mathsf{setup}, 2\lambda)$. Both parties have λ, s, and a description of circuit C_1 such that $C_1(x,y) = f(x,y)$. The parties have a reliable channel through which they can indicate that they were the target of a compelled request.

Output: G receives $f(x,y)$,V receives no output.

Throughout the protocol, if either party receives a compelled request, communicate this on the reliable channel and halt.

Preprocessing:

0. G generates a fresh salt $\mathsf{salt} \leftarrow \mathcal{F}_{\text{pbkdf}}(\mathsf{refresh})$.
1. Circuit construction and modification: The parties append a single no-op gate to the output of C_1; call the resulting circuit C_2. The parties then modify C_2 as in [213].
2. Commitment Construction:
   (a) Output Wire Label Generation: Let the output wire of C_2 (the output of the no-op gate) be indexed by out. For $c \in [s]$ and for $b \in \{0,1\}$, G creates $w_{c,\mathsf{out},b} = \mathcal{F}_{\text{pbkdf}}(\mathsf{query}, \mathsf{salt}, (c, \mathsf{out}, b))_{1\cdots\lambda}$. No table from this wire to the output is created.
   (b) Remaining Circuit and Commitment Generation: G generates the rest of the wire labels and garbled tables as normal. She also computes commitments to V's input wires, and commitment-sets for her own input wires as normal (see [213]). Let $w_{c,i,b}$ be the wire label for circuit c, input wire i, and wire value b. Let $r_{c,i,b}$ be the commitment randomness used to commit to $w_{c,i,b}$.

Evaluation Part 1 (V's input known):

3. Execute Oblivious Transfers as usual
4. Send Circuits and Commitments. (Note that V does not have the ability to map the final gate output to 0 or 1, it only gets $w_{c,\mathsf{out},b}$ which it must send back to G.)
5. Prepare challenge strings
6. Decommitment phase for check-circuits. Let the remaining set of evaluation circuits be $C \subset [n]$.

Continued in Fig. 2·3b.

(a)

Figure 2·3: *2PC protocol Π that securely computes $f : \{0,1\}^n \times \{0,1\}^n \rightarrow \{0,1\}$ among garbler G and evaluator V with malicious security for both parties, and with FC-resilience for the garbler. (Continued in Fig. 2·3b)*

Continued from Fig. 2·3a.

Preparation for G input:

7. Generation of Permuted Tables
    (a) Tag and mask sampling: For $c \in C, i \in [n], b \in \{0,1\}$, G queries $z_{c,i,b} = \mathcal{F}_{\text{pbkdf}}(\mathsf{query}, \mathsf{salt}, (c,i,b))$, and parses $z_{c,i,b} = (t_{c,i,b}, m_{c,i,b})$ as a "tag" $t_{c,i,b}$ and a "mask" $m_{c,i,b}$, each of length λ.
    (b) Permutation bit sampling: G picks $p_{c,i} \leftarrow \{0,1\}$ uniformly at random for $c \in C, i \in [n]$.
    (c) Build permuted masked tables: For each circuit $c \in C$, for each input wire $i \in [n]$, mask the wire value and commitment randomness with $m_{c,i,b}$, label it with tag $t_{c,i,b}$ and permute the rows for $b = 0$ and $b = 1$ depending on $p_{c,i}$. That is, record rows $(c, i, t_{c,i,p_{c,i}}, m_{c,i,p_{c,i}} \oplus (w_{c,i,p_{c,i}} \| r_{c,i,p_{c,i}}))$ and $(c, i, t_{c,i,1-p_{c,i}}, m_{c,i,1-p_{c,i}} \oplus (w_{c,i,1-p_{c,i}} \| r_{c,i,1-p_{c,i}}))$ ordered by $p_{c,i}$.
8. Secure deletion: G securely deletes all information except the permuted table from step 7c and the salt.

Evaluation Part 2 (G's input known):

9. Decommitment phase for G's input in evaluation-circuits:
    (a) Now that G has her input $x = x_1 \cdots x_n$, she recomputes $(t_{c,i,x_i}, m_{c,i,x_i}) = \mathcal{F}_{\text{pbkdf}}(\mathsf{query}, \mathsf{salt}, (c,i,x_i))$ for $c \in C, i \in [n]$.
    (b) G finds the table row indexed by c, i, t_{c,i,x_i} and uses m_{c,i,x_i} to unmask w_{c,i,x_i} and r_{c,i,x_i}.
    (c) G decommits to w_{c,i,x_i} and sends that to V (as usual).
10. 10. Correctness and consistency checks
11. 11. Circuit evaluation
    (a) For circuits $c \in C$, V evaluates the circuit up until the final wire label, $w_{c,\mathsf{out},b}$. It sends this value back to G.
    (b) G queries $w'_{c,\mathsf{out},b'} = \mathcal{F}_{\text{pbkdf}}(\mathsf{query}, \mathsf{salt}, (c, \mathsf{out}, b'))$ for circuits $c \in C$ and $b' \in \{0,1\}$. If neither of these match the value sent by V, G aborts and outputs $\perp$. If $w'_{c,\mathsf{out},b'}$ matches one of the values sent by V for all $c \in C$, then G takes the b' that appears the most and outputs it.

(b)

Figure 2·3: *2PC protocol Π that securely computes $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ among garbler G and evaluator V with malicious security for both parties, and with FC-resilience for the garbler. (Continued from Fig. 2·3a)*

Chapter 3

Fairly post-processing calibrated non-binary classifiers to binary decisions

This chapter is based on joint work with Ran Canetti, Aloni Cohen, Nishanth Dikkala, Govind Ramnarayan, and Adam Smith [70].

3.1 Introduction

In this chapter, we turn to the question of *fair post-processing.* We suppose that individuals belong to one of two or more disjoint *protected groups*. Our overall task is to decide whether a given individual has some hidden binary property B in a way that ensures "fair balancing of errors" across the groups. To this end, we consider the following two-stage mechanism. Stage 1 is a nonbinary ("soft") classifier $\hat{S}$ which, when run on input x, returns a nonbinary score $s \in [0, 1]$, representing the classifier's guess at the chance that x has property B. To ensure this score s is "meaningful," we require that $\hat{S}$ be *groupwise calibrated*: within each group, for each $s \in [0, 1]$, the fraction of individuals in the group that get score s is s. This stage's goal is to gather information and provide the best accuracy possible, with minimal regard to fairness. Stage 2 takes as input the output $s = \hat{S}(x)$ of the first stage, and x's group. It then outputs a binary ("hard") decision: its best guess at whether x has property B. This stage is aimed at distributing the errors of the first process "fairly." The main thrust of our work is to design the second stage.

Following [81, 162] we consider the following four performance measures based on the confusion matrix of the resulting hard classifier. *Positive predictive value (PPV)* is the fraction of individuals that have the property among all the individuals the classifier predicted to have the property. *Negative predictive value (NPV)* is defined analogously. The *false positive rate (FPR)* is the fraction of individuals predicted to have the property among all individuals that actually do not have the property. *False negative rate (FNR)* is defined analogously. Ideally, we would seek to equalize all four of these measures between the groups: that is, the PPV (resp. NPV, FPR, FNR) for group 1 is equal to the PPV (resp. NPV, FPR, FNR) for group 2. Unfortunately, this is impossible in general [81, 200]. Our broad motivating question is thus:

> Under what conditions can we post-process a calibrated soft classifier's outputs so that the resulting hard classifier equates a subset of $\{\mathsf{PPV}, \mathsf{NPV}, \mathsf{FNR}, \mathsf{FPR}\}$ across a set of protected groups? How can we balance these conflicting goals?

Our first set of results involves post-processing via a simple threshold mechanism. We will show in §3.3 that using a single threshold across all groups cannot in general guarantee equality of either PPV or NPV across groups. Using different thresholds for the different groups, *either* PPV or NPV can be equalized, but not both, assuming a non-degeneracy property on the outputs of $\hat{S}$.

Our second set of results (§3.5) allows the hard classifier to *defer* some of its outputs, that is, to respond with $\perp$ or "I don't know." The deferred inputs are then sent to a different process; we describe the nuances of this choice in §3.7. With deferrals, one can equalize both the PPV and NPV of the groups using thresholds, conditioned on the output not being $\perp$. Beyond thresholds, we describe several strategies for equalizing all four quantities (PPV, NPV, FPR, FNR) across all groups.

3.1.1 Related work

We briefly describe the works most closely related to ours, though both the list of works and their summaries are inevitably too short. Our work fits in a research program on group fairness notions following the work of Chouldechova [81] and Kleinberg et al. [200]. Those works demonstrate the inherent infeasibility of simultaneously equalizing a collection of measures of group accuracy. Our work considers the notions of calibration as formalized in [248] and those of PPV, NPV, FPR, and FNR from [81] and [200].

The power of post-processing calibrated scores into decisions using threshold classifiers in the context of fairness has been previously studied by Corbett-Davies, Pierson, Feller, Goel, and Huq [92]. As in our work, they show that it is feasible to equalize certain statistical fairness notions across groups using (possibly different) thresholds. They additionally show that these thresholds are in some sense optimal. Whereas [92] focuses on statistical parity, conditional statistical parity, and false positive rate, our most comparable results consider PPV. In our work, we further show that in some cases thresholds fail to equalize both PPV and NPV (called *predictive parity* by [81]), unless we also allow our post-processor to defer on some inputs. Our work also studies methods of post-processing that are much more powerful than thresholding, especially when allowing deferrals. On the technical side, [92] assumes that their soft classifiers are supported on the continuous interval $[0,1]$, simplifying the analyses. We instead study classifiers with finite support as it is closer to true practice in many settings (e.g., COMPAS risk scores).

Using deferrals to promote fairness has been considered also in the work of Madras, Pitassi, and Zemel [217]. Specifically they consider how deferring on some inputs may promote a combination of accuracy and fairness, especially when taking explicit account of the downstream decision maker. They make use of two-threshold deferring post-processors like those discussed in Section 5. While it helped inform our work, [217] takes a more experimental approach and focuses on minimizing the "disparate impact," a measure of total difference in classification error between groups, while maximizing accuracy. One important difference between our works is that Madras et al. distinguish between "rejecting" and "deferring." Rejecting is oblivious as to properties of the downstream decision maker, while deferring tries to counteract the biases of the decision maker. Our work considers only the former notion, but uses the term "defer" instead of "reject."

An earlier chapter in algorithmic fairness

The most modern incarnation of the field of algorithmic fairness is still in a nascent rapidly-changing phase, where the methods are highly variable and it is not yet clear which parts of the field will coalesce into long-lasting areas of study. However, the concepts in the field are much older than they seem. Cole and Zieky provide an excellent summary of fairness research in the 1960s and 70s applied to the problem of standardized testing in the civil rights movement [87], and Hutchinson et al. [171] provide an in-depth analysis of how that research relates to modern algorithmic fairness as applied to machine learning. Several definitions were proposed in the late 1960s and early 70s, most prominently Cleary's definition of fair prediction via underpredictions for one group (highly related to groupwise calibration) [83], Kirkpatrick's methods for predicting job performance using biased measurement tools [199], Thorndike's notion of fair representation [286], and Darlington's four fairness definitions for correlations which roughly correspond to calibration, proportionality, equalized odds, and demographic parity [102]. In the midst of all these proposed definitions to resolve inequities in standardized testing, some researchers began to notice that these definitions are inherently incompatible with each other unless the underlying dataset has some specific convenient properties [245].

These incompatibilities were re-discovered in the setting of machine learning in Kleinberg, Mullainathan, and Raghavan's influential work *Inherent trade-offs in the fair determination of risk scores* [200], and independently by Chouldechova in *Fair prediction with disparate impact* [81]. Essentially, these two works show two variants of the same theme: the four main error metrics we would wish to equalize between groups, positive predictive value (PPV), negative predictive value (NPV), false positive rate (FPR), and false negative rate (FNR), cannot all be simultaneously equalized between groups simultaneously. The only exception to this rule is when the groups have equal base rates between the two groups, or the classifier makes no errors. As we described in §1.2.2, this impossibility result greatly limits the possible mathematical approaches to achieving fairness.

3.1.2 Organization

We list several useful definitions and preliminaries in §3.2. §3.3 contains our main analysis of post-processing groupwise-calibrated soft classifiers with thresholds; some additional extensions of this analysis are presented in §3.4. In §3.5, we present our results on post-processing groupwise-calibrated soft classifiers with deferrals, with several different mechanisms. In §3.6 we provide some experimental data using our methods, and in §3.7 we discuss various deferral models.

3.2 Preliminaries

We study the problem of binary classification. An *instance* is an element, usually denoted x, of a universe $\mathcal{X}$. We restrict our attention to instances sampled uniformly at random from the universe, denoted $X \sim \mathcal{X}$. Our theory extends directly to any other distribution on $\mathcal{X}$; that distribution does not need to be known to the classifiers. Each instance x is associated with a *true type* $Y(x) \in \{0, 1\}$. Each instance x is also associated with a *group* $G(x) \in \mathcal{G}$, where $\mathcal{G}$ is the set of groups. We restrict our attention to sets $\mathcal{G}$ that form a partition of the universe $\mathcal{X}$. We denote by $\mathcal{X}_g$ the set of instances x in group g, and by X_g the random variable distributed uniformly over $\mathcal{X}_g$. Note that for any events E_1 and E_2, $\Pr_{X \sim \mathcal{X}_g}[E_1 \mid E_2] = \Pr_{X \sim \mathcal{X}}[E_1 \mid E_2, G(X) = g]$.

Definition 3.2.0.1 (Base rate (BR))**.** The *base rate* of a group $g \in \mathcal{G}$, is

$$\mathsf{BR}_g = \Pr[Y(X_g) = 1] = \mathbb{E}[Y(X_g)]. \tag{3.1}$$

When $\mathcal{X}$ is finite, BR_g is simply the fraction of individuals x in the group g for whom $Y(x) = 1$.

A *classifier* is a randomized function with domain $\mathcal{X} \times \mathcal{G}$.[1]A *hard classifier*, denoted $\hat{Y}$, outputs a *prediction* in $\{0, 1\}$, interpreted as a guess of the true type $Y(x)$. A *soft classifier*, denoted $\hat{S}$, outputs a *score* $s \in [0, 1]$, interpreted as a measure of confidence that $Y(x) = 1$. We restrict our attention to soft classifiers with finite image. We call a classifier *group blind* if its output is independent of the input group g. For all groups $g \in \mathcal{G}$, we call a hard classifier $\hat{Y}$ *non-trivial on* g if $\Pr[\hat{Y}(X_g) = 1] > 0$ and $\Pr[\hat{Y}(X_g) = 0] > 0$. Hard classifiers are *trivial on* g if they are not non-trivial on g.

A *post-processor* is a randomized function with domain $[0, 1] \times \mathcal{G}$. As with classifiers, a post-processor can be *hard* or *soft*. A hard post-processor, denoted $\hat{D}$, outputs a prediction in $\{0, 1\}$. A soft post-processor, denoted $\hat{D}^{\mathsf{soft}}$, outputs a score $s \in [0, 1]$. Observe that for a soft classifier $\hat{S}$, $\hat{D} \circ \hat{S}$ is a hard classifier, and $\hat{D}^{\mathsf{soft}} \circ \hat{S}$ is a soft classifier. As with classifiers, we call a post-processor group blind if its output is independent of the group g, and we restrict our attention to post-processors with finite image. The restriction to finite image is for mathematical convenience (and also because digital memory leads to discrete universes); our results generalize to infinite images as well.

In Section 3.5, we expand the definitions of both classifier and post-processors to allow an additional input or output: the special symbol $\perp$.

As the focus of this paper is on the post processing of classifiers, we set aside questions such as the origin of a given classifier, including the randomness used in training, the origin or quality of the training data, and societal factors affecting the classifier. In particular, the classifiers we consider in this work are memoryless: they do not remember inputs or random choices from previous invocations. That is, we assume that if X, X' are two independent random variables drawn from $\mathcal{X}$ then $\hat{S}(X)$ and $\hat{S}(X')$ (respectively $\hat{Y}(X)$ and $\hat{Y}(X')$) are also independent random variables. Our formalism can be naturally extended also to classifiers with initial randomized preprocessing, by considering the family of derivative classifiers, where for each derivative classifier the random choices made at preprocessing are fixed to some value. The formalism can then be applied separately to each derivative classifier.

$$\begin{tikzcd}[column sep=huge]
x \in \mathcal{X} \arrow[r, "\text{“soft”}", "\hat{S}"'] \arrow[rr, bend left, "\text{“hard”}", "\hat{Y}"'] & s \in [0,1] \arrow[r, "\text{“hard post-processor”}", "\hat{D}"'] \arrow[loop below, "\hat{D}^{\mathsf{soft}}", "\text{“soft post-processor”}"'] & \hat{y} \in \{0,1\}
\end{tikzcd}$$

Figure 3·1: *We call a classifier that returns results in* $[0,1]$ *a* soft classifier *to differentiate it from those which return results in* $\{0,1\}$, *which we call* hard classifiers. *We refer to classifiers that take as input the output of a soft classifier as* post-processors.

3.2.1 Calibration

Several works concerning algorithmic fairness focus on various notions of *calibration*. The following calibration notions are defined only over soft classifiers:

Definition 3.2.1.1 (Calibration (Soft)). We say a soft classifier $\hat{S}$ is *calibrated* if $\forall s \in [0,1]$ for which $\Pr_{X\sim\mathcal{X}}[\hat{S}(X) = s] > 0$,

$$\Pr_{X\sim\mathcal{X}}[Y(X) = 1 \mid \hat{S}(X) = s] = \mathop{\mathbb{E}}_{X\sim\mathcal{X}}[Y(X) \mid \hat{S}(X) = s] = s.$$

The probability above is taken over the sampling of X, as well as random choices made by $\hat{S}$ at classification time.

Definition 3.2.1.2 (Groupwise Calibration (Soft))**.** We say that a soft classifier $\hat{S}$ is *groupwise calibrated* if it is calibrated within all groups. That is, $\forall g \in \mathcal{G}$ and $\forall s \in [0,1]$ for which $\Pr[\hat{S}(X_g) = s] > 0$, we have that

$$\Pr[Y(X_g) = 1 \mid \hat{S}(X_g) = s] = s.$$

Groupwise calibration is essentially the same notion as *multicalibration* [164] with the difference that in their case the true types are values in $[0,1]$. We use a different term to emphasize that we restrict our attention to collections of groups $\mathcal{G}$ that form a partition of the universe $\mathcal{X}$.

The two definitions above are stated for soft classifiers whose output distribution is discrete, since we must be able to condition on the event $\hat{S}(X) = s$ or $\hat{S}(X_g) = s$. That said, it extends naturally to classifiers with continuously-distributed outputs provided that the conditional probabilities are well defined.

3.2.2 Accuracy Profiles (APs)

Throughout this work, we make repeated reference to the probability mass function of the random variable $\hat{S}(X_g)$ for a calibrated soft classifier $\hat{S}$ acting on a randomly distributed input X_g. We call this distribution on calibrated scores an *accuracy profile* (AP).

Definition 3.2.2.1 (Accuracy Profile (AP))**.** The *accuracy profile (AP)* of a calibrated soft classifier $\hat{S}$ for a group g, denoted by $\hat{\mathcal{P}}_g$, is the PMF of $\hat{S}(X_g)$. That is, for $s \in [0,1]$, $\hat{\mathcal{P}}_g(s) = \Pr[\hat{S}(X_g) = s]$.

Abusing notation, we denote by $\hat{\mathcal{P}}$ the collection $\{\hat{\mathcal{P}}_g\}_{g\in\mathcal{G}}$, and call it the *AP of* $\hat{S}$. We denote by $\mathsf{Supp}(\hat{\mathcal{P}}_g)$ the support of the AP $\hat{\mathcal{P}}_g$, namely the set $\mathsf{Supp}(\hat{\mathcal{P}}) = \{s : \exists x \in \mathcal{X}_g, \exists r \text{ s.t. } \hat{S}(x,r) = s\} \subseteq [0,1]$.

An accuracy profile is a distribution of scores for a calibrated classifier $\hat{S}$. Because $\hat{S}$ is calibrated, the AP conveys information about the performance of $\hat{S}$, and is constrained by properties of the underlying distribution on X. For example, the AP's expectation is exactly the base rate for the population:

Proposition 3.2.2.2. *For any groupwise calibrated soft classifier $\hat{S}$, for all groups $g \in \mathcal{G}$: $\mathsf{BR}_g = \mathbb{E}[\hat{S}(X_g)]$.*

Proof of Proposition 3.2.2.2.

$$\begin{aligned}
\mathsf{BR}_g &= \Pr[Y(X_g) = 1] \\
&= \sum_{s \in \mathsf{Supp}(\hat{\mathcal{P}}_g)} \Pr[Y(X_g) = 1 \mid \hat{S}(X_g) = s] \Pr[\hat{S}(X_g) = s] \\
&= \sum_{s \in \mathsf{Supp}(\hat{\mathcal{P}}_g)} s \Pr[\hat{S}(X_g) = s] \\
&= \mathbb{E}[\hat{S}(X_g)]
\end{aligned}$$

where the third line follows from the definition of a calibrated classifier (Definition 3.2.1.2). □

Accuracy profiles also provide useful geometric intuition for reasoning about the effects of post-processing calibrated scores. We elaborate on this in Section 3.3.1 (see Figure 3·2).

3.2.3 Group fairness measures

Several well-studied measures of statistical "fairness" (e.g., [81,162,164,189,200,248]) look at how the following key performance measures of a classifier differ across groups. The *false positive rate* (FPR) of a hard classifier $\hat{Y}$ for a group g is the rate at which $\hat{Y}$ gives a positive classification among instances $x \in \mathcal{X}_g$ with true type 0. The *false negative rate* (FNR) is defined analogously for predicted negative instances with true type 1. *Positive predictive value* (PPV) and *negative predictive value* (NPV) track the rate of mistakes within instances that share a predicted type. Informally, positive predictive value captures how much meaning can be given to a predicted 1, and negative predictive value is similar for predicted 0. We now define these statistics formally.

Definition 3.2.3.1. Given a hard classifier $\hat{Y}$ and a group g, we define
the *false positive rate* of $\hat{Y}$ for g: $\mathsf{FPR}_{\hat{Y},g} = \Pr[\hat{Y}(X_g) = 1 \mid Y(X_g) = 0]$;
the *false negative rate* of $\hat{Y}$ for g: $\mathsf{FNR}_{\hat{Y},g} = \Pr[\hat{Y}(X_g) = 0 \mid Y(X_g) = 1]$;
the *positive predictive value* of $\hat{Y}$ for g: $\mathsf{PPV}_{\hat{Y},g} = \Pr[Y(X_g) = 1 \mid \hat{Y}(X_g) = 1]$;
the *negative predictive value* of $\hat{Y}$ for g: $\mathsf{NPV}_{\hat{Y},g} = \Pr[Y(X_g) = 0 \mid \hat{Y}(X_g) = 0]$.

The probability statements in the definitions above reflect two sources of randomness: the sampling of X_g from the group g and any random choices made by the classifier $\hat{Y}$.

Among previous works, some [162,200] focus on equalizing only one or both of the false positive rates and false negative rates across groups, called *balance* for the negative and positive classes, respectively. Equalizing positive and negative predictive value across groups is often combined into one condition called *predictive parity* [81]. We split the value out to be a separate condition for the positive and negative predictive classes. Predictive parity appears to be a hard-classifier analogue of calibration: both can be interpreted as saying that the output of the classifier (hard or soft) contains all the information contained in group membership. Our results highlight that the relationship between these notions is more subtle than it first appears; see Section 3.3 for further discussion.

3.3 The limits of post-processing

Suppose throughout this section that $\hat{S}$ is a groupwise calibrated soft classifier. Our goal in this section is to make binary predictions based on $\hat{S}(x)$ — and possibly the group $G(x)$ — subject to equalizing PPV and/or NPV among groups. That is, we wish to make a prediction using a hard post-processor $\hat{D}$ such that $\hat{Y} = \hat{D} \circ \hat{S}$ equalizes PPV

and/or NPV among groups. We chose to concentrate first on (the difficulties with) equalizing PPV and NPV rather than FPR and FNR due to the conceptual similarity of PPV and NPV to calibration. Also, the case of equalizing false positive rates with thresholds is addressed in [92].

3.3.1 Fairness conditions for post-processors

We begin by making a simple observation about post-processing that provides some geometric intuition for the rest of this section. Just as in Proposition 3.2.2.2, we can express $\mathsf{PPV}_{\hat{Y},g}$ and $\mathsf{NPV}_{\hat{Y},g}$ succinctly in terms of conditional expectations over the AP $\hat{\mathcal{P}}_g$.

Proposition 3.3.1.1. *Let $\hat{Y} = \hat{D} \circ \hat{S}$ be a hard classifier that is non-trivial for all $g \in \mathcal{G}$ where $\hat{S}$ is groupwise calibrated with respect to $\mathcal{G}$. For any $g \in \mathcal{G}$ we have:*

$$\mathsf{PPV}_{\hat{Y},g} = \mathbb{E}[\hat{S}(X_g) \mid \hat{Y}(X_g) = 1]$$
$$\mathsf{NPV}_{\hat{Y},g} = 1 - \mathbb{E}[\hat{S}(X_g) \mid \hat{Y}(X_g) = 0]$$

Proof of Proposition 3.3.1.1. We first observe that the output of a post-processor is conditionally independent of the true type, conditioned on the output of the soft classifier it is post-processing and the group membership:

Fact 3.3.1.2. *Consider any randomized function $\hat{D} : [0,1] \times \mathcal{G} \to \{0,1\}$. Since $\hat{D}$ is a randomized function with inputs $s \in [0,1]$ and $g \in \mathcal{G}$, we have that*

$$(\hat{D}(\hat{S}(X), G(X)) \perp Y(X)) \mid (\hat{S}(X), G(X)) \tag{3.2}$$

or in other words that $\hat{D}(\hat{S}(X), G(X))$ is conditionally independent of the true type $Y(X)$, since fixing the inputs to $\hat{D}$ makes its output purely a function of its random string.

Now recall that $\mathsf{PPV}_{\hat{Y},g}$ and $\mathsf{NPV}_{\hat{Y},g}$ are well-defined for all groups because $\hat{Y}$ is non-trivial on all groups. We then have

$$\begin{aligned}
\mathsf{PPV}_{\hat{Y},g} &= \Pr[Y(X_g) = 1 \mid \hat{Y}(X_g) = 1] \\
&= \sum_{s \in \mathsf{Supp}(\hat{\mathcal{P}}_g)} \Pr[Y(X_g) = 1, \hat{S}(X_g) = s \mid \hat{D}(\hat{S}(X_g), g) = 1] \\
&= \sum_{s \in \mathsf{Supp}(\hat{\mathcal{P}}_g)} \Pr[Y(X_g) = 1 \mid \hat{S}(X_g) = s, \hat{D}(\hat{S}(X_g), g)] \\
&\qquad\qquad\qquad \cdot \Pr[\hat{S}(X_g) = s | \hat{D}(\hat{S}(X_g), g) = 1] \\
&= \sum_{s \in \mathsf{Supp}(\hat{\mathcal{P}}_g)} s \Pr[\hat{S}(X_g) = s | \hat{D}(\hat{S}(X_g), g) = 1] \\
&= \mathbb{E}[\hat{S}(X_g) \mid \hat{Y}(X_g) = 1]
\end{aligned}$$

where the fourth line follows from the fact that the group g is fixed within $\mathcal{X}_g$, which lets us apply Fact 3.3.1.2, and the fact that $\hat{S}$ is calibrated on g. Similar simplifications give us that

$$\begin{aligned}
\mathsf{NPV}_{\hat{Y},g} &= \Pr[Y(X_g) = 0 \mid \hat{D}(\hat{S}(X_g), g) = 0] \\
&= 1 - \Pr[Y(X_g) = 1 \mid \hat{D}(\hat{S}(X_g), g) = 0] \\
&= 1 - \mathbb{E}[\hat{S}(X_g) \mid \hat{D}(\hat{S}(X_g), g) = 0]
\end{aligned}$$

□

Using Proposition 3.3.1.1, we can geometrically see how certain post-processing decision rules will interact with the AP for a group g. For example, using a threshold, the expected true positives, true negatives, false positives, and false negatives can be estimated, as shown in Figure 3·2.

Proposition 3.3.1.3 below gives a characterizations of the false positive and false negative rates in a manner analogous to how Proposition 3.3.1.1 describes PPV and NPV:

Proposition 3.3.1.3. *Let $\hat{Y}$ and $\hat{S}$ be hard and soft classifiers as in Proposition*

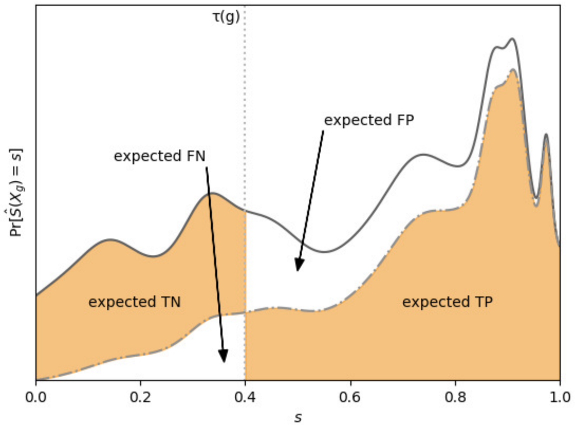

Figure 3·2: *Accuracy Profiles (APs, definition 3.2.2.1) yield useful geometric intuitions, which come from the calibration property (definition 3.2.1.1). With a threshold, the expected PPV, NPV, FPR, and FNR can be seen visually.*

3.3.1.1. Then for any $g \in \mathcal{G}$,

$$\mathit{FPR}_{\hat{Y},g} = \Pr[\hat{Y}(X_g) = 1] \cdot \frac{1 - \mathbb{E}[\hat{S}(X_g) \mid \hat{Y}(X_g) = 1]}{1 - \mathbb{E}[\hat{S}(X_g)]}$$
$$\mathit{FNR}_{\hat{Y},g} = \Pr[\hat{Y}(X_g) = 0] \cdot \frac{\mathbb{E}[\hat{S}(X_g) \mid \hat{Y}(X_g) = 0]}{\mathbb{E}[\hat{S}(X_g)]}$$

Assume that $Pr[Y(X_g) = 1] > 0$ and $Pr[Y(X_g) = 0] > 0$ (that is, assume $0 < \mathit{BR}_g < 1$) so that FPR and FNR are well-defined.

Proof of Proposition 3.3.1.3. We give the proof for FPR, and the proof for FNR is similar. By applying Bayes' rule, we can write

$$\begin{aligned} \mathsf{FPR}_{\hat{Y},g} &= \Pr[\hat{Y}(X_g) = 1 \mid Y(X_g) = 0] \\ &= \Pr[Y(X_g) = 0 \mid \hat{Y}(X_g) = 1] \cdot \frac{\Pr[\hat{Y}(X_g) = 1]}{\Pr[Y(X_g) = 0]} \end{aligned} \tag{3.3}$$

Noting that $\Pr[Y(X_g) = 0 \mid \hat{Y}(X_g) = 1] = 1 - \mathsf{PPV}_{\hat{Y},g}$, we can apply Proposition 3.3.1.1 and rearrange to write the RHS of Equation 3.3 as follows.

$$\text{RHS of (3.3)} = \Pr[\hat{Y}(X_g) = 1] \cdot \frac{1 - \mathbb{E}[\hat{S}(X_g) \mid \hat{Y}(X_g) = 1]}{\Pr[Y(X_g) = 0]} \tag{3.4}$$

We note that $\Pr[Y(X_g) = 0] = 1 - \mathbb{E}[\hat{S}(X_g)]$ (Proposition 3.2.2.2). Substituting this in to the RHS of Equation 3.4, we conclude the result. □

3.3.2 General impossibility of equalizing Positive and Negative Predictive Value

It is not always possible to directly post-process a soft groupwise calibrated classifier into a hard one with equalized PPV (or NPV) for all groups, as we demonstrate by counterexample in Proposition 3.3.2.1. Before proceeding, we note that our counterexample is somewhat contrived—in particular, the AP induced by the soft classifier $\hat{S}$ in the proof of Proposition 3.3.2.1 takes only one value on each group. When the AP of $\hat{S}$ is more nicely structured on each group (which we will make explicit in Definition 3.3.3.1), we will see that there are general methods to equalize PPV (or NPV).

Proposition 3.3.2.1. *Fix two disjoint groups g_1 and g_2 with respective base rates BR_1 and BR_2 such that $\mathsf{BR}_1 \neq \mathsf{BR}_2$. Then there exists a soft classifier $\hat{S}$ that is groupwise calibrated, but for which there is no post-processor $\hat{D} : [0,1] \times \mathcal{G} \to \{0,1\}$ such that $\hat{D} \circ \hat{S}$ equalizes PPV, unless $\Pr[\hat{D}(\mathsf{BR}_i, g_i) = 1] = 0$ for $i = 1$ or 2.*

Proof of Proposition 3.3.2.1. Consider the classifier $\hat{S}$ such that $\hat{S}(x) = \mathsf{BR}_1$ if $x \in g_1$ and $\hat{S}(x) = \mathsf{BR}_2$ if $x \in g_2$. This classifier is trivially groupwise calibrated. Since $\Pr[\hat{D}(\mathsf{BR}_i, g_i) = 1] > 0$ for $i = 1$ and 2, we conclude that $\mathsf{PPV}_{\hat{Y},g_i}$ is well-defined for g_1 and g_2. The proof now follows from the characterization of PPV in Proposition 3.3.1.1. This is because $\mathsf{PPV}_{\hat{Y},g_i}$ is equal to the expectation of $\hat{S}(X)$ where X is drawn from a distribution with support contained in g_i, and hence it is equal to BR_i, and $\mathsf{BR}_1 \neq \mathsf{BR}_2$. □

The analogous statement regarding impossibility of equalizing NPV is formulated as Proposition 3.4.1.1 in §3.4.1.

3.3.3 A niceness condition for APs

We now give a non-degeneracy condition on APs motivated by the impossibility result for post-processing given by Proposition 3.3.2.1.

Definition 3.3.3.1 (Niceness of APs)**.** Let $\mathcal{G}$ be a set of groups. A distribution on calibrated scores $\hat{\mathcal{P}}$ is *nice* if $\mathsf{Supp}(\hat{\mathcal{P}}_g)$ is the same for all $g \in \mathcal{G}$.

Note that this condition rules out the counterexample given by Proposition 3.3.2.1, since the APs in the counterexample had different (in fact, disjoint) supports for different groups. Hence, we can hope to successfully post-process soft classifiers with nice APs.

3.3.4 Equalizing Positive and Negative Predictive Value by thresholding

We pay special attention to thresholds because they are simple to understand and therefore very widely used. We use one slight modification to deterministic thresholds that adds an element of randomness: if a score is *at* the threshold, we randomly determine which side of the threshold it falls on, according to a distribution defined below.

Definition 3.3.4.1 (Threshold Post-Processor)**.** A threshold post-processor $\hat{D}_{(\tau,\rho)} : [0,1] \times \mathcal{G} \to \{1, 0\}$ is a function from a score $s \in [0,1]$ and a group $g \in \mathcal{G}$, parameterized by τ and ρ. The threshold parameter $\tau : \mathcal{G} \to [0,1]$ specifies the threshold for the group g, and $\rho : \mathcal{G} \to [0,1]$ is the probability of returning 1 when the input score s is on the threshold $\tau(g)$. It returns the following outputs:

$$\hat{D}_{(\tau,\rho)}(s,g) = \begin{cases} 1 & s > \tau(g) \\ 0 & s < \tau(g) \\ 1 \text{ w.p. } \rho(g) \text{ else } 0 & s = \tau(g) \end{cases}$$

In the setting of an infinite number of scores and a continuous domain (i.e. scores are represented by a probability density function instead of a probability mass function), we can use purely deterministic threshold functions in which $\rho \equiv 1$, and achieve very similar results for the rest of this section.

If both $\tau(g)$ and $\rho(g)$ do not vary across groups $g \in \mathcal{G}$, then the post-processor is the same across groups. In this case, we will call the post-processor a *group blind* threshold post-processor, and will overload τ and ρ to be constants.

We now study the effectiveness of thresholds for post-processing soft classifiers with nice APs. The main takeaways are:

1. If the APs are nice, then threshold post-processors can equalize PPV (Propositions 3.3.4.2 and 3.3.4.5).

2. However, group blind threshold post-processors are rather limited in their ability to equalize PPV (Proposition 3.3.4.3).

3. Furthermore, equalizing PPV with thresholds (group blind or otherwise) may have undesirable social consequences (Example 3.3.4.8).

4. Thresholds cannot always equalize PPV and NPV simultaneously, even for nice APs (Proposition 3.3.4.9).

Results 1-3 also apply to NPV (see Proposition 3.4.1.3).

Group Blind Thresholds

We begin by classifying which group blind threshold post-processors can equalize PPV across all groups (Propositions 3.3.4.2 and 3.3.4.3). By symmetry, our arguments give a similar characterization for equalizing NPV.

Proposition 3.3.4.2. *For every nice groupwise calibrated soft classifier $\hat{S}$ and for every group blind threshold post-processor $\hat{D}_{(\tau,\rho)}$ such that $\tau(g) = \max(\mathsf{Supp}(\hat{\mathcal{P}}_g))$ for all g, then the composed classifier $\hat{Y} = \hat{D}_{(\tau,\rho)} \circ \hat{S}$ equalizes PPV across all groups for which $\hat{Y}$ is non-trivial.*

The existence of the threshold post-processors in Proposition 3.3.4.2 follows from the assumed finiteness of the range of the soft classifier. In the case where the range of the soft classifier is infinite, such post-processors may not exist.

Proof of Proposition 3.3.4.2. Any of the given post-processors only ever maps the largest score in the support of $\hat{\mathcal{P}}_g$ to 1, for all groups g. Hence, PPV_g is exactly the largest score in $\mathsf{Supp}(\hat{\mathcal{P}}_g)$. By the assumption that $\hat{\mathcal{P}}$ is nice, $\mathsf{Supp}(\hat{\mathcal{P}}_g)$ is the same for all groups g, and hence the PPV is equalized across groups. □

We prove the analogous statement for NPV in Proposition 3.4.1.2 in the §3.4.1. We proceed to show that the post-processors described in Proposition 3.3.4.2 are the *only* non-trivial, group blind post-processors that equalize PPV across groups in general, as we prove in Proposition 3.3.4.3.

Proposition 3.3.4.3. *There exists a groupwise-calibrated soft classifier with a nice AP for which no non-trivial group blind threshold post-processor, other than the ones in Proposition 3.3.4.2, can equalize PPV across groups.*

At a high level, the proof of Proposition 3.3.4.3 works as follows: We can make the AP on one group uniform, and the AP of another group strictly increasing. Then, threshold post-processors naturally favor the latter group, as the AP for that group gives more weight to higher scores than lower ones when compared to the former AP. Our characterization of PPV (Proposition 3.3.1.1) features prominently in the proof.

In preparation for proving Proposition 3.3.4.3, we first prove the following lemma:

Lemma 3.3.4.4. *Let $g_1, g_2 \in \mathcal{G}$ be two different groups, and fix a group blind threshold post-processor $\hat{D}_{(\tau,\rho)}$. Let $\hat{\mathcal{P}}_{g_1,(\tau,\rho)}$ be the expected conditional AP on scores $\geq \tau$ that results from starting with the AP $\hat{\mathcal{P}}_{g_1}$ over scores in group g_1 and conditioning on the scores that $\hat{D}_{(\tau,\rho)}$ sends to 1, and similarly let $\hat{\mathcal{P}}_{g_2,(\tau,\rho)}$ denote the same type of conditional AP when starting with the $\hat{\mathcal{P}}_{g_2}$ over scores in group g_2.*
If $\hat{\mathcal{P}}_{g_2,(\tau,\rho)}$ strictly stochastically dominates $\hat{\mathcal{P}}_{g_1,(\tau,\rho)}$, then

$$\mathsf{PPV}_{\hat{D}_{(\tau,\rho)}\circ\hat{S},g_1} < \mathsf{PPV}_{\hat{D}_{(\tau,\rho)}\circ\hat{S},g_2}$$

Proof. We use the characterization of PPV given in Proposition 3.3.1.1, for the special case where the post-processor thresholds as described above. We can write the PPV

for group g_1 as follows:

$$\begin{aligned}\mathsf{PPV}_{\hat{D}_{(\tau,\rho)}\circ\hat{S},g_1} &= \mathop{\mathbb{E}}_{X_1\sim\mathcal{X}_{g_1}}[\hat{S}(X_1) \mid \hat{D}_{(\tau,\rho)}\circ\hat{S}(X_1) = 1] \\ &= \mathop{\mathbb{E}}_{s\sim\hat{\mathcal{P}}_{g_1,(\tau,\rho)}}[s] \qquad (3.5)\end{aligned}$$

where the second line follows from the definition of $\hat{\mathcal{P}}_{g_1,(\tau,\rho)}$.

Similarly, we have that

$$\mathsf{PPV}_{\hat{D}_{(\tau,\rho)}\circ\hat{S},g_2} = \mathop{\mathbb{E}}_{s\sim\hat{\mathcal{P}}_{g_2,(\tau,\rho)}}[s] \qquad (3.6)$$

Since $\hat{\mathcal{P}}_{g_2,(\tau,\rho)}$ stochastically dominates $\hat{\mathcal{P}}_{g_1,(\tau,\rho)}$, the expectation on the RHS of Equation 3.6 is larger than the expectation on the RHS of Equation 3.5, yielding the result. □

Proof of Proposition 3.3.4.3. Fix two groups g_1 and g_2 and a finite set of points $S \subset [0,1]$ such that the PMFs of the soft classifier $\hat{S}$ on g_1 and g_2 have support equal to S - that is, $\mathsf{Supp}(\hat{\mathcal{P}}_{g_1}) = \mathsf{Supp}(\hat{\mathcal{P}}_{g_2}) = S$.

Let the PMFs of the soft classifier $\hat{S}$ on these two groups respectively be given by $\hat{\mathcal{P}}_{g_1}(s) = 1/|\mathsf{Supp}(\hat{\mathcal{P}}_{g_1})|$ and $\hat{\mathcal{P}}_{g_2}(s) \propto s$ for all $s \in S$, where $\hat{\mathcal{P}}_{g_2}$ is normalized with a constant such that it sums to 1. Fix a group blind threshold post-processor $\hat{D}_{(\tau,\rho)}$ that is not one of the ones mentioned in Proposition 3.3.4.2. Since $\hat{S}_{(\tau,\rho)}$ is group blind, its threshold function is a constant which we name τ.

Let $\hat{\mathcal{P}}_{g_1,(\tau,\rho)}$ be the expected conditional AP on scores $\geq \tau$ that results from starting with the AP $\hat{\mathcal{P}}_{g_1}$ over scores in group g_1 and conditioning on the scores that $\hat{D}_{(\tau,\rho)}$ sends to 1. We can get this conditional PMF by removing scores $s < \tau$, multiplying $\hat{\mathcal{P}}_{g_1}(\tau)$ by ρ, and re-normalizing the remaining values to get a distribution. Let $\hat{\mathcal{P}}_{g_2,(\tau,\rho)}$ be defined similarly.

We claim that $\hat{\mathcal{P}}_{g_2,(\tau,\rho)}$ strictly stochastically dominates $\hat{\mathcal{P}}_{g_1,(\tau,\rho)}$, which allows us to invoke Lemma 3.3.4.4 to conclude that the PPV on the two groups are unequal. We now show that $\hat{\mathcal{P}}_{g_2,(\tau,\rho)}$ strictly stochastically dominates $\hat{\mathcal{P}}_{g_1,(\tau,\rho)}$. This is clearly true by design if $\rho = 0$ or 1: in this case, the post-processor is simply a deterministic threshold function, and we know by design that $\hat{\mathcal{P}}_{g_1,(\tau,\rho)}$ is uniform while $\hat{\mathcal{P}}_{g_2,(\tau,\rho)}$ is a strictly increasing function. If $\rho = r$ for some $r \in (0,1)$, then we can write $\hat{\mathcal{P}}_{g_1,(\tau,\rho)}$ as a convex combination of $\hat{\mathcal{P}}_{g_1,\tau,0}$ and $\hat{\mathcal{P}}_{g_1,\tau,1}$ (with weight r on the distribution where $\rho = 1$, and weight $1-r$ on the distribution where $\rho = 0$). We can write $\hat{\mathcal{P}}_{g_2,(\tau,\rho)}$ as the same convex combination of the conditional distributions over g_2 where $\rho = 0$ and $\rho = 1$. Since we already established stochastic domination for the cases where $\rho = 0$ and $\rho = 1$, this establishes stochastic domination for the case where $\rho \in (0, 1)$. □

We achieve the same result for NPV in Proposition 3.4.1.3. In the setting where the range of the soft classifier is infinite and continuous, we show in Proposition 3.4.2.1 that a similar negative result holds, but without the existence of the classifiers in Proposition 3.3.4.2.

Propositions 3.3.4.2 and 3.3.4.3 demonstrate the limits of group blind thresholds on calibrated scores. Though this method of post-processing has social appeal, it does not actually preserve the fairness properties that one would expect. In the next section we repeat our analysis but relax our group blindness requirement.

Group-Aware Thresholds

If we allow the different groups to have different thresholds, then we grant ourselves more degrees of freedom to be able to satisfy binary fairness constraints. In particular, we can equalize PPV across groups in a more meaningful way than done in Proposition 3.3.4.2.

Recall that the group blind threshold post-processors in Proposition 3.3.4.2 are the only group blind threshold post-processors that work on certain nice APs (shown in Proposition 3.3.4.3). However, these post-processors have the property that the only score they map to 1 is the largest score in the support, which can be undesirable for many applications.

In particular, all classifiers in Proposition 3.3.4.3 make the PPV on each group g_i equal to the maximum score in the support of $\hat{\mathcal{P}}_{g_i}$. However, the (not necessarily group blind) threshold post-processors in Proposition 3.3.4.5 below can make the PPV on each group equal to any fixed value between the maximum base rate of g_i and the maximum score in $\mathsf{Supp}(\hat{\mathcal{P}}_{g_i})$.

Proposition 3.3.4.5. *Let $\mathcal{G}$ be a set of groups. For any soft classifier $\hat{S}$ with a nice AP $\hat{\mathcal{P}}$ such that $\hat{S}$ is groupwise calibrated over $\mathcal{G}$ and $|\mathsf{Supp}(\hat{\mathcal{P}}_g)| \geq 2$ for all $g \in \mathcal{G}$, then there exists a non group blind, non-trivial threshold post-processor $\hat{D}_{(\tau,\rho)}$ that is not one of the ones from Proposition 3.3.4.2 such that the hard classifier $\hat{Y} = \hat{D}_{(\tau,\rho)} \circ \hat{S}$ equalizes PPV across $\mathcal{G}$.*

This holds even if we require that the PPV of all the groups is equal to an arbitrary value in $(\max_i \mathsf{BR}_{g_i}, s_{max})$, where $\max_i \mathsf{BR}_{g_i}$ is the maximum base rate among the groups $g_i \in \mathcal{G}$ and s_{max} is the maximum score in the support of $\hat{\mathcal{P}}_{g_i}$. For the case where the support of $\hat{\mathcal{P}}_{g_i}$ is infinite, s_{max} is instead the supremum of scores.

Moreover, since this post-processor is not group blind, it is not one of the post-processors described in Proposition 3.3.4.2.

In preparation for proving Proposition 3.3.4.5, we first prove the following two claims:

Claim 3.3.4.6 (Monotonicity of PPV and NPV)**.** *Fix a soft classifier $\hat{S}$ and corresponding AP $\hat{\mathcal{P}}$, as well as a group g. Fix group blind threshold post-processors $\hat{D}_{\tau_1,\rho_1}$ and $\hat{D}_{\tau_2,\rho_2}$ such that either $\tau_1 < \tau_2$ or both $\tau_1 = \tau_2$ and $\rho_1 \geq \rho_2$. Then:*
(a) $\mathsf{PPV}_{\hat{D}_{\tau_1,\rho_1} \circ \hat{S}, g} \leq \mathsf{PPV}_{\hat{D}_{\tau_2,\rho_2} \circ \hat{S}, g}$
(b) $\mathsf{NPV}_{\hat{D}_{\tau_1,\rho_1} \circ \hat{S}, g} \leq \mathsf{NPV}_{\hat{D}_{\tau_2,\rho_2} \circ \hat{S}, g}$

Proof. We show conclusion (a); conclusion (b) is shown analogously. Define $\hat{\mathcal{P}}_{g,\tau_1,\rho_1}$ to be the conditional PMF on scores $\geq \tau_1$ that results from starting with the AP $\hat{\mathcal{P}}_g$ over scores in group g and conditioning on the scores that $\hat{D}_{\tau_1,\rho_1}$ sends to 1, and let $\hat{\mathcal{P}}_{g,\tau_2,\rho_2}$ be defined similarly for the threshold post-processor $\hat{D}_{\tau_2,\rho_2}$.

We claim that $\hat{\mathcal{P}}_{g,\tau_2,\rho_2}$ stochastically dominates $\hat{\mathcal{P}}_{g,\tau_1,\rho_1}$, which yields the desired result by the characterization of PPV given in Proposition 3.3.1.1 and more explicitly written in Equations 3.5 and 3.6. □

Claim 3.3.4.7 (Continuity of PPV)**.** *Fix a soft classifier $\hat{S}$ and corresponding AP $\hat{\mathcal{P}}$, as well as a group g. Suppose we have two post-processing algorithms, $\hat{D}_1$ and $\hat{D}_2$. Let $\hat{\mathcal{P}}_{g,\hat{D}_1}$ be the expected conditional AP that results from starting with the AP $\hat{\mathcal{P}}_g$ over scores in group g and conditioning on the scores that $\hat{D}_1$ sends to 1, and define $\hat{\mathcal{P}}_{g,\hat{D}_2}$ similarly. If $d_{TV}(\hat{\mathcal{P}}_{g,\hat{D}_1}, \hat{\mathcal{P}}_{g,\hat{D}_2}) < \epsilon$, then $|\mathsf{PPV}_{g,\hat{D}_1 \circ \hat{S}} - \mathsf{PPV}_{g,\hat{D}_2 \circ \hat{S}}| < O(\epsilon)$. Or in words, if the distance between the conditional APs is small, then the difference in PPV is small.*

Proof. Recall the characterization of PPV given in Proposition 3.3.1.1 (and more explicitly written in Equation 3.5). This tells us that the PPV of group g for the classifier $\hat{D}_1 \circ \hat{S}$ is exactly the expectation of a random variable distributed according to $\hat{\mathcal{P}}_{g,\hat{D}_1}$. Similarly, the PPV of group g for the classifier $\hat{D}_2 \circ \hat{S}$ is the expectation of a r.v. distributed according to $\hat{\mathcal{P}}_{g,\hat{D}_2}$. Since both $\hat{\mathcal{P}}_{g,\hat{D}_1}$ and $\hat{\mathcal{P}}_{g,\hat{D}_2}$ have support bounded between 0 and 1, their expectations can differ by at most ϵ, from which the claim follows. For completeness, we prove this below.

Suppose wlog that $\hat{\mathcal{P}}_{g,\hat{D}_1}$ has the larger expectation. Let $S = \{s \in \mathsf{Supp}(\hat{\mathcal{P}}_g) : \hat{\mathcal{P}}_{g,\hat{D}_1}(s) > \hat{\mathcal{P}}_{g,\hat{D}_2}(s)\}$. Then:

$$\begin{aligned}
\mathsf{PPV}_{g,\hat{D}_1 \circ \hat{S}} &= \sum_{s \in \mathsf{Supp}(\hat{\mathcal{P}}_g)} s\hat{\mathcal{P}}_{g,\hat{D}_1}(s) \\
&= \mathsf{PPV}_{g,\hat{D}_2 \circ \hat{S}} + \sum_{s \in S} s(\hat{\mathcal{P}}_{g,\hat{D}_1}(s) - \hat{\mathcal{P}}_{g,\hat{D}_2}(s)) \\
&< \mathsf{PPV}_{g,\hat{D}_2 \circ \hat{S}} + \epsilon
\end{aligned}$$

where in the second line we use the fact that $\mathsf{PPV}_{g,\hat{D}_2 \circ \hat{S}}$ is the expectation of $\hat{\mathcal{P}}_{g,\hat{D}_2}$, and in the last line we use the fact that $s \in [0,1]$ and that the TV-distance between the two distributions is less than ϵ. □

Now we are ready to prove Proposition 3.3.4.5.

Proof of Proposition 3.3.4.5. Fix a soft classifier $\hat{S}$ with a nice AP $\hat{\mathcal{P}}$ that is groupwise calibrated over $g_1, \ldots, g_n$, and fix a desired value $v \in (\max_i \mathsf{BR}_{g_i}, s_{max})$. We will show that we can design a threshold post-processor (τ, ρ) such that $\mathsf{PPV}_{g,\hat{D}_{(\tau,\rho)} \circ \hat{S}} = v$ for all groups g.

Fix an arbitrary group g_j. We proceed via a continuity argument to show that we can tune the threshold on g_j to achieve PPV equal to v. The maximum possible value for $\mathsf{PPV}_{g_j,\hat{D}_{(\tau,\rho)}}$ is s_{max} (achieved when $\tau = s_{max}$, by Claim 3.3.4.6), where s_{max} is the largest score in the support, as defined in the proposition statement. We ignore the trivial post-processor that never maps anything to 1, and hence leaves the PPV undefined.

Furthermore, note that, for any group, a lower bound on the PPV of a hard classifier on that group is the base rate of the group, where the lower bound is matched by the trivial post-processor that sends every score to 1. This follows from Claim 3.3.4.6.

We now claim that there is a setting of $\tau(g_j)$ and $\rho(g_j)$ that achieves $\mathsf{PPV}_{g,\hat{D}_{(\tau,\rho)}\circ\hat{S}} = v$. We accomplish this by showing that there is a way to change $(\tau(g_j), \rho(g_j))$ such that the PPV decreases continuously. We first show:

Now, consider the following way to change $(\tau(g_j), \rho(g_j))$. Fix $\epsilon > 0$, and an initial setting for $(\tau(g_j), \rho(g_j))$ s.t. $\tau(g_j)$ is not the smallest item in the support or $\rho(g_j) > \epsilon$. Reduce $\rho(g_j)$ by ϵ, wrapping around on the interval $(0, 1]$ and decreasing $\tau(g_j)$ to the next largest item in the support when this would otherwise make $\rho(g_j)$ negative.

This very minor transformation to the threshold changes the AP conditional on outputting 1 very slightly - so slightly that the TV distance between the old conditional AP and the new AP is at most some ϵ' which is a function of ϵ. This lets us apply Claim 3.3.4.7 to show that the PPV changes by at most a function of ϵ. So as we take ϵ going towards 0, this shows that the PPV changes by an amount going towards 0. This establishes that the PPV changes "continuously" with respect to this deforming procedure.

By Claim 3.3.4.6, we have that the above deforming procedure can only decrease the PPV. Therefore, we can continuously decrease the PPV, starting from s_{max}, by continuously deforming the threshold post-processor with the method above. Note that $s_{\max} > v > \max_i \mathsf{BR}_{g_i} \geq \mathsf{BR}_{g_j}$. By the Intermediate Value Theorem, there must be a setting of $(\tau(g_j), \rho(g_j))$ such that $\mathsf{PPV}_{g_j,\hat{D}_{(\tau,\rho)}\circ\hat{S}} = v$.

□

We assert the analogous statement for the case of NPV in Claim 3.4.1.5. The corresponding statement for the case of soft classifiers with infinite range is asserted in Proposition 3.4.2.2.

The Limits of Thresholding

While Proposition 3.3.4.5 shows that a threshold post-processor can equalize the PPV across n groups, this threshold post-processor can be unsatisfying. Consider an example with two groups g_1 and g_2 in an image classification setting, where group g_2 is found to post more disallowed imagery. Suppose that we have an AP that is decreasing with respect to score on group g_1, and increasing with respect to score on group g_2. This is illustrated in Example 3.3.4.8 and Figure 3·3. This means that a group blind threshold post-processor yields larger PPV on g_2, since large scores are given more weight in g_2. So, to equalize the PPV between the two groups, we will classify more low scores as positive in g_2 than g_1. This effectively means that our threshold on group g_2 is *more lenient* than our threshold on g_1, which seems blatantly unfair, since g_2 is the posting more undesirable content in the first place!

Example 3.3.4.8 (Socially Unsatisfying Example)**.** Fix groups g_1 and g_2, and we fix the AP of the soft classifier $\hat{S}$ as follows. Let $\mathsf{Supp}(\hat{\mathcal{P}}_{g_1}) = \mathsf{Supp}(\hat{\mathcal{P}}_{g_2})$, let $\hat{\mathcal{P}}_{g_1}(s) \propto a - s$ for appropriately selected constant $a > 0$ and let $\hat{\mathcal{P}}_{g_2}(s) \propto s$. Group g_2 has a higher base rate and may have social advantages over group g_1.

Let $\hat{D}_{(\tau,\rho)}$ be a non-trivial post-processor. If $\hat{D}_{(\tau,\rho)}$ were group blind, then by Lemma 3.3.4.4, since $\hat{\mathcal{P}}_{g_2,\hat{D}_{(\tau,\rho)}}$ stochastically dominates $\hat{\mathcal{P}}_{g_1,\hat{D}_{(\tau,\rho)}}$, the PPV on g_2 must be larger than the PPV of g_1.

To equalize PPV, by Claim 3.3.4.6, we must have either $\tau(g_2) < \tau(g_1)$, or $\tau(g_2) = \tau(g_1)$ and $\rho(g_2) < \rho(g_1)$. Group g_1 is now held to a *higher* standard than the group that posts more undesirable content (g_2) in order to maintain equality of PPV. Figure 3·3 illustrates example thresholds that equalize the PPV.

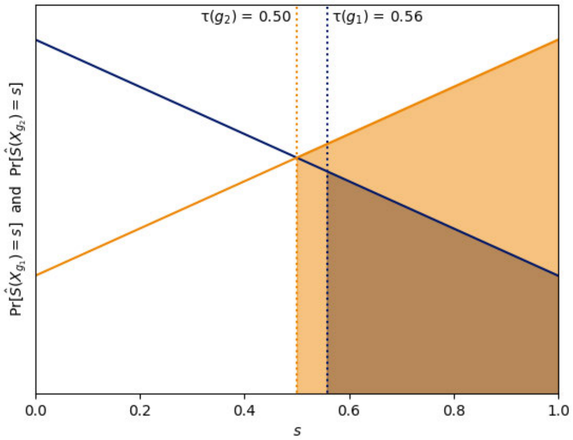

Figure 3·3: *Accompanying Example 3.3.4.8, the* PPV *for both groups is 0.77. However, the threshold for* g_1 *(dark blue) is* higher *than the threshold for* g_2 *(orange), even though* g_2 *is the more problematic group.*

The only property needed for Example 3.3.4.8 is that the AP of one group stochastically dominates the AP of another group. We suspect that this to will occur in many settings. Furthermore, thresholding cannot in general equalize both PPV and NPV simultaneously, even for nice APs and using non-group blind thresholds.

Proposition 3.3.4.9. *Fix groups g_1 and g_2. There exists a soft classifier $\hat{S}$ with a nice AP $\hat{\mathcal{P}}$ such that no threshold post-processor can simultaneously equalize PPV and NPV between groups g_1 and g_2.*

Before proving the statement, we first show that the base rates of g_1 and g_2 can be written as convex combinations of the PPV and $1-\mathsf{NPV}$ on the respective groups:

Claim 3.3.4.10. *Fix any group $g \in \mathcal{G}$, and let the hard classifier $\hat{Y}$ be non-trivial. Then the base rate of g can be written as a convex combination of $\mathsf{PPV}_{g,\hat{Y}}$ and $1-\mathsf{NPV}_{g,\hat{Y}}$*

Proof.

$$\begin{aligned}
\mathsf{BR}_g &= \Pr[Y(X_g)=1] \\
&= \Pr[Y(X_g)=1 \mid \hat{Y}(X_g)=1]\Pr[\hat{Y}(X_g)=1] \\
&\quad + \Pr[Y(X_g)=1 \mid \hat{Y}(X_g)=0]\Pr[\hat{Y}(X_g)=0] \\
&= \mathsf{PPV}_{g,\hat{Y}} \cdot \theta + (1-\mathsf{NPV}_{g,\hat{Y}}) \cdot (1-\theta)
\end{aligned}$$

where $\theta := \Pr[\hat{Y}(X_g)=1]$. □

A simple intuition for the proof of Proposition 3.3.4.9 is as follows. Suppose we have two groups, and the soft classifier is almost perfect on one group - for all but a small fraction of inputs it gives the correct binary score, and gives the remaining inputs score 0.5. On the other group, it is the opposite - almost every input is given score 0.5, and there are only a few inputs given their ground truth score. The corresponding APs of each group have equal supports, and therefore are “nice.” However, it is clear that any threshold post-processor on the first group will have extremely high PPV and NPV, while any threshold post-processor on the second group will have to make a decision on where to round the inputs in the 0.5 bucket, and will correspondingly either have low PPV or NPV.

This example is somewhat unsatisfying, as the AP satisfies niceness by a technicality (for example, it is extremely close to a AP with disjoint supports between groups). The proof generalizes the above example to a case where scores for one group are slightly more "correlated" with the ground truth labels than scores for the other group. §3.4.2 shows the corresponding result when the range of the soft classifier is infinite.

Proof of Proposition 3.3.4.9. Let $\mathsf{Supp}(\hat{\mathcal{P}}_{g_1}) = \mathsf{Supp}(\hat{\mathcal{P}}_{g_2})$. Let $\hat{\mathcal{P}}_{g_1}$ be uniform over scores, and let $\hat{\mathcal{P}}_{g_2}(s) = -a(s-1/2)^2 + b$ for some appropriately chosen constants $a, b \geq 0$ such that $\hat{\mathcal{P}}_{g_2}$ is a valid probability distribution.

We claim that there is no threshold post-processor that can equalize PPV and NPV simultaneously for these two groups. First, note that the base rates for the two groups are equal to 1/2 by design, due to the symmetric nature of the AP on each group. Just like in the proof of Proposition 3.3.4.3, we use the notation $\hat{\mathcal{P}}_{g_1,\geq(\tau,\rho)}$ to denote the conditional AP supported on scores $\geq \tau$ that results from starting with the AP $\hat{\mathcal{P}}_{g_1}$ over all scores in group g_1 and conditioning on the scores that $\hat{D}_{(\tau,\rho)}$ sends to 1, and define $\hat{\mathcal{P}}_{g_2,\geq(\tau,\rho)}$ similarly. Let $\hat{\mathcal{P}}_{g_i,\leq(\tau,\rho)}$ denote the conditional AP starting from group g_i and conditioning on the scores that $\hat{D}_{(\tau,\rho)}$ sends to 0.

We proceed by a case analysis on the location of the threshold for group g_1 - that is, on $\tau(g_1)$.

Case 1: $\tau(g_1) \geq 1/2$. First, suppose additionally that $\hat{D}_{(\tau,\rho)}$ is group blind over g_1, g_2, then $\hat{\mathcal{P}}_{g_2,(\tau,\rho)}$ is strictly stochastically dominated by $\hat{\mathcal{P}}_{g_1,(\tau,\rho)}$ so the PPV is lower on g_2 than g_1 (Lemma 3.3.4.4). Therefore, to equalize the PPV between the two groups, the threshold on group g_2 must be to the "right" of the threshold on g_1 (that is, either $\tau(g_2) > \tau(g_1)$ or $\tau(g_2) = \tau(g_1)$ and $\rho(g_2) < \rho(g_1)$), which follows from Claim 3.3.4.6.

However, we claim that this setting of thresholds makes the NPV on group g_2 lower than the NPV on group g_1. It suffices to show that, for any constant threshold post-processor classifier $\hat{D}_{(\tau,\rho)}$ with $\tau(g_1) = \tau(g_2) \geq 1/2$, the NPV on group g_2 is lower than the NPV on group g_1. This implies that the same statement holds when either $\tau(g_2) > \tau(g_1)$ or $\tau(g_2) = \tau(g_1)$ and $\rho(g_2) < \rho(g_1)$ as well, due to the monotonicity property of NPV (Claim 3.3.4.6). This suffices to prove Proposition 3.3.4.9 for the case when $\tau(g_1) \geq 1/2$: we need to set $\tau(g_2), \rho(g_2)$ such that either $\tau(g_2) > \tau(g_1)$ or

$\tau(g_2) = \tau(g_1)$ and $\rho(g_2) < \rho(g_1)$ in order to equalize PPV, but this leaves the NPV on group g_2 lower than the NPV on group g_1.

Now we proceed to show that the NPV is lower on g_2 when the threshold post-processor is constant. Fix a constant threshold post-processor $\hat{D}_{(\tau,\rho)}$ with $\tau \geq 1/2$. We have already established that this means that $\mathsf{PPV}_{g_2,\hat{D}_{(\tau,\rho)}\circ\hat{S}} < \mathsf{PPV}_{g_1,\hat{D}_{(\tau,\rho)}\circ\hat{S}}$.

Now, since the base rates of g_1 and g_2 are equal, this implies that

$$1 - \mathsf{NPV}_{g_2,\hat{D}_{(\tau,\rho)}\circ\hat{S}} > 1 - \mathsf{NPV}_{g_1,\hat{D}_{(\tau,\rho)}\circ\hat{S}}$$

Rearranging, this shows that $\mathsf{NPV}_{g_2,\hat{D}_{(\tau,\rho)}\circ\hat{S}} < \mathsf{NPV}_{g_1,\hat{D}_{(\tau,\rho)}\circ\hat{S}}$, finishing the proof of this case.

Case 2: $\tau(g_1) < 1/2$. The argument in this case is symmetrical to the previous case, where we switch the roles of PPV and NPV in the argument. We sketch it for the sake of completeness.

Fix a constant threshold post-processor such that $\tau(g_1) = \tau(g_2) < 1/2$. By design, $\hat{\mathcal{P}}_{g_2,\leq(\tau,\rho)}$ strictly stochastically dominates $\hat{\mathcal{P}}_{g_1,\leq(\tau,\rho)}$. Hence, the NPV on group g_2 is smaller than the NPV on group g_1 (this follows from an analogous version of Lemma 3.3.4.4 for NPV instead of PPV). This means that the threshold post-processor cannot be constant to equalize NPVs - it must be moved such that either $\tau(g_2) < \tau(g_1)$ or $\tau(g_2) = \tau(g_1)$ and $\rho(g_2) > \rho(g_1)$ (Claim 3.3.4.6).

However, by the same convex combination argument as in the previous case, we get that any such constant threshold post-processor must make the PPV on g_2 strictly smaller than the PPV on g_1. By the monotonicity of PPV, this means that any threshold post-processor with either $\tau(g_2) < \tau(g_1)$ or $\tau(g_2) = \tau(g_1)$ and $\rho(g_2) > \rho(g_1)$ must also make the PPV on g_2 smaller than the PPV on g_1, finishing the proof. □

3.3.5 Equalizing APs

While thresholding is a conceptually simple approach to post-processing a soft classifier, its power is limited. We now consider a very different approach using soft post-processors to equalize the APs across groups of a soft classifier. The intuition is that if the APs are equal across groups, then any hard post-processor that is *group blind* should result in equal PPV, NPV, FPR, and FNR. We formalize this intuition in Claim 3.3.5.1.

Let $\hat{S}$ be a soft classifier and for each group $g \in \mathcal{G}$, let $\hat{\mathcal{P}}_g$ be the AP of $\hat{S}$ for group g. For a soft post-processor $\hat{D}^{\mathsf{soft}}$, let $\hat{S}' = \hat{D}^{\mathsf{soft}} \circ \hat{S}$ and let $\hat{\mathcal{P}}'_g$ be the corresponding AP for group g.

Our goal is to find a soft post-processor $\hat{D}^{\mathsf{soft}}$ such that $\hat{S}'$ is groupwise calibrated, and $\hat{\mathcal{P}}'_g = \hat{\mathcal{P}}'_{g'}$ for all $g, g' \in \mathcal{G}$. In this section, we describe only one approach to constructing $\hat{D}^{\mathsf{soft}}$ which we call *mass averaging.*

The approach of equalizing APs has a fundamental weakness: if $\hat{\mathcal{P}}'_g = \hat{\mathcal{P}}'_{g'}$ and both are calibrated, then $\mathsf{BR}_g = \mathsf{BR}_{g'}$. This severely limits applicability of this approach. However, this limitation will be removed in Section 3.5.2 by allowing deferrals.

Claim 3.3.5.1. *If the APs are equal for two groups, then* PPV*,* NPV*,* FPR*, and* FNR *are equalized by any hard post-processor* $\hat{D}$ *satisfying group blindness.*

The group blindness requirement in the claim is necessary: consider the (not group blind) post-processor that outputs 0 on one group and 1 on the other; PPV will not be equalized.

Proof of Claim: 3.3.5.1. We prove only that PPV is equalized; the remaining properties may be proved similarly. Let $\hat{Y} = \hat{D} \circ \hat{S}$ be a hard classifier $\hat{Y}$ that is a group blind post-processor $\hat{D}$ composed with a calibrated soft classifier $\hat{S}$. All probabilities below are over $X_g \sim \mathcal{X}_g$, and the coins of $\hat{S}$ and $\hat{D}$.

$$
\begin{aligned}
&\mathsf{PPV}_{\hat{Y},g} = \Pr[Y(X_g) = 1 \mid \hat{Y}(X_g, g) = 1] \\
&= \frac{\Pr[Y(X) = 1]}{\Pr[\hat{Y}(X_g, g) = 1]} \\
&\quad \cdot \sum_{s \in \mathsf{Supp}(\hat{\mathcal{P}}_g)} \Bigg(\frac{\Pr[Y(X_g) = 1 \mid \hat{S}(X_g) = s] \cdot \Pr[\hat{S}(X_g) = s]}{\Pr[Y(X_g) = 1]} \\
&\qquad\qquad \cdot \Pr[\hat{Y}(X_g, g) = 1 \mid \hat{S}(X_g) = s, Y(X_g) = 1] \Bigg)
\end{aligned}
$$

Each factor in this product is equal across groups by the assumptions. Namely, $\Pr[Y(X_g) = 1]$ is equalized by calibration and equalized APs; $\Pr[\hat{Y}(X_g, g) = 1]$ by group blindness and equalized APs; $\Pr[\hat{Y}(X_g, g) = 1 \mid \hat{S}(X_g) = s, Y(X_g) = 1]$ by group blindness; $\Pr[Y(X_g) = 1 \mid \hat{S}(X_g) = s]$ by calibration; and finally $\Pr[\hat{S}(X_g) = s]$ by equalized APs. □

Mass Averaging

The mass-averaging technique is best illustrated with an example. Suppose that $\hat{\mathcal{P}}_{g_1}$ is uniform over $\{0, 0.5, 1\}$, and $\hat{\mathcal{P}}_{g_2}$ is uniform over $\{0, 1\}$. It is easy to define a soft post-processor $\hat{D}^{\mathsf{soft}}$ which equalizes these two APs. On g_1, we leave the score unchanged: $\hat{D}^{\mathsf{soft}}(s, g_1) = s$. On g_2, we compute the output as

$$\hat{D}^{\mathsf{soft}}(s, g_2) = \begin{cases} s & \text{w.p. } 2/3 \\ 0.5 & \text{w.p. } 1/3 \end{cases}.$$

The APs for groups g_1 and g_2 of the resulting soft classifier $\hat{S}' = \hat{D}^{\mathsf{soft}} \circ \hat{S}$ are equal, and are equal to $\hat{\mathcal{P}}_{g_1}$.

In the example, the probability mass is being redistributed by averaging the scores. This can be equivalently viewed as adding noise to the scores and then recalibrating the scores, something discussed in [92].

More generally, a mass-averaging post processor $\hat{D}^{\mathsf{soft}}$ assigns to each possible pair (s, g) a distribution over possible output scores s'. Such a $\hat{D}^{\mathsf{soft}}$ is fully specified by $k \cdot k' \cdot |\mathcal{G}|$ parameters, where k is the number of possible values of s and k' is the number of possible values of s'. Given a soft classifier $\hat{S}$ and a mass-averaging post processor $\hat{D}^{\mathsf{soft}}$, the constraint that the resulting APs are equalized across groups is linear in these parameters. Such classifiers, therefore, may be found by a linear program. We do not explore the choice of mass-averaging post-processors further.

Note that this method can only "remove" information from the distribution.

3.4 Results for Negative Predictive Value and continuous, full support APs

3.4.1 Results for Negative Predictive Value

In Section 3.3, we proved limits on the ability of post-processors to equalize PPV given a distribution on calibrated scores with finite support. For completeness, in this section, we give the statements of the analogous limits for equalizing NPV and for continuous probability density functions with full support $[0, 1]$.

We start with the analogous statement of Proposition 3.3.2.1 for NPV instead of PPV.

Proposition 3.4.1.1. *Fix two disjoint groups g_1 and g_2 with respective base rates BR_1 and BR_2 such that $\mathsf{BR}_1 \neq \mathsf{BR}_2$. Then there exists a soft-valued classifier $\hat{S}$ that is groupwise calibrated, but for which there is no post-processor $\hat{D} : [0, 1] \times \mathcal{G} \to \{0, 1\}$ such that $\hat{D} \circ \hat{S}$ equalizes NPV, unless $\Pr[\hat{D}(BR_i, g_i) = 0] = 0$ for $i = 1$ or 2.*

The nontriviality condition ensures that the NPV is well-defined on both groups (which can be compared to the nontriviality condition in Proposition 3.3.2.1, which ensures that the PPV is well-defined on both groups). The proof is essentially identical to the proof of Proposition 3.3.2.1: the fraction of predicted 0's in group g_i that are true 0's is $1 - \mathsf{BR}_i$, as the post-processor has no other information by which to make its decision, and hence the NPV remains unequal due to the differing base rates.

We now proceed to give the NPV analogs of our results on threshold post-processors in Section 3.3. We start with the analogous statement for Proposition 3.3.4.2 - that there is a class of simple group blind threshold post-processors that equalizes the NPV across groups.

Proposition 3.4.1.2. *For every nice groupwise calibrated soft classifier $\hat{S}$ and for every group blind threshold post-processor $\hat{D}_{(\tau,\rho)}$ such that $\tau(g) = \min(\mathsf{Supp}(\hat{\mathcal{P}}_g))$ and $\rho(g) < 1$ for all g, the composed classifier $\hat{D} \circ \hat{S}$ equalizes NPVs across all groups.*

The existence of the threshold post-processors in Proposition 3.4.1.2 follows from the assumed finiteness of the range of the soft classifier. In the case where the range of the soft classifier is infinite, such post-processors may not exist (as there may be no minimum element of the support). The proof is once again analogous to the proof of Proposition 3.3.4.2: these classifiers only ever output 0 on the minimum element of the support of $\hat{\mathcal{P}}_g$, and hence the NPV is simply 1 minus the smallest element of the support for each group.

However, much like the case for PPV, other group blind threshold post-processors cannot possibly equalize NPV.

Proposition 3.4.1.3. *There exists a groupwise-calibrated soft classifier with a nice AP for which no non-trivial group blind threshold post-processor, other than the ones in Proposition 3.3.4.2, can equalize NPV across groups.*

The example of the groupwise-calibrated soft classifier is the exact same one that shows the statement for PPV. Indeed, we can see this in the following way. We can make the example in the proof of Proposition 3.4.1.3 have equal base rates across groups, in which case it follows from Proposition 3.3.4.3 due to Claim 3.3.4.10.

When we turn to threshold post-processors that are not group blind, we again get analogous results for NPV.

Proposition 3.4.1.4. *Let $\mathcal{G}$ be a set of groups. For any soft classifier $\hat{S}$ with a nice AP $\hat{\mathcal{P}}$ such that $\hat{S}$ is groupwise-calibrated over $\mathcal{G}$ and $|\mathsf{Supp}(\hat{\mathcal{P}}_g)| \geq 2$ for all $g \in \mathcal{G}$, there exists a (*non-group blind*), non-trivial threshold post-processor $\hat{D}_{(\tau,\rho)}$ that is* not *one of the group blind post-processors in Proposition 3.3.4.2, such that the hard classifier $\hat{Y} = \hat{D}_{(\tau,\rho)} \circ \hat{S}$ equalizes NPV across $\mathcal{G}$.*

This holds even if we require that the NPV of all the groups is an arbitrary value in $(s_{min}, \min_i \mathsf{BR}_{g_i})$, where $\min_i \mathsf{BR}_{g_i}$ is the minimum base rate among the groups and s_{min} is the minimum score in the support of $\hat{\mathcal{P}}_{g_i}$.[2]

Again the proof of this follows via the same kind of continuity argument as we used to prove Proposition 3.3.4.5. By definition, each group has base rate at least

[2] For the case where the support of $\hat{\mathcal{P}}_{g_i}$ is infinite, s_{min} should be the infimum of scores.

$\min_i \mathsf{BR}_{g_i}$, and so if the post-processor always says 0 on some group i, then $\mathsf{NPV}_{g_i} = \mathsf{BR}_{g_i} \geq \min_i \mathsf{BR}_{g_i}$. Hence, for each group, we can start with the always-0 classifier and slide down the threshold until the desired NPV is reached.

Finally, the socially unsatisfying example also generalizes to NPV. A privileged group will have higher scores than a disadvantaged group in general, and hence if they are given the same threshold, the NPV will be lower on the privileged group. To rectify this, the threshold for the disadvantaged group will have to moved higher, to decrease the NPV. But then, the disadvantaged group is being subjected to a harsher standard.

Finally we note that Claim 3.3.4.7 also can be written with NPV instead of PPV, where the proof follows from using the characterization of NPV given in Proposition 3.3.1.1:

Claim 3.4.1.5 (Continuity of NPV). *Fix a soft classifier $\hat{S}$ and a corresponding AP $\hat{\mathcal{P}}$, as well as a group g. Let $\hat{D}_1$ and $\hat{D}_2$ be two post-processing algorithms. Let $\hat{\mathcal{P}}_{g,\hat{D}_1}$ be the expected conditional AP that results from starting with the AP $\hat{\mathcal{P}}_g$ over scores in group g and then conditioning on the scores that $\hat{D}_1$ sends to 0, and define $\hat{\mathcal{P}}_{g,\hat{D}_2}$ similarly. If $d_{TV}(\hat{\mathcal{P}}_{g,\hat{D}_1}, \hat{\mathcal{P}}_{g,\hat{D}_2}) < \epsilon$, then $|\mathsf{NPV}_{g,\hat{D}_1 \circ \hat{S}} - \mathsf{NPV}_{g,\hat{D}_2 \circ \hat{S}}| < O(\epsilon)$.*

We omit the proof of Claim 3.4.1.5, which resembles the proof of Claim 3.3.4.7 and follows from Proposition 3.3.1.1.

3.4.2 Results for continuous, full support APs

In this section, we briefly address how to extend our results on thresholds in Section 3.3 to the setting where every AP $\hat{\mathcal{P}}$ is a continuous probability distribution with $\mathsf{Supp}(\hat{\mathcal{P}}_g) = [0, 1]$ for all $g \in G$ - that is, the support equals the entire interval $[0, 1]$ for each group $g \in G$. Note that this automatically makes $\hat{\mathcal{P}}$ a "nice" AP, and hence rules out the general counterexample we came up with in Proposition 3.3.2.1. For the purposes of this section, call such an AP that 1) $\hat{\mathcal{P}}_g$ is a continuous probability density function for every $g \in G$ and 2) $\mathsf{Supp}(\hat{\mathcal{P}}_g) = [0,1]$ for all $g \in G$ a *very nice* AP. As the name suggests, we can extend the remaining results in Section 3.3 to the setting of very nice AP. We give the results for equalizing PPV as done in Section 3.3: extending these results to equalizing NPV in the setting of continuous, full support AP can be accomplished by combining the statements here with the modifications described in Section 3.4.1.

First, we note that a threshold post-processor can be described much more easily in the continuous setting than in the setting where the AP has finite support on each group. Indeed, in the setting where $\hat{\mathcal{P}}_g$ is a continuous density function for all $g \in G$, the post-processor can truly be a threshold, with no question of how to classify the score that is exactly equal to the threshold τ. This is because the score τ has probability 0 under the density $\hat{\mathcal{P}}_g$.

Hence, Proposition 3.3.4.2 has no true analog in this setting. This follows because the maximum element of the support in this case is 1, and a threshold at $\tau = 1$ sends every score (outside of a measure 0 set) to 0.

This allows us to strengthen Proposition 3.3.4.3 accordingly.

Proposition 3.4.2.1. *There exists a groupwise-calibrated soft classifier with a very nice AP for which no non-trivial group blind threshold post-processor can equalize* PPV *across groups.*

This follows from a nearly identical stochastic domination argument to the one used for Proposition 3.3.4.3 - in fact, the natural generalization of the distributions given for the proof of Proposition 3.3.4.3 to the continuous and full-support setting can be used in this proof.

A non group blind threshold can still always equalize PPV for very nice APs.

Proposition 3.4.2.2. *Let $\mathcal{G}$ be a set of groups. For any soft classifier $\hat{S}$ with a very nice AP $\hat{\mathcal{P}}$ such that $\hat{S}$ is groupwise-calibrated over $\mathcal{G}$, then there exists a non group blind, non-trivial threshold post-processor such that the hard classifier equalizes* PPV *across $\mathcal{G}$.*

This holds even if we require that the PPV of all the groups is equal to an arbitrary value in $(\max_i \mathsf{BR}_{g_i}, 1)$*, where* $\max_i \mathsf{BR}_{g_i}$ *is the maximum base rate among the groups* $g_i \in \mathcal{G}$*.*

This follows from the same continuity approach as the proof of Proposition 3.3.4.5, by sliding the threshold continuously down from 1 until the PPV reaches the desired value $v \in (\max_i \mathsf{BR}_{g_i}, 1)$.

The socially unsatisfying example generalizes in the natural way - Example 3.3.4.8 consists of one group having a monotonically increasing PMF and another one having a monotonically decreasing PMF. We can skip the discretization step in the definition of these PMFs and have them be continuous PDFs, and the example still goes through.

We cannot equalize PPV and NPV simultaneously in general, just like in the finite support case (Proposition 3.3.4.9).

Proposition 3.4.2.3. *Fix groups g_1 and g_2. There exists a soft classifier $\hat{S}$ with a very nice AP $\hat{\mathcal{P}}$ such that no threshold post-processor can simultaneously equalize PPV and NPV between groups g_1 and g_2.*

The proof we give of Proposition 3.3.4.9 for finite support naturally generalizes to this case - in fact, we can simply use the same proof but without discretizing the probability distributions. The necessary lemmas about monotonicity of PPV and being able to express the base rate as a convex combination of PPV and NPV still hold.

It is unsurprising that the result in Section 3.5 on using thresholds *with deferrals* to equalize PPV and NPV also goes through for very nice AP.

Proposition 3.4.2.4. *Let $\hat{S}$ be groupwise calibrated for the n groups $g_1, \ldots, g_n$, and suppose that $\hat{S}$ has a very nice AP. Then there exists a nontrivial threshold decision rule such that the hard classifier $\hat{Y} = \hat{D} \circ \hat{S}$ equalizes PPV and NPV for $\mathcal{G}$.*

Again, the explanation is very similar to the one for Proposition 3.4.2.2. PPV and NPV change continuously when we slide the respective thresholds, and unlike the case without deferrals, we can change the PPV *without* changing the NPV, by keeping the "0" threshold still and sliding the "1" threshold (and deferring in the middle). Hence, we can simply continuously slide the two thresholds on each group until they reached the desired values.

3.5 Post-processing calibrated classifiers with deferrals

In the first part of the paper, we considered the problem of post-processing calibrated soft classifiers, which output a score $s \in [0,1]$, into fair hard classifiers, which output a decision in $\hat{y} \in \{0,1\}$, subject to a number of group fairness conditions. In the remainder of this work, we reconsider this problem, but with one important change: we allow classifiers to "refuse to decide" by outputting the special symbol $\perp$. We call such classifiers *deferring* classifiers, borrowing the nomenclature from [217]. The output $\perp$ is the deferring classifier's way of refusing to make a decision and deferring to a downstream decision maker. For example, a risk assessment tool might aid a parole board to make a decision by categorizing an individual as high risk or low risk, or it might output $\perp$, providing no advice and deferring to the judgment of the board.

We now modify our notation appropriately. Instances x are still associated with a true type $Y(x) \in \{0,1\}$ and a group $G(x) \in \mathcal{G}$. A deferring hard classifier $\hat{Y}$ is a randomized function $\hat{Y} : \mathcal{X} \to \{0,1,\perp\}$. A deferring soft classifier is a randomized function $\hat{S} : \mathcal{X} \to [0,1] \cup \{\perp\}$. A deferring hard (resp. soft) post-processor is a randomized function $\hat{D} : [0,1] \cup \{\perp\} \times \mathcal{G} \to \{0,1,\perp\}$ (resp. $\hat{D}^{\mathsf{soft}} : [0,1] \cup \{\perp\} \times \mathcal{G} \to [0,1] \cup \{\perp\}$) that takes as input the output of a deferring soft and post-processes it into a deferring hard (resp. soft) classifier. We also introduce new versions of the FPR and FNR, conditioned on not deferring.

Definition 3.5.0.1. The *conditional false positive rate* and *conditional false negative* *rate* of a deferring hard classifier $\hat{Y}$ for a group g are, respectively:

$$\begin{aligned}\mathsf{cFPR}_{\hat{Y},g} &= \Pr[\hat{Y}(X_g) = 1 \mid Y(X_g) = 0, \hat{Y}(X_g) \neq \bot] \\ \mathsf{cFNR}_{\hat{Y},g} &= \Pr[\hat{Y}(X_g) = 0 \mid Y(X_g) = 1, \hat{Y}(X_g) \neq \bot].\end{aligned}$$

We additionally consider a version of the accuracy profile conditioned on not deferring, which we call the *conditional AP*. For non-deferring soft classifiers, Definitions 3.5.0.2 and 3.2.2.1 coincide.

Definition 3.5.0.2. The *conditional AP* $\hat{\mathcal{P}}_g$ of a classifier $\hat{S}$ for a group g is the PMF of $\hat{S}(X_g)$, conditioned on not outputting $\bot$. That is, for $s \in [0,1]$, $\hat{\mathcal{P}}_g(s) = \Pr[\hat{S}(X_g) = s \mid \hat{S}(X_g) \neq \bot]$. Note that the conditional AP is undefined if $\Pr[\hat{S}(X_g) \neq \bot] = 0$.

Abusing notation, we denote by $\hat{\mathcal{P}}$ the collection $\{\hat{\mathcal{P}}_g\}_{g \in \mathcal{G}}$, and call it the *conditional AP of* $\hat{S}$.

The conditional error rates are applicable generally, but they can be difficult to interpret. The consequences of using the conditional FPR and FNR are discussed further in Section 3.7 along with a discussion of different deferral models. They are also amenable to the consideration of additional goals which we will briefly address. For example, one could seek to minimize the total deferral rate, equalize the deferral rate among groups, or prefer deferrals on positive instances.

3.5.1 Thresholding with deferrals

We return now to the problem of post-processing of calibrated soft classifiers, but now with the extra power of deferring on some inputs. We revisit the two approaches discussed in Section 3.3: thresholding and equalizing APs.

Proposition 3.3.2.1 stated PPV and NPV cannot both be equalized across groups in general when using only a single threshold per group. By using two thresholds per groups and deferring on some inputs, PPV and NPV can always be equalized across groups.

We post-process using two thresholds per group as follows: return 0 when s is lower than the first threshold, return $\perp$ between the thresholds, and return 1 above the second threshold, as shown in Figure 3·4. This buys us more degrees of freedom when equalizing binary constraints, and it has the useful property that we say $\perp$ on the instances where we are the least confident about the predicted type.

We adapt our notation as follows:

Definition 3.5.1.1 (Deferring Threshold Post-Processor)**.** A deferring threshold post-processor $\hat{D}_{(\tau_0,\tau_1,\rho_0,\rho_1)}$ assigns to each group g two thresholds $\tau_0(g), \tau_1(g) \in \mathsf{Supp}(\hat{\mathcal{P}}_g)$, and two probabilities $\rho_0(g), \rho_1(g) \in [0,1]$, with the following requirements:

1. for all $g \in \mathcal{G}$, $\tau_0(g) \leq \tau_1(g)$
2. for all $g \in \mathcal{G}$ for which $\tau_0(g) = \tau_1(g)$, $\rho_1(g) + \rho_0(g) \leq 1$. This corresponds to the case where the two thresholds are the same, and therefore individuals with that score must be mapped to 1 with probability $\rho_1(g)$, and to 0 with probability $\rho_0(g)$, with the remainder mapped to $\perp$.

The corresponding threshold post-processor is defined as follows:

$$\hat{D}_{(\tau_0,\tau_1,\rho_0,\rho_1)}(s,g) = \begin{cases} 1 & s > \tau_1(g) \\ 0 & s < \tau_0(g) \\ \perp & \tau_0(g) < s < \tau_1(g) \\ 1 \text{ w.p. } \rho_1(g), \text{ else } \perp & s = \tau_1(g) \\ 0 \text{ w.p. } \rho_0(g), \text{ else } \perp & s = \tau_0(g) \\ 1 \text{ w.p. } \rho_1(g), 0 \text{ w.p. } \rho_0(g), \text{ else } \perp & s = \tau_0(g) = \tau_1(g) \\ \perp & s = \perp \end{cases}$$

Using two thresholds allows the equalization of both PPV and NPV across groups in general, whereas without deferrals we could only equalize one or the other. We first demonstrate the existence of post-processors that are fairly limited, analogously to those defined in Proposition 3.3.4.2.

Proposition 3.5.1.2. *Let $\hat{S}$ be a soft classifier with a nice AP for a set of groups $\mathcal{G}$. Then every threshold post-processor $\hat{D}_{(\tau_0,\tau_1,\rho_0,\rho_1)}$ satisfying the following properties*

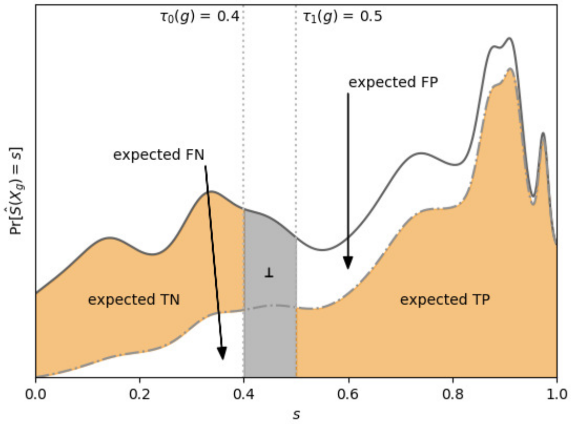

Figure 3·4: *For threshold post-processors with deferrals, defer between the thresholds.*

equalizes both the PPV *and* NPV *for all groups in* $\mathcal{G}$ *of the composed classifier* $\hat{Y} = \hat{D}_{(\tau_0,\tau_1,\rho_0,\rho_1)} \circ \hat{S}$*:*

1. $\tau_0(g) = \min(\mathsf{Supp}(\hat{\mathcal{P}}_g))$ *for all* g
2. $\rho_0(g) > 0$ *for all* g.
3. $\tau_1(g) = \max(\mathsf{Supp}(\hat{\mathcal{P}}_g))$ *for all* g
4. $\rho_1(g) > 0$ *for all* g.

Notice that these classifiers cannot be trivial, because we defined ρ_0 and ρ_1 in a way that prohibits the possibility that the composed classifier never returns 0 or 1. For the cases where the range of soft classifier outputs is infinite and there is no max or min element, these classifiers do not exist.

Proof of Proposition 3.5.1.2. The reasoning is similar to the non-deferral case for equality of PPV alone. The thresholds only allow one score s to map to 0 and one score to map to 1. Thus, PPV for both groups is equal to the largest score in the support and NPV for both groups is equal to 1 minus the smallest score in the support. □

Now, much like in Proposition 3.3.4.5, which showed the existence of meaningful non-trivial threshold post-processors that equalized PPV across groups, we show the existence of meaningful, nontrivial *deferring* threshold post-processors that equalize PPV and NPV across groups.

Proposition 3.5.1.3. *Let $\hat{S}$ be a soft classifier with a nice AP that is groupwise calibrated for a set of groups $\mathcal{G}$. Suppose that $|\mathsf{Supp}(\hat{\mathcal{P}}_g)| \geq 2$ for all $g \in \mathcal{G}$. Then there exists a non-trivial threshold post-processor $\hat{D}_{(\tau_0,\tau_1,\rho_0,\rho_1)}$ that is not one of those defined in Proposition 3.5.1.2, such that the hard classifier $\hat{Y} = \hat{D}_{(\tau_0,\tau_1,\rho_0,\rho_1)} \circ \hat{S}$ equalizes PPV_g and NPV_g for all $g \in \mathcal{G}$.*

The main idea of the proof of this proposition is to use Proposition 3.3.4.5 twice: once for getting thresholds to equalize the PPV, and once for thresholds to equalize the NPV. These thresholds may be invalid because there may be a group g for which $\tau_0(g) > \tau_1(g)$. We use Claims 3.3.4.6 and 3.3.4.7 to allow ourselves to push the PPV thresholds toward 1 and the NPV thresholds toward 0 until they no longer overlap, while still maintaining equalization of PPV and NPV for the other groups.

Proof of Proposition 3.5.1.3. Recall by Claim 3.3.4.6 that the PPV of $\hat{S}$ on a group g monotonically increases as $\tau_1(g)$ increases and, if $\tau_1(g)$ is constant, as $\rho_1(g)$ increases. By Claim 3.3.4.6, NPV monotonically increases as $\tau_0(g)$ decreases, and, if $\tau_0(g)$ is constant, as $\rho_0(g)$ increases. Recall by Claim 3.3.4.7 that if the total variance distance between conditional APs (conditioned on being post-processed to a result of 1) for different groups is at most ϵ, then the PPV difference for these groups is bounded by $O(\epsilon)$. Thus, PPV_g is continuous and monotonically increasing with regard to $(\tau_1(g), \rho_1(g))$. Similarly, NPV_g is continuous and monotonically increasing with $(-\tau_0(g), \rho_0(g))$.

We know by Proposition 3.3.4.5 that there exists a non-group blind threshold rule (without deferrals) that equalizes the PPV among the groups. By the analogous Proposition 3.4.1.4, there exists a (different) non group blind threshold rule that equalizes the NPV among the groups. For both of these, we know that they are not the classifiers from Proposition 3.5.1.2.

If the thresholds meet the conditions of being a deferring post-processor listed in Definition 3.5.1.1, then the statement is proven. If they do not meet the conditions because the thresholds "overlap," we repeat the following procedure until the conditions are met:

1. Let g be a group for which the conditions are not met, i.e. either $\tau_0(g) > \tau_1(g)$, or $\tau_0(g) = \tau_1(g)$ and $\rho_0(g) + \rho_1(g) > 1$.

2. If $\tau_0(g) > \tau_1(g)$, define $t' = \frac{\tau_0(g)+\tau_1(g)}{2}$. Let $t = \arg\min_{s \in \mathsf{Supp}(\hat{\mathcal{P}}_g)} |s - t'|$. Notice that $t \leq \tau_0(g)$ and $t \geq \tau_1(g)$, but because by assumption $\tau_0(g) > \tau_1(g)$, t cannot be equal to both thresholds. Set the new value for both thresholds to t: $\tau_0(g) = \tau_1(g) = t(g)$.

3. If $\rho_0(g) + \rho(g) > 1$, then do the following:

   (a) If $\tau_0(g)$ remained unaltered in the previous step, then keep $\rho_0(g)$ the same, and set $\rho_1(g) = 1 - \rho_0(g)$.

   (b) If $\tau_1(g)$ remained unaltered in the previous step, then set $\rho_1(g)$ the same and set $\rho_0(g) = 1 - \rho_1(g)$.

   (c) If neither of these is true, then let $r = \frac{\rho_0(g)}{\rho_0(g)+\rho_1(g)}$. Set $\rho_0(g) = r$ and $\rho_1(g) = 1 - r$.

4. These thresholds are no longer overlapping, but they altered PPV_g and NPV_g. Notice that, by the monotonicity properties described above, the threshold rules were changed in ways that can only increase PPV_g or NPV_g:

   (a) $\tau_1(g)$ has increased or remained constant

   (b) if $\tau_1(g)$ remained constant, then $\rho_1(g)$ also remained constant

   (c) $\tau_0(g)$ has decreased or remained constant

   (d) if $\tau_0(g)$ remained constant, then $\rho_0(g)$ remained constant

   The new PPV and NPV for g may now be higher than those of the other groups.

5. For all other groups $g' \neq g$, by the Intermediate Value Theorem and the continuity of NPV, there exists some $(\tau_0(g'), \rho(g'))$ that sets $\mathsf{NPV}_{g'} = \mathsf{NPV}_g$, and by the monotonicity of NPV, this threshold is lower than the old one. Similarly, there exists some $(\tau_1(g'), \rho(g'))$ that sets $\mathsf{PPV}_{g'} = \mathsf{PPV}_g$ and it is higher than the old one. By the monotonicity of PPV and NPV, we know that this process will not cause non-overlapping thresholds to become overlapping.

The ultimate effect of these steps was to reduce the number of overlapping thresholds by at least one. We can repeat this process up to $2|\mathcal{G}|$ times until none of the thresholds overlap.

Notice that this classifier is not one of the ones from Proposition 3.5.1.2 - if we did not have to correct for "overlapping," then this is true by assumption, and if we did do the correction process, then $\tau_0(g) = \tau_1(g)$ for at least one g, and by assumption we had $|\mathsf{Supp}(\hat{\mathcal{P}})| \geq 2$.

Thus, we have created valid a post-processor $\hat{D}_{(\tau_0,\tau_1,\rho_0,\rho_1)}$ that equalizes PPV and NPV for all groups simultaneously and is not one of the ones in Proposition 3.5.1.2, proving the claim. □

The following example demonstrates that it is sometimes possible to equalize PPV, NPV, FPR, and FNR using deferrals, but without equalizing the APs themselves:

Example 3.5.1.4 (Equalizing PPV, NPV, cFPR, and cFNR with Thresholds)**.** This example is presented with continuous support $[0, 1]$ for simplicity. Consider two APs, one for group g_1 and one for g_2. Let the AP for g_1 be uniform (with density give by the line $\hat{\mathcal{P}}(s) = 1$), and let the AP for group g_2 have density given by the parabola $\hat{\mathcal{P}}(s) = 6s(1 - s)$, as shown in Figure 3·5.

Consider the post-processor $\hat{D}^{\mathsf{soft}}_{(\tau_0,\tau_1)}$. (Note that in the case where the distributions are continuous, the randomization values that determine what happens when the result is *on* the threshold, ρ_0 and ρ_1, are meaningless because $\Pr[s = \tau_0(g)] = \Pr[s = \tau_1(g)] = 0 \forall s$.) Let $\tau_0(g_1) = \tau_0(g_1) = 0.5$, let $\tau_0(g_2) = \frac{1}{6}(5 - \sqrt{7})$ and let $\tau_1(g_2) = 1 - \frac{1}{6}(5 - \sqrt{7})$ as shown in Figure 3·5.

The PPV and NPV of both groups is $\frac{3}{4}$, and the cFPR and cFNR of both is $\frac{1}{4}$, thus equalizing all four values.

This example is somewhat unsatisfactory because the base rates are equal in the two groups. We did not find a similar example without equal base rates.

3.5.2 Equalizing APs with deferrals

As with Claim 3.3.5.1, equalizing the conditional APs between groups renders trivial the task of downstream decision-making subject to equality of PPV, NPV, cFPR, and cFNR. Importantly, unlike in Section 3.3.5, equalizing the conditional APs between groups does not require the groups to have equal base rates, greatly increasing the applicability of this approach.

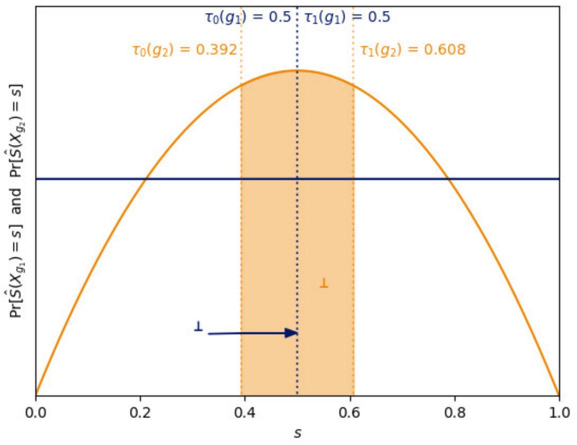

Figure 3·5: *This threshold post-processor equalizes* PPV*,* NPV*,* cFPR*, and* cFNR *as described in Example 3.5.1.4.*

Claim 3.5.2.1. *If the conditional APs are equal for two groups, then* PPV*,* NPV*,* cFPR*, and* cFNR *are equalized (or simultaneously undefined) by any hard deferring post-processor $\hat{D}$ satisfying (1)* group blindness *and (2) $\hat{D}(\perp, g) = \perp$ $(\forall g)$.*

The additional condition that $\hat{D}$ defers on input $\perp$ is necessary: if $\hat{D}$ output 1 on all inputs (even on $\perp$), then PPV would remain unequal as long as the base rates differed. The proof is similar to the proof of Claim 3.3.5.1.

Proof of Claim 3.5.2.1. We prove only that PPV is equalized; the remaining properties may be proved similarly. Let $\hat{Y} = \hat{D} \circ \hat{S}$. All probabilities below are over $X_g \sim \mathcal{X}_g$, and the coins of $\hat{S}$ and $\hat{D}$.

$$
\begin{aligned}
\mathsf{PPV}_{\hat{Y},g} &= \Pr[Y(X_g) = 1 \mid \hat{Y}(X_g, g) = 1] \\
&= \frac{\Pr[Y(X_g) = 1 \mid \hat{S}(X_g) \neq \perp]}{\Pr[\hat{Y}(X_g, g) = 1 \mid \hat{S}(X_g) \neq \perp]} \\
&\quad \cdot \sum_{s \in \mathsf{Supp}(\hat{\mathcal{P}}_g)} \Bigg(\frac{\Pr[Y(X_g) = 1 \mid \hat{S}(X_g) = s] \cdot \Pr[\hat{S}(X_g) = s \mid \hat{S}(X_g) \neq \perp]}{\Pr[Y(X_g) = 1 \mid \hat{S}(X_g) \neq \perp]} \\
&\qquad\qquad \cdot \Pr[\hat{D}(s, g) = 1 \mid \hat{S}(X_g) = s, Y(X_g) = 1] \Bigg)
\end{aligned}
$$

Each factor in this product is equal across groups by the assumptions. Namely, $\Pr[Y(X_g) = 1 \mid \hat{S}(X_g) \neq \perp]$ is equalized by calibration and equalized APs; $\Pr[\hat{Y}(X_g, g) = 1 \mid \hat{S}(X_g) \neq \perp]$ by group blindness and equalized APs; $\Pr[\hat{D}(s, g) =$

$1 \mid \hat{S}(X_g) = s, Y(X_g) = 1]$ by group blindness; $\Pr[Y(X_g) = 1 \mid \hat{S}(X_g) = s]$ by calibration; and finally $\Pr[\hat{S}(X_g) = s \mid \hat{S}(X_g) \neq \bot]$ by equalized APs.

□

Deferrals are a powerful tool for manipulating, and thereby equalizing, conditional APs. Consider a function $Q : (s, g) \mapsto [0, 1]$

$$\hat{D}_Q^{\mathsf{soft}}(s, g) = \begin{cases} \bot & \text{if } s = \bot \\ \bot \text{ w.p. } Q(s, g), \text{ else } s & \text{otherwise} \end{cases}$$

If $\hat{S}$ is a calibrated classifier, the soft deferring classifier $\hat{S}' := \hat{D}_Q^{\mathsf{soft}} \circ \hat{S}$ is still calibrated. For a group g, let $\hat{\mathcal{P}}_g$ be the AP of $\hat{S}$ and $\hat{\mathcal{P}}'_g$ be the AP of $\hat{S}'$. There is a simple graphical intuition for the shape of $\hat{\mathcal{P}}'_g$, as shown in Figure 3·6. More formally,

$$\hat{\mathcal{P}}'_g(s) = \frac{\hat{\mathcal{P}}_g(s)(1 - Q(s, g))}{1 - \Delta} \tag{3.7}$$

where $\Delta := \Pr[\hat{S}'(X_g) = \bot \mid \hat{S}(X_g) \neq \bot] = \sum_{s \in \mathsf{Supp}(\hat{\mathcal{P}}_g)} \hat{\mathcal{P}}_g(s) Q(s, g)$.

By appropriate choice of Q, any conditional AP can be transformed into almost any other conditional AP.

Theorem 3.5.2.2. *Let $\hat{\mathcal{P}}_g$ be a conditional AP of a soft classifier $\hat{S}$ on group g, and let $\hat{\mathcal{P}}^*$ be any probability mass function such that $\mathsf{Supp}(\hat{\mathcal{P}}^*) \subseteq \mathsf{Supp}(\hat{\mathcal{P}}_g)$. Then there exists Q for which the calibrated AP $\hat{\mathcal{P}}'_g$ of $\hat{D}_Q^{\mathsf{soft}} \circ \hat{S}$ is equal to $\hat{\mathcal{P}}^*$.*

Proof of Theorem 3.5.2.2. Let $\Delta = 1 - \min_{s \in \mathsf{Supp}(\hat{\mathcal{P}}_g)} \frac{\hat{\mathcal{P}}_g(s)}{\hat{\mathcal{P}}^*(s)}$. For all $s \in \mathsf{Supp}(\hat{\mathcal{P}}_g)$, let

$$Q(s, g) = 1 - \frac{\hat{\mathcal{P}}^*(s)}{\hat{\mathcal{P}}_g(s)} \cdot (1 - \Delta).$$

Observe that

$$\begin{aligned} \sum_{s \in \mathsf{Supp}(\hat{\mathcal{P}}_g)} \hat{\mathcal{P}}_g(s) Q(s, g) &= \sum_{s \in \mathsf{Supp}(\hat{\mathcal{P}}_g)} \Big(\hat{\mathcal{P}}_g(s) - (1 - \Delta)\hat{\mathcal{P}}^*(s) \Big) \\ &= \Delta \end{aligned}$$

and hence Δ is defined as in Equation 3.7, where we used the fact that $\sum \hat{\mathcal{P}}_g(s) = \sum \hat{\mathcal{P}}^*(s) = 1$ in the last line. Plugging into the earlier formula for $\hat{\mathcal{P}}'_g$ (Equation 3.7) completes the proof. □

Together, Theorem 3.5.2.2 and Claim 3.5.2.1 suggest a general framework for using deferrals to post-process a soft, possibly deferring classifier $\hat{S}$ which is groupwise calibrated into a hard deferring classifier which simultaneously equalizes PPV, NPV, cFPR, and cFNR across groups, as follows.

For each $g \in \mathcal{G}$, let $\hat{\mathcal{P}}_g$ be the conditional AP of $\hat{S}$ for group g. Let $\hat{\mathcal{P}}^*$ be any conditional AP such that $\mathsf{Supp}(\hat{\mathcal{P}}^*) \subseteq \cap_{g \in \mathcal{G}} \mathsf{Supp}(\hat{\mathcal{P}}_g)$. Use Theorem 3.5.2.2 to equalize the conditional APs for all groups $g \in \mathcal{G}$. Then use any hard post-processor $\hat{D}$ satisfying the requirements of Claim 3.5.2.1 to make the ultimate deferring hard classifier. This method is shown in Figure 3·6.

This framework allows for enormous flexibility in the choice of both $\hat{\mathcal{P}}^*$ and $\hat{D}$, even when considering just two groups g_1 and g_2. In Figure 3·10, we illustrate the first step of the framework on a COMPAS dataset using $\min\{\hat{\mathcal{P}}_{g_1}, \hat{\mathcal{P}}_{g_2}\}$ as $\hat{\mathcal{P}}^*$, where g_1 is African-Americans and g_2 is Caucasians. In Figures 3·8 and 3·9 in Section 3.6, we also use $\hat{\mathcal{P}}_{g_1}$ and $\hat{\mathcal{P}}_{g_2}$ as $\hat{\mathcal{P}}^*$.

One can design $\hat{\mathcal{P}}^*$ to achieve additional goals. For example, the choice $\hat{\mathcal{P}}^* = \min\{\hat{\mathcal{P}}_{g_1}, \hat{\mathcal{P}}_{g_2}\}$ results in *equal deferral rate* across each group (equal to the total variation distance between the two initial conditional APs). The framework can be further expanded by combining deferrals with other methods for manipulating conditional APs, including the mass-averaging discussed in Section 3.3.5. A better understanding of these techniques is left for future work.

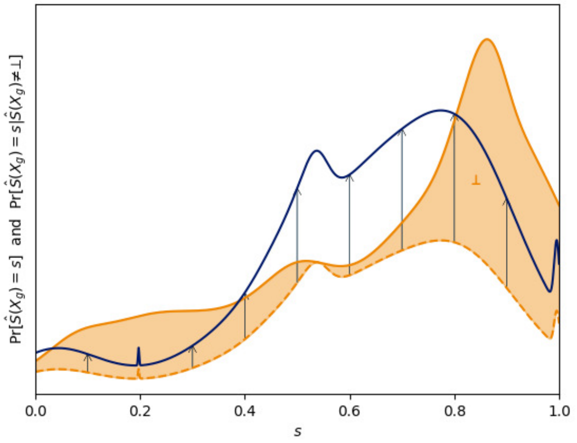

Figure 3·6: *Choosing deferrals cleverly allows transforming one AP into another (conditional) AP. In this example, the solid orange line is the original AP* $\hat{\mathcal{P}}_g = \Pr[\hat{S}(X_g) = s]$*. By deferring at the rates indicated by the shaded region, the resulting conditional AP* $\hat{\mathcal{P}}'_g = \Pr[\hat{S}(X_g) = s \mid \hat{S}(X_g) \neq \perp]$ *is represented by the dark blue line. The area of the shaded region is* Δ.

3.6 Experiments on COMPAS data

We test our methodology on the Broward County data made publicly available by ProPublica [11]. This data set contains the recidivism risk decile scores given by the COMPAS tool, 2-year recidivism outcomes, and a number of demographic and crime-related variables on individuals who were scored in 2013 and 2014. We restrict our attention to the subset of defendants whose race is recorded as African-American or Caucasian. These will form the two groups with respect to which we wish to examine different fairness criteria. After applying the same data pre-processing and filtering as reported in the ProPublica analysis, we are left with a data set on n = 5278 individuals, of whom 3175 are African-American and 2103 are Caucasian.

It has been shown that the COMPAS scoring mechanism is an approximately calibrated soft classifier with 10 possible outcomes [81, 124]. We note here that the distribution of the COMPAS scores differs significantly across the two groups. In particular, the scores for Caucasians are more evenly distributed as opposed to the skewed distribution seen with African-Americans.

3.6.1 Thresholding with deferrals

We first ran our two-threshold post-processing mechanism (Section 3.5.1) and obtained a binary decision algorithm with deferrals which equalizes both PPV and NPV across Caucasians and African-Americans (See Figure 3·7). For simplicity we avoid using randomization for members within a particular decile score and instead settle for approximate equalization of PPV and NPV. We observe that the percent of deferrals in total is smaller than 20% of the decisions to be made which shows that a fairly large number of the defendants can be classified without having to defer to a downstream decision maker.

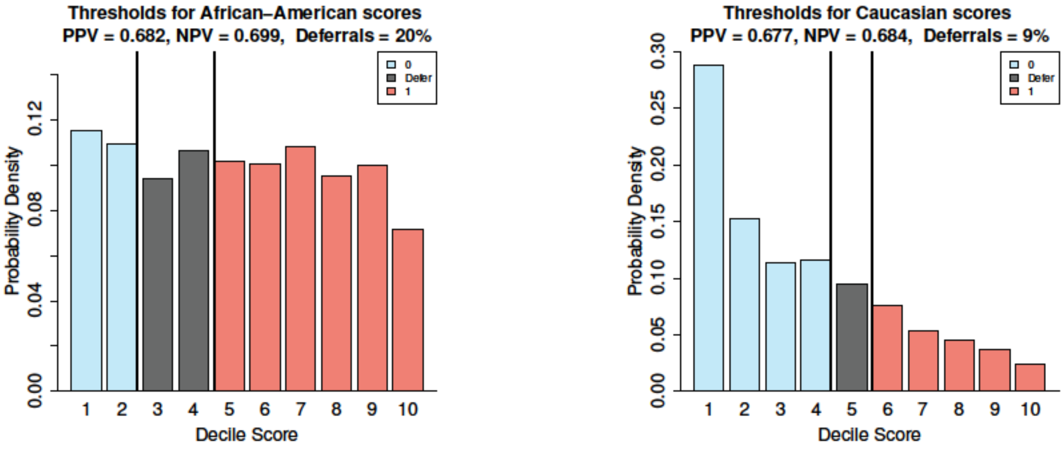

Figure 3·7: Two thresholds are applied to each AP for the COMPAS data from 2016, (approximately) equalizing PPV and NPV. In the left plot we show the thresholds for the African American group, and in the right plot, we show the thresholds for the Caucasian group.

The thresholds suffer from the issue highlighted in Example 3.3.4.8, demonstrating that blindly equalizing PPV and NPV using thresholds can be problematic. Namely, the resulting deferring hard classifier is much stricter for African-Americans than for Caucasians.

Next we look at our post-processing mechanisms to equalize all four quantities PPV, NPV, FPR, and FNR using deferrals (Section 3.5.2). As was noted earlier in the paper, equalizing the APs of the two groups post-deferral achieves the goal of equalizing all four of the above quantities. We implement two methods for doing so.

3.6.2 Converting one AP into another

In the first method, decisions are deferred only on one group so as to convert its AP into that of the other group. First, we consider deferring only on Caucasians to convert their AP into that of African-Americans (Figure 3·9); next, decisions are deferred only for African Americans (Figure 3·8).

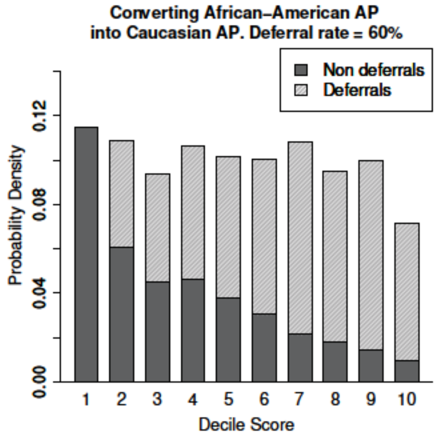

Figure 3·8: Our conditional AP equalization method applied to COMPAS data from 2016. We use deferrals to create a conditional AP for African Americans that matches the AP for Caucasians.

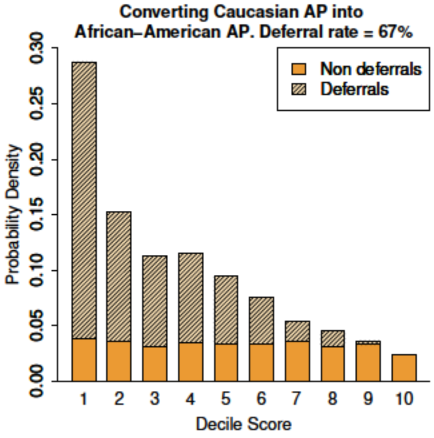

Figure 3·9: We create a conditional AP for Caucasians using the AP for African Americans. Notice the difference in the rates and distributions of deferrals between the two methods.

3.6.3 Equalizing APs

Alternately we have a second method where decisions are deferred for an equal fraction of Caucasians and of African Americans (Figure 3·10). This fraction is precisely equal to the statistical (*total variation*) distance between the distributions of scores produced by the soft classifier on the two groups.

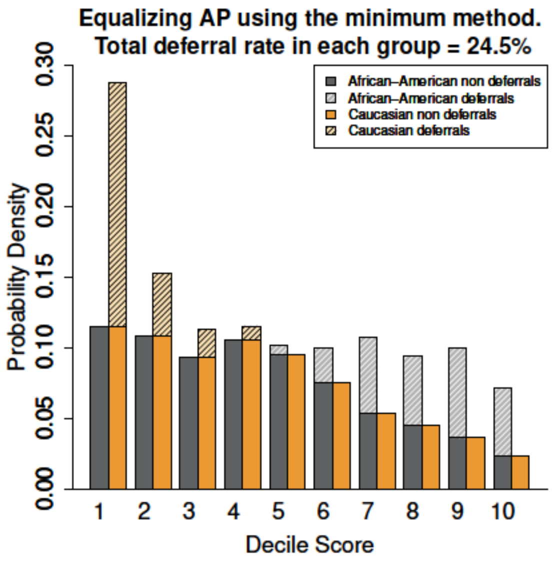

Figure 3·10: *A version of our conditional AP equalization method applied to COMPAS data from 2016. The AP for African Americans and the AP for Caucasians are converted into a conditional AP that has the same distribution as the pointwise minimum of the two APs. Notice that the total deferral rate is equalized between the two groups (this is equal to precisely the total variation distance between the two APs), but the distribution of deferrals across scores is not.*

We observe several striking phenomena on the COMPAS data set. First, using the method of deferring only on African-Americans we defer on roughly 36% of the total decisions. This number goes down to roughly 25% when we defer only on Caucasians. The total deferral fraction is also roughly 25% when we defer on an equal fraction of Caucasians and African-Americans.

Second, for all three methods that equalize the score distributions, on this particular dataset, deferrals happen more on the "extremes", namely on individuals with respect to which the classifier had relatively high confidence (either close to 0 or close to 1). This stands in sharp contrast to how the two-threshold method (Figure 3·7) distributes its deferrals—they occur only in the middle of the distribution (namely for elements for which the classifier is "unsure"). More work is needed to better understand the full space of deferral strategies.

Second, the closer the shapes of the accuracy profiles the lower the number of deferrals will be with this method.

3.7 Models of deferring

Whether or not a classifier is thought of as promoting fairness depends on the context; this is true for both deferring and non-deferring classifiers. In addition to the myriad considerations present for non-deferring classifiers, deferring classifiers and downstream decision makers introduce some additional axes for consideration.

Cost to the individual. Even though it is not intended to be a final decision, a deferral may impose burdensome costs to an individual being classified. It may mean that a defendant remains in jail while additional hearings are scheduled, that invasive and expensive medical tests are ordered, or that continued investigation engenders social stigma. These costs may not be borne equally by all individuals, and may depend on their group membership, their true type, or other factors. For example, a delay in granting a loan to an applicant may overly burden poorer applicants, even those very likely to repay.

Clear-cut answer. Some classification tasks have relatively clear-cut answers. Image classification (e.g. for content moderation) often has a relatively simple "true" answer as to whether the image contains a certain object or not. The more "factual" the question, the more clear-cut the answer (e.g. "does this image contain nudity" has a clearer-cut answer than "is this image offensive." However, even the murkiest image classification task is still more clear-cut than a prediction question. Recidivism prediction tools tend to identify a group of people where a certain percentage go on to recidivate later. This is no longer a matter of interpretation, it is now a matter of prediction. Even though a calibrated tool will likely return accurate percentages over a period of time, it is difficult to attribute that score to every person that has it, and furthermore if the circumstances change, the prediction is no longer valid.

Cost to the decider. Allowing deferrals might make the decision process more cost-effective: Given that in most cases making a determination is cheap, one may now invest more in the deferred cases. For instance, a team of trained moderators might be hired to manually review content on which an automated content filter defers, or an expensive investigation might be required to adjudicate insurance claims that are not cut-and-dry.

Accuracy of downstream decision. One reason to defer is to introduce a delay that will allow for a more accurate decision. Thus the usefulness of allowing a classifier to defer depends on the accuracy of the downstream decision maker. Additional medical tests might allow for highly accurate diagnoses. But a judge deciding bail will be prone to a variety of errors and biases.

"Fairness" of downstream decision (and of composed classifier). Similar to the above, the fairness of the downstream decision maker (however one wants to interpret that) will impact our interpretation of the deferring classifier. Here one should take into account also the "procedural" aspect of the two-step evaluation; here it is important that the downstream classifier will be deemed as "more fair" and "more knowledgeable" than the first stage. Exploring fairness criteria for systems of deferring classifiers and downstream decision-makers, e.g. as done in [57] did for non-deferring classifiers, is an interesting direction for future work.

Scale and scope of classification algorithm. Is the same algorithm being used across many different locales (e.g. image classification by large social media companies)? Or are many different algorithms being used for small settings (e.g. judges will judge differently from each other)?

Frequency of decisions. In many settings, the deferring classifier is a fast, automated test (e.g., automated risk assessment) while the downstream decision maker is a slow, manual process (e.g., parole board). However, we anticipate situations in which there may be repeated deferring classifiers chained together which comprise the complete decision making pipeline. For example, a doctor might have a sequence of diagnostic tests at her disposal as needed, or a bank might allow many rounds of appeal for loan applications, but with lengthy delays. Some applications might even permit hundreds or thousands of near-continuous deferring classifiers. As an example, consider a live video streaming platform that passively monitors streams for inappropriate content in real time. The automated passive monitor might decide the content is inappropriate, and shut it down; appropriate, and continue passive monitoring; or suspicious (by deferring), and begin active monitoring by devoting more computing resources or bringing in a human moderator.

3.7.1 Technical implications of deferral model

The contextual considerations discussed above directly impact the appropriate application of a deferring classifier and its goals. An obvious goal is to minimize the overall rate of deferrals while maintaining the best possible FPR, FNR, PPV, and NPV for the classifier conditioned on not deferring, and without considering the distribution of deferrals. However, one might desire very different properties from the distribution of deferrals in different contexts. The deferrals may be distributed differently among individuals with different true type, group membership, or soft classifier scores, while the burden imposed by deferrals and errors may differ greatly between different populations.

In a medical diagnosis scenario, a false negative (i.e., failing to diagnose a disease) may have serious consequences, and deferring to run additional non-invasive and inexpensive additional tests may be generally acceptable. On the other hand, an insurance provider may prefer to minimize expensive investigations by paying out more false claims.

The context may also affect the way one defines the deferral analogues of FPR and FNR. While calibration, PPV, and NPV apply directly to deferring classifiers, it is not clear how best to generalize the definitions of error rates. For example, consider false positive rate: by Definition 3.2.3.1, the false positive rate of a non-deferring hard classifier $\hat{Y}$ for a group g is $\mathsf{FPR}_{\hat{Y},g} = \Pr[\hat{Y}(X_g) = 1 \mid Y(X_g) = 0]$.

The approach we take in Section 3.5 is to condition on not deferring (Definition 3.5.0.1). A deferring classifier $\hat{Y}$ that output 1 on half of true negative instances (within a g) would have conditional false positive rate as low as 0.5 (if it never output $\perp$ on true negatives) or as high as 1 (if it never output 0 on true negatives). The conditional false positive rate is agnostic towards the downstream decision maker. It codifies no value judgements as to whether a deferral is desirable or undesirable as an individual nor whether deferrals ultimately result in accurate or inaccurate decisions. (This is itself a value judgement.)

A second approach is to leave the original definition unchanged. The same deferring hard classifier as above would have unconditional false positive rate 0.5. This would be true regardless of whether $\hat{Y}$ output 0 or $\perp$ on the other half of true negative instances. We call this the *unconditional false positive rate*. The unconditional false positive rate effectively categorizes deferrals as correct outputs. This may be appropriate if the downstream decision maker has very high accuracy. If, for example, a doctor orders an additional, more accurate diagnostic test in response to a deferral, the unconditional false positive rate might be appropriate.

Finally, a third approach is to base our measure of inaccuracy on true negatives instead of false positives, a reverse of the above.

Just as in the case of non-deferring classifiers, the relationships among these contrasting group statistics, their meaningfulness in different settings, and their application in different settings are not well understood and deserve further study.

Chapter 4

Improvements to zero-knowledge argument systems

This section is based on joint work with Yaron Gvili and Mayank Varia [157], and with Yaron Gvili, Julie Ha, Mayank Varia, Ziling Yang, and Xinyuan Zhang [155].

4.1 Introduction

As we described in §1.2.3, the work in this chapter is not inherently interdisciplinary, instead it focuses on improvements to the proof size of two zero-knowledge arguments of knowledge. We provided several uses of zero-knowledge proofs in both the legal setting and a more typical setting in §1.2.3 which described how the improvement of these protocols is a meaningful step forward in scenarios where these proofs are used. We will first describe the myriad of uses zero-knowledge proofs have found in both legal settings and more traditional cryptography settings, then we briefly summarize the main methods for creating zero knowledge proofs and their tradeoffs, and then finally we describe our two specific zero-knowledge proof improvements for the remainder of the chapter.

Uses of zero-knowledge proofs

Zero-knowledge proofs have transformed from a theoretical idea to a highly practical, widely-used cryptographic primitive. Today they find uses in delegated computation [17, 73, 82], verifiable computation on secret data [94, 121, 243], digital signature schemes [28, 188], identification schemes [67], malicious-secure multi-party computation protocols [22, 38, 142, 190, 213], anonymous credentials and proofs of identity [27, 68, 116], digital watermarking [2, 3, 97], voting [151, 236], and a wide variety of roles in distributed computation blockchain, and anonymous cryptocurrency applications [33, 34, 225, 244, 321].

Beyond these typical use cases, they have also found more specialized roles in various legal areas, which we proceed to describe in this section.

Glaser et al. [141] propose a zero-knowledge proof for nuclear armament verification. Under some nuclear arms-control agreements, international inspectors must inspect individual nuclear warheads to verify that they are of the type claimed, i.e. they contain specific quantities of various materials. However, states generally do not want to reveal detailed classified information about the warheads themselves to the inspectors. Glaser et al. propose a zero-knowledge proof for verifying the substances in a warhead, taking advantage of the fact that two warheads containing the same contents would result in similar "radiographs": patterns of neutron transmission and emission counts when irradiated with high-energy neutrons (aside from statistical noise), whereas changing the contents of the warhead would yield a different "fingerprint." Rather than revealing the fingerprint itself, inspectors would use a device to measure the noisy "difference" between the expected and actual radiographs. If the warheads are of the type they say they are, then the inspector will see random noise (and learn nothing about the highly classified radiograph of the warhead itself). However, if the warhead actually contains different material than is claimed, the inspector will see a large value: the difference between the expected and actual radiographs. This will inform them that the materials in the warhead are different than its owner claimed. This example demonstrates the potential for use of zero-knowledge proofs even in highly sensitive and political situations.

Fisch et al. [122] provided a zero-knowledge protocol for proving non-matches of DNA with crime scene evidence while keeping the genome itself private, and one for proving a parent-child relationship in DNA. Their methods address the privacy of not only the abstract DNA-comparison process itself, but also the physical preparation of samples.

A number of works propose zero-knowledge proof-based methods for cryptographically enforcing rules governing data surveillance or warrants [126, 147, 204]. This number grows further if you liberally consider "zero-knowledge" to include multi-party computation, private information retrieval, or private set intersection-based approaches [186, 265]. The goal of these methods is to verify cryptographically that policies are actually being followed, since the normal transparency that would allow easy auditability is not present in these classified or otherwise secret contexts.

A whole host of functionalities have been proposed to be performed as "smart contracts" [277], contracts that consist of a block of code that executes when various conditions are met, typically enabled by a blockchain of some kind (see [227] for a general overview, and see [16] for Bitcoin-specific smart contracts). Most smart contracts make significant use of zero-knowledge proofs; a typical contract would be for Alice to transfer some funds to Bob if and only if Bob proves (in zero knowledge) that he has fulfilled the terms of the contract.

Bamberger et al.'s work on "verification dilemmas" [19] is an interdisciplinary work that describes four particular areas of law in which zero-knowledge proofs would be particularly helpful. The first is identity verification, where zero-knowledge proofs would provide an appealing alternative to high amounts of aggregated data about individuals being used as a surrogate attempt at verification. "Anonymous credentials" are a classic zero-knowledge use case [67], for instance proving your age, address, citizenship, or other properties, all without disclosing your identity. However, as a practical matter, the key management and other infrastructure that would be required to enable this sort of interaction is unlikely to garner the political support needed for its adoption in America anytime soon. The one exception is in finance, where e- and cryptocurrency have warmed people to the idea of proving that one has the funds to execute a transaction without revealing any other information. The second verification dilemma involves the buying and selling of information, avoiding Arrow's fundamental paradox that in order to sell information to a potential buyer, a seller must first describe that information, but doing so gives up the information and nothing remains to sell (modulo external legal protection of the information, e.g. intellectual property law). As the authors note, much information of this style is "not easily digitized," however there are good opportunities here for multiple companies to determine what portion of their user base overlaps, prove the efficacy of a proprietary machine learning algorithm or a training dataset, or show that an algorithm is "novel" by showing that it is different from other algorithms. Their third category is cryptographic verification of government oversight, similar to the examples described earlier but focusing more on tax auditing by the IRS. Last, since in order to accuse an organization of trade secret violation, the plaintiff would have to reveal some information about their secret information to the courts or the other company (which is required so as to allow the defendants to adequately defend themselves), many potential plaintiffs choose not to sue at all, since revealing the information may be more costly than whatever they would recover from the defendant. Zero knowledge proofs might be used to demonstrate some property of the information without being forced to disclose the information itself, potentially avoiding needing to reveal the trade secret to the court in order to prosecute.

We hope these examples demonstrate that there is a good deal of interest in improving zero-knowledge proofs for use in the legal setting. We next proceed to provide an introduction to zero-knowledge proofs in general, and then describe the two specific proof improvements we created as part of this work.

The landscape of zero-knowledge proofs and arguments

As described in the last section, zero-knowledge proofs are a useful cryptographic primitive for verifiable yet confidential computing. Both the interactive [144, 146] and non-interactive [52, 105] variants of zero-knowledge (ZK) proofs (respectively, arguments) allow an unbounded (resp., polynomially-bounded) prover $\mathcal{P}$ to convince a verifier $\mathcal{V}$ that a relation C is satisfiable while hiding the witness to this fact. Our work in §4 focuses on ZK arguments.

There have been substantial advances over the past decade to improve the efficiency of ZK arguments along several metrics. We categorize these advances into three groups based on their tradeoffs between proof size (or total communication for interactive protocols), RAM requirements, and whether the proofs are verifiable to the general public or a single designated verifier.

First, ZK-SNARKs and ZK-STARKs offer sublinear proof size and verification time (between logarithmic and square root of the circuit size $|C|$) but require the prover to use enormous amounts of memory. There is a long line of research into ZK succinct interactive arguments of knowledge (SNARKs), building upon the work of Killian [197]. Initial constructions required superlinear prover time and per-circuit structured setup [34, 37, 44, 55, 94, 101, 136, 152, 153, 214, 243], and subsequent work achieved linear prover time and permitted universal structured setup [56, 80, 130, 131, 154, 218, 324]. The newest ZK-SNARKs and ZK scalable transparent arguments of knowledge (STARKs) leverage ideas from interactive oracle proofs [36, 252] or the sumcheck protocol [79, 215] to remove structured setup altogether but have slightly higher proof size [10, 35, 42, 64, 65, 267, 268, 312, 328].

Second, there exist ZK arguments that scale to large statements due to their moderate RAM requirements (approximately security parameter × circuit size) and linear prover and verifier runtime, but that sacrifice public verification because they need a designated verifier to maintain secret randomness. ZK proofs based on privacy-free garbled circuits [127, 133, 163, 182, 326] require a designated verifier to garble the circuit and keep the wire labels hidden until the end of the protocol. A separate line of research [24, 316] uses vector oblivious linear evaluation (VOLE) [60, 61, 232] to build proofs with a highly efficient (and optionally non-interactive) online phase, after a one-time interactive preprocessing phase is used to establish correlated randomness between the prover $\mathcal{P}$ and verifier $\mathcal{V}$.

Organization of this chapter

In §4.2-§4.6, we provide a concrete proof size improvement in the MPC-in-the-head paradigm of Ishai et al. [177]. In §4.7-§4.11, we describe a concrete proof size improvement based on Ligero [10].

4.2 Introduction to TurboIKOS

We begin with our improvement in the MPC-in-the-head paradigm. Our construction TurboIKOS, like most MPC-in-the-head proofs, has low RAM requirements, is publicly verifiable, has transparent setup, has linear prover and verifier runtime in the circuit size $|C|$, and can be made non-interactive in the Random Oracle Model via the Fiat-Shamir transform [120].

We present two variants of TurboIKOS. The first variant (§4.4.3), for large fields, improves Baum and Nof's [25] protocol based on sacrificing Beaver triples, and reduces the number of field elements sent per multiplication gate from 4 to 3. The second version (§4.4.4) reduces the number of field elements per multiplication gate further, from 3 to 2, and may be used in smaller fields.

4.2.1 The MPC-in-the-head paradigm

MPC-in-the-head is a method to construct a zero knowledge proof from a secure multiparty computation (MPC) protocol. Given an NP relation encoded as a circuit C, the prover $\mathcal{P}$ runs all parties in a secure computation of C beginning with a sharing of the witness, and the verifier $\mathcal{V}$ challenges $\mathcal{P}$ to open some of the views. Zero knowledge follows from the privacy of the MPC protocol, and soundness is achieved because a malicious $\mathcal{P}$ must have created inconsistent views and $\mathcal{V}$ finds them with noticeable probability. The seminal work of Ishai et al. [177] demonstrated that this transformation works for any MPC protocol. Subsequently, a line of works designed specific protocols with increasingly better proof size: ZKBoo [138], ZKB++ [76], Katz et al. [188], and Baum-Nof [25].

Table 4.1 shows proof sizes for MPC-in-the-head constructions in which the prover $\mathcal{P}$ runs R iterations of an MPC protocol, each of which involves N parties securely evaluating a circuit C with I input wires, O output wires, and M multiplication gates. When using an ordinary MPC protocol like SPDZ [100], a multiplication gate requires all parties to broadcast one message that is stored in the resulting proof, yielding in a proof size of $\Omega(MNR)$. To do better, MPC-in-the-head constructions make optimizations that are not acceptable for "normal" MPC protocols: they design *circuit decompositions* that look like MPC party views, yet can only be computed when a single entity $\mathcal{P}$ knows the inputs of all MPC parties. In circuit decompositions, the emulated MPC parties don't communicate to compute the views, but rather only to check their consistency.

We briefly survey the main ideas in each construction and the impact they have on the proof size per multiplication gate, which tends to be the largest contributor to the proof size.

- ZKBoo [138] and ZKB++ [76] are based on the $N = 3$ party replicated secret

Protocol	**Proof size**	**Soundness error**
IKOS+SPDZ [100, 178]	$R \cdot (6MN + (I + O)N)$	$(1/N)^R$
ZKBoo [138]	$R \cdot (2M + 2I + 2O)$	$(2/3)^R$
ZKB++ [76]	$R \cdot (M + I)$	$(2/3)^R$
Katz et al. [188]	$R \cdot (2M + I + \log N + \log_R(P))$	$\max_{0 \le i \le R} \frac{\binom{P-R+i}{P-R}}{\binom{P}{P-R} N^i}$
Baum-Nof [25]	$R \cdot (4M + I + \log N)$	$(1/N)^R$
$\Pi_{\mathit{TurboIKOS}}$ *(this work)*	$R \cdot (3M + I + \log N)$	$(1/N)^R$
$\tilde{\Pi}_{\mathit{TurboIKOS}}$ *(this work)*	$R \cdot (2M + I + \log N + NU)$	See Theorem 4.5.0.3

Table 4.1: *Proof size (in # of field elements) and soundness error (for large fields) for several MPC-in-the-head protocols. Some lower-order terms are omitted for legibility. N is the number of parties, M is the circuit size (number of multiplication gates), I and O are the number of input and output wires for the circuit, respectively, and R is the number of times the protocol is repeated. Note that ZKBoo and ZKB++ are only constructed for $N = 3$. P is a parameter specific to [188] indicating how many Beaver triples are committed to in advance. U is a new parameter introduced in §4.4 of this work.*

sharing MPC protocol of Araki et al. [13]; they do not generalize to arbitrary choices of N. All data is secret shared using 3-out-of-3 additive sharing, and addition can be done locally. Multiplication requires sending 3 messages, each of which is a function of a different subset of 2 of the 3 shares of the input wires. The verifier $\mathcal{V}$ receives two shares, and therefore can verify 1 of the 3 messages sent during each multiplication.

- ZKB++ and all subsequent works sample shares pseudorandomly. Given a seed σ_p for each party p, to share a value v_w on wire w, only the *offset* $e_w = v_w + \sum_p \mathsf{PRF}(\sigma_p, w)$ is recorded in the proof, reducing the cost per multiplication gate but requiring a (cheap) initial setup to distribute seeds.

- Katz et al. [188] extends MPC-in-the-head to accommodate MPC protocols with preprocessing. They build Beaver triples using a cut-and-choose approach, where some triples are opened and checked during preprocessing. The proof size ($R \log_R(P)$) required to assist $\mathcal{V}$ in the preprocessing step is independent of the circuit size. The remaining Beaver triples are assumed to be valid and used to verify the real execution.

- Baum-Nof [25] also uses pseudorandom shares and Beaver triples in a variant of the SPDZ MPC protocol, but avoids cut-and-choose in favor of *sacrificing* one Beaver triple to check the validity of each multiplication gate.

For each multiplication gate: ZKB++ requires 1 field element to represent the offset e_w for the output value (but requires more repetitions than the rest), Katz et al. requires 1 more field element to represent the offset for the Beaver triple value, and Baum-Nof requires 2 more field elements to test whether the sacrificed Beaver triple and the circuit values are consistent. In this work, we introduce two new sacrificing-based MPC-in-the-head constructions that require 1 and then 0 field elements to perform this consistency test; the latter introduces an additive overhead that can be smaller than that of Katz et al. for some parameter settings. See Table 4.1 for more details about the proof size for each protocol.

4.2.2 Overview of our construction

The simplest way to describe our first protocol variant is that we combine the techniques used in the Baum-Nof ZK proof with the Turbospeedz MPC protocol [31] so that sacrificing a Beaver triple costs only one field element instead of two, while preserving the soundness error. Our second variant replaces the remaining field element per multiplication gate with some prover advice about the overall circuit, reducing the proof size so that it is competitive with Katz et al. [188] but with a different set of parameter tradeoffs. In this section, we briefly describe the Turbospeedz construction and explain the challenge when integrating it into MPC-in-the-head.

SPDZ and Turbospeedz. The SPDZ line of works [38, 100, 237] is a popular family of MPC protocols that offloads the (expensive) generation of Beaver triples into a preprocessing phase so that the online phase has free additions and only requires broadcasting 2 elements per multiplication gate (1 per input wire). Turbospeedz [31] saves 1 element per multiplication gate by exploiting a redundancy: when generating shares of an input wire w pseudorandomly such that the shares of the value are $[v_w] = e_w + [\lambda_w]$, the public offsets e_w can also serve "for free" as the broadcast values for the input wires, and the only effort required is to create the new offset for the 1 output wire.

The challenge of TurboIKOS. When SPDZ is used in MPC-in-the-head to check a multiplication gate whose input and output wires are claimed to be a Beaver triple $\langle v_x, v_y, v_z \rangle$, it suffices to use the semi-honest protocol without MAC checks, and for the prover $\mathcal{P}$ to cheaply generate an independent Beaver triple $\langle \hat{\lambda}_x, \hat{\lambda}_y, \hat{v}_z \rangle$. However, with Turbospeedz there is a problem: the protocol transmits 2 field elements in the preprocessing stage, in addition to the 1 field element in the online stage. This is fine from an MPC perspective where preprocessing work might be viewed as "free," but is unacceptable for MPC-in-the-head where all elements add equally to the proof size.

To overcome this issue, we turn to another member of the SPDZ family: Overdrive [190]. The Overdrive protocol includes a clever method for generating a partially-correlated Beaver triple $\langle \lambda_x, \hat{\lambda}_y, \hat{v}_z \rangle$ where the shares $[\lambda_x]$ for the first element of the Beaver triple are the *same* as the shares for the true value v_x. With a common element between the two Beaver triples, all of the setup calculations become linear steps that can be computed locally by the parties. Integrating Turbospeedz's function-dependent preprocessing with Overdrive's Beaver triple generation mechanism is one of the accomplishments of our TurboIKOS protocol.

Implementing Picnic digital signatures. We provide an open source implementation of our protocol [156] and evaluate our proof size when using a variant of the

Picnic post-quantum digital signature scheme [224] that uses AES as its block cipher, following the techniques introduced by BBQ [104]. Picnic signatures are based on an MPC-in-the-head proof of knowledge of a secret key k such that $\mathrm{AES}_k(x) = y$, where the corresponding public key is (x, y). As we show in §4.6.1, our protocol returns the smallest proof size among streaming- and memory-friendly systems using less than 32 emulated MPC parties. Our signature sizes are also competitive with those of Banquet [23], an independent recent work that involves a memory-intensive polynomial interpolation over the entire circuit.

4.3 Preliminaries

4.3.1 Notation

Throughout this work, $\mathcal{P}$ denotes the prover and $\mathcal{V}$ denotes the verifier. We let C denote an arithmetic circuit corresponding to an NP relation with a canonical output message corresponding to logical true (i.e., the witness satisfies the relation). We use ADD, MUL to denote addition and multiplication gates, respectively.

The circuit has a set of gates G of which a subset M are MUL gates, as well as a set W of wires, of which there are subsets I of inputs to the circuit and outputs of MUL gates. $O \subseteq W$ denotes the output wires for the circuit. By abuse of notation, we use the same variables to denote the size of each set; for instance, we let M denote the number of multiplication gates when it is clear from context that we are describing an integer rather than a set.

We consider an MPC-in-the-head protocol execution with N parties that is repeated R times. If a single iteration of a protocol has soundness error δ, then we can run $R = \lceil \frac{\kappa}{\log(1/\delta)} \rceil$ independent iterations to reduce the soundness error to $2^{-\kappa}$ (where all logarithms are taken base-2 in this work).

For computation and equations, we use $\mathbb{F}$ to refer to a finite field and $\mathbb{F}^*$ to refer to the units of that field. We generally use κ as our security parameter and $[v]$ to refer to an additive secret sharing of a value v among the N parties.

We say a party is p.p.t. to denote that it is probabilistic polynomial time.

4.3.2 Definitions

Pseudorandom Functions and Commitments. We require the existence of a pseudorandom function PRF and a computationally hiding commitment scheme Com in our security analysis in §4.5. Our implementation uses hash-based commitments that model the hash function as a random oracle and assume that AES acts as a PRF. Below we give the formal definitions for PRF and Com:

Definition 4.3.2.1 (Pseudorandom Function)**.** Let $\mathcal{F}$: $\{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*$ be an efficient, length-preserving, and keyed function. $\mathcal{F}$ is a pseudorandom function with soundness κ if for all adversaries A that run in at most q time steps, A's advantage $\mathrm{Adv}_{\mathsf{PRF}}(\mathsf{A}) = |\Pr_k[\mathsf{A}^{\mathcal{F}(k,-)} = 1] - \Pr_H[\mathsf{A}^H = 1]|$ at distinguishing the pseudorandom function from a random oracle H is at most $q/2^\kappa$.

Definition 4.3.2.2 (Commitment)**.** A commitment scheme is a protocol between two parties $\mathcal{S}$ and $\mathcal{R}$ with the following algorithms:

- $\mathsf{Com}(m; r)$: The sender $\mathcal{S}$ has an input message $m \in \{0,1\}^*$ and security parameter 1^n. The algorithm Commit outputs a public commitment c to the receiver $\mathcal{R}$ and stores the private randomness r.
- $\mathsf{Decom}(c, m, r)$: the sender $\mathcal{S}$ sends (c, m, r) to the receiver $\mathcal{R}$, who then either accepts and outputs m or rejects.

A computationally secure commitment scheme satisfies the following properties:

- **Completeness:** If $(c, r) = \mathsf{Com}(m; r)$, then in $\mathsf{Decom}(c, m, r)$ the receiver $\mathcal{R}$ accepts and outputs m.
- (Computational) **Hiding:** For any two message pairs $m, m' \in \{0,1\}^*$, any receiver $\mathcal{R}^*$ running in q time cannot distinguish their respective commitments $\mathrm{Adv}_{\mathsf{Com}}(\mathcal{R}^*) = |\Pr[\mathcal{R}^*(\mathsf{Com}(m; r)) = 1] - \Pr[\mathcal{R}^*(\mathsf{Com}(m'; r')) = 1]|$ except with probability at most $q/2^\kappa$.

- (Computational) **Binding:** No adversarial sender $\mathcal{S}^*$ running in at most q time has more than probability $q/2^\kappa$ of outputting c, m, m', r, r' such that $m \neq m'$, and $\mathsf{Decom}(c, m, r)$ and $\mathsf{Decom}(c, m', r')$ both accept.

While our main construction can support arbitrary commitment schemes, in this work we focus on the hash-based commitment scheme in the random oracle model, in which $\mathsf{Com}(m; r) = H(m, r)$ feeds the input message and randomness into the random oracle and $\mathsf{Decom}(c, m, r) = (m, r)$ provides the preimage to the hash. The binding of this scheme follows from a birthday bound analysis: if a random oracle has 2κ bit output length and an adversary makes at most q queries to this oracle, then the probability that the adversary finds a collision in the oracle is at most $q^2/2^{2\kappa}$, and a collision is necessary to break the binding property of the commitment scheme. The hiding property can be proved similarly.

There are a few optimizations that prior works have used here to save space. First, when committing to a list of messages $\langle m_1, m_2, \ldots, m_\ell \rangle$, the sender can provide a succinct commitment $H(\mathsf{Com}(m_1, r_1), \ldots, \mathsf{Com}(m_\ell, r_\ell))$ to the entire list, again thanks to collision resistance. Second, if m is already known to the receiver, then it suffices to send only r during decommitment. Third and most ambitiously, because we will only commit to strings that already have min-entropy κ, when generating a signature scheme we can go further and remove the randomness r from the Com and Decom algorithms to create a deterministic scheme in which decommitments are free when m is already known to the receiver. This strategy breaks the hiding property of the commitment and thus the zero knowledge property of the schemes we will construct, but it will suffice for our signature construction; we refer readers to Katz et al. [188, §3.1] for details.

Honest Verifier Zero-knowledge Argument of Knowledge. Next, we formally define the notion of ZK arguments over an NP-relation $\mathcal{R}(x, w)$ as a two-party protocol involving two p.p.t. algorithms, a prover $\mathcal{P}$ and a verifier $\mathcal{V}$. Both parties have the same NP statement x, and only the prover receives its corresponding witness w. The parties interact to determine whether $\mathcal{R}(x, w) = 1$ without revealing the witness. We restrict our attention to the honest verifier setting in which $\mathcal{V}$ never deviates from the protocol.

Definition 4.3.2.3. The protocol ($\mathcal{P}$, $\mathcal{V}$) is an honest verifier ZK argument for the relation $\mathcal{R}(x, w)$ if it satisfies the following properties:

- **Completeness:** If $\mathcal{P}$ and $\mathcal{V}$ are honest and $\mathcal{R}(x, w) = 1$, $\mathcal{V}$ always accepts.
- **Soundness:** For any malicious and computationally bounded prover $\mathcal{P}^*$, there is a negligible function $\mathsf{negl}(\cdot)$ such that a statement x is not in the language (i.e., $\mathcal{R}(x, w) = 0$ for all w), then $\mathcal{V}$ rejects on x with probability $\geq 1 - \mathsf{negl}(|x|)$ when interacting with $\mathcal{P}^*$.
- **Honest verifier computational zero knowledge:** Let $View_{\mathcal{V}(x,w)}$ be a random variable describing the distribution of messages received by $\mathcal{V}(x)$ from $\mathcal{P}(x, w)$. Then, there exists a p.p.t. simulator Sim such that for all x in the language, the distributions of $\mathsf{Sim}(x)$ and $\mathsf{View}_{\mathcal{V}(x,w)}$ are computationally indistinguishable.

In this work, we will construct a ZK argument of knowledge, which provides a stronger *knowledge soundness* guarantee that if a bounded-time malicious prover $\mathcal{P}^*$ can make the verifier accept a statement x with non-negligible probability, then there exists an extractor $E^{\mathcal{P}^*}(x)$ that can output a witness w such that the relation holds $\mathcal{R}(x, w) = 1$.

Additionally, we restrict our attention to honest verifier ZK in this work because our protocol TurboIKOS is also public coin and constant round, so it can be transformed into a non-interactive argument using the Fiat-Shamir transform.

Secure Multi Party Computation (MPC). An MPC protocol allows N players to jointly compute a function of their respective inputs while maintaining the privacy of their individual inputs and the correctness of the output. In addition, the protocol should prevent an adversary who may corrupt a subset of players, from learning additional information or harming the protocol execution. A party's *view* in MPC contains that party's input, randomness, and any messages received by that party. For use in MPC-in-the-head, secure computation protocols must satisfy t-privacy, meaning that the view of any subset $t < N$ of the parties can be simulated (see [177] for a formal definition).

4.4 Construction

In this section we describe the two versions of our protocol. $\Pi_{\text{TurboIKOS}}$ has size roughly $3M$ and is described in Figures 4·1 and 4·2. $\tilde{\Pi}_{\text{TurboIKOS}}$ has size roughly $2M$ and is described in Figures 4·1 and 4·4. We start by describing the Baum-Nof [25] SPDZ-like protocol and the Turbospeedz MPC protocol [31]. Then, we show how to incorporate Turbospeedz into the MPC-in-the-head paradigm to reduce the amount of communication per MUL gate, creating protocol $\Pi_{\text{TurboIKOS}}$. Last, we show how the prover can provide a different commitment to some of the proof components, removing another element per MUL gate and yielding protocol $\tilde{\Pi}_{\text{TurboIKOS}}$.

4.4.1 Starting point: SPDZ and Baum-Nof

We use the MPC-in-the-head paradigm introduced by Ishai et al. [177] combined with a semi-honest version of the $(N-1)$-private SPDZ MPC protocol [100] as a starting point for our zero-knowledge proof using MPC-in-the-head protocol. In IKOS, a prover simulates an MPC protocol for all parties and commits to a view for each party containing the party's randomness, input, and messages received. To save proof space, an additional "broadcast channel" is committed to for messages that are sent to all parties, rather than writing the same value in all party views. Then the verifier chooses a subset of the parties and challenges the prover to open the committed views of these parties. The verifier then confirms that the views of the opened parties are *consistent*, that is, the message party i sent to party j is the same in views of both those parties. For N-party MPC protocols that *only* send broadcast messages and do not contain any private messages between parties, the verifier opening T parties will have a $\frac{T}{N}$ chance of catching a prover who cheats by creating inconsistent views: the "receiving" half of the message is always revealed in the broadcast channel, and these are checked for consistency with the revealed parties' "sent" messages. By repeating this process R times with fresh randomness, the verifier can shrink the probability of error by a power of R.

We start with the variant of semi-honest SPDZ [100] used by Baum-Nof [25]. Let N denote the set of parties and M denote the set of multiplication gates in the circuit C. The parties hold sharings of the inputs $[x_m]$ and $[y_m]$ for each MUL gate $m \in M$; since this MPC protocol is being emulated by a prover who knows the value on the wire, the parties additionally have (and commit to) a sharing of the gate's output $[z_m]$. The prover generates a random multiplication triple, $\langle a_m, b_m, c_m \rangle$, which will be "sacrificed" to check a multiplication constraint in a MUL gate. The verifier will send a random challenge $\varepsilon_m \leftarrow \mathbb{F}$. Each party does the following:

1. Broadcast $[f_m] = \varepsilon_m[x_m] + [a_m]$ and $[g_m] = [y_m] + [b_m]$
2. Use the recombined f and g to compute

$$[\zeta_m] = \varepsilon_m[z_m] - f_m g_m + f_m[b_m] + g_m[a_m] - [c_m]. \tag{4.1}$$

Baum and Nof show that if either $\langle a_m, b_m, c_m \rangle$ or $\langle x_m, y_m, z_m \rangle$ is not a valid multiplication triple, this value ζ_m will be nonzero with probability at least $1 - 1/|\mathbb{F}|$ over the choice of ε_m. We will prove similar claims for different values, and linear combinations thereof, in Lemmas 4.4.3.1-4.4.3.2.

To save proof space, rather than broadcasting the ζ_m values for each MUL gate m, an additional challenge variable $\hat{\varepsilon}_m \in \mathbb{F}$ is sent by the verifier $\mathcal{V}$ and the prover $\mathcal{P}$ responds by sending a linear combination $[Z] = \sum_{m \in M} \hat{\varepsilon}_m [\zeta_m]$ of the secret values and public coefficients. If the prover is honest then $Z = 0$. Baum and Nof show (Proposition 1 of [25]) that Z will be nonzero if at least one MUL gate constraint is violated with probability at least $1 - 2/|\mathbb{F}|$. Later in Lemma 4.4.3.2 of this paper we will improve this bound for a batched set of ζ_m values to a $1/|\mathbb{F}|$ error using a very-slightly different batching technique.

If $1/|\mathbb{F}|$ does not yield sufficient soundness error, we can reduce this error by doing multiple batched checks. To do so, we reveal linear combinations $[Z_1], \ldots, [Z_U]$, all over the same $[\zeta_m]$ shares, but using different random $\hat{\varepsilon}_m$ choices provided by the verifier. Let U be the number of these checks.

Naively, this protocol broadcasts $(2M + U)N$ elements since each party broadcasts their f shares and g shares for each multiplication gate, plus $[Z_1], \ldots, [Z_U]$. Later in this work we will show multiple ways to reduce this size with different tradeoffs, by taking advantage of the fact that $\mathcal{V}$ has corrupted $N - 1$ parties. We emphasize that all parties' shares must still be committed to before $\mathcal{P}$ knows which party will remain uncorrupted.

To compress the parties' views, we can generate the shares of all values pseudorandomly, with only one public "offset" value per wire. Then, for each multiplication gate, the prover only needs to broadcast the offset values for f and g, along with the offsets of the true output wire z and the Beaver triple product c. Hence, the proof contains 4 field elements per multiplication gate, as shown in Table 4.1.

4.4.2 Introducing Turbospeedz

Turbospeedz [31] generally shows how to have only one broadcast per multiplication gate instead of two in normal SPDZ by adding a function-dependent preprocessing step where the circuit to be computed is known, but the input to the circuit need not yet be known. The idea is to add a sharing of a "mask" on each wire, propagated additively (but not multiplicatively) during preprocessing. Then, the masks of the input wires can serve as the first two elements of a Beaver triple, which is also generated during the preprocessing. Let x and y denote the input wire and z denote the output wire of any gate. Let v_x, v_y, v_z denote the real values on the wires. In the preprocessing phase, the prover performs the following:

1. For each party, the prover generates random "masking shares" $[\lambda_w]$ for each input wire and the output wire of each MUL, w.

2. The prover homomorphically computes the mask shares for each ADD gate internal output wire, $[\lambda_z] = [\lambda_x] + [\lambda_y]$.

3. For each wire w, the prover computes external value, $e_w = \lambda_w + v_w$. In MPC, these external values are public to all parties. In MPC-in-the-head, $\mathcal{P}$ will give them to $\mathcal{V}$ in the clear.

In Turbospeedz, given e_x, e_y, and a Beaver triple $\langle a_m, b_m, c_m \rangle$, each party computes their share of MUL gate m's output wire by locally computing

$$\begin{aligned}[v_z] &= e_x e_y - e_y[a_m] - e_x[b_m] + [c_m] \\ &= (v_x + a_m)(v_x + b_m) - (v_y + b_m)[a_m] - (v_y + a_m)[b_m] + [c_m] \\ &= [v_x v_y].\end{aligned}$$

The parties then proceed to compute and open $[e_z] = [v_z] + [\lambda_z]$. Note that this relies on the parties already possessing a sharing of a *valid* Beaver triple $\langle a_m, b_m, c_m \rangle$ in advance.

The upshot of this method is that multiplication gates can be computed using only one opening (e_z) instead of two (f and g in the previous section).

4.4.3 Adapting Turbospeedz into sacrificing-based MPC-in-the-head

Input: The prover $\mathcal{P}$ and verifier $\mathcal{V}$ receive an input circuit C comprising a set of gates G of which a subset M are MUL gates, along with a set of wires W with subsets of input and output wires I and O, respectively. $\mathcal{P}$ is the sole recipient of a witness. Both parties also receive constants R and N (the latter of which we equate with the set $\{1, \ldots, N\}$ by abuse of notation). The prover $\mathcal{P}$ and verifier $\mathcal{V}$ run R independent executions of the following protocol in parallel.

Function-dependent preprocessing: $\mathcal{P}$ pseudorandomly derives shares $[\lambda_w]$ for each wire $w \in W$ and a Beaver triple for each multiplication gate as follows.

1. Generate a random master seed σ^*. Pseudorandomly derive a binary tree with root σ^* until there are as many leaves as parties. Assign the p^{th} leaf as party key σ_p.
2. For each input wire $w \in I$ and party $p \in N$, pseudorandomly derive share $[\lambda_w]$ from key σ_p.
3. Go through C layer by layer, starting at the input layer. For every $g \in G$, do the following on gate C_g with input wires x, y and output wire z.
   - If C_g is an ADD gate: assign $[\lambda_z] \coloneqq [\lambda_x] + [\lambda_y]$.
   - If C_g is a MUL gate:
     - Derive $[\lambda_z]$, $[\hat{\lambda}_{y,g}]$, and $[\hat{\lambda}_z]$ for every $p \in N$ from σ_p. (Note that y can be an input to many MUL gates, hence the two indices in $[\hat{\lambda}_{y,g}]$.)
     - Set $\hat{e}_z \coloneqq \lambda_x \cdot \hat{\lambda}_{y,g} + \hat{\lambda}_z$, which creates a Beaver triple $\langle \lambda_x, \hat{\lambda}_{y,g}, \hat{e}_z - \hat{\lambda}_z \rangle$.

Interactive phase: Once $\mathcal{P}$ receives the witness, the parties interact as follows.

1. $(\mathcal{P} \to \mathcal{V})$ $\mathcal{P}$ executes the circuit to determine the value v_w of each wire $w \in W$, and assigns the offset $e_w \coloneqq v_w + \lambda_w$ for each wire $w \in W$. It sends to $\mathcal{V}$:
   - Offsets e_w for input wires $w \in I$, and offsets e_z and $\hat{e}_z$ for the outputs of MUL gates. (The remaining offsets can then be computed.)
   - A commitment to all shares $[\lambda_w]$ for all output wires $w \in O$.
   - A commitment to the seeds σ_p for all parties $p \in N$.
2. $(\mathcal{V} \to \mathcal{P})$ $\mathcal{V}$ randomly selects two elements $\varepsilon_m, \hat{\varepsilon}_m \leftarrow \mathbb{F}$ for every MUL gate $m \in M$.

[There are two ways to complete this protocol, shown in Figures 4·2 and 4·4.]

Figure 4·1: *Beginning of an interactive zero knowledge protocol between prover $\mathcal{P}$ and honest verifier $\mathcal{V}$, given a relation represented as an arithmetic circuit with ADD and MUL gates over a field $\mathbb{F}$. All verifier messages are public coins, so the protocol can be made non-interactive using the Fiat-Shamir transform. There are two different endings to this protocol, given in Figures 4·2 and 4·4.*

In this section, we incorporate a modified version of the Turbospeedz method from Section 4.4.2 into the SPDZ-based MPC-in-the-head framework described above from Section 4.4.1. For large field sizes, the resulting MPC-in-the-head protocol $\Pi_{\text{TurboIKOS}}$ will require only 3 field elements per multiplication, rather than the 4 elements used

Interactive phase, continued from Fig. 4·1:

3. $(\mathcal{P} \to \mathcal{V})$ $\mathcal{P}$ sends all α_m values and commits to $\mathcal{P}$ commits to all $[\alpha_m]$ and $[Z]$ shares. These variables are computed as follows.
   - For every MUL gate $m \in M$, assign $[\alpha_m] := \varepsilon_m[\lambda_y] + \hat{\varepsilon}_m[\hat{\lambda}_{y,m}]$.
   - For every party $p \in N$, assign $[Z] := \sum_{m \in M}[\zeta_m]$, which is the sum of all $[\zeta_m] := \varepsilon_m e_z - \varepsilon_m e_x e_y + \hat{\varepsilon}_m \hat{e}_z + (\varepsilon_m e_y - \alpha_m)[\lambda_x] + \varepsilon_m e_x[\lambda_y] - \varepsilon_m[\lambda_z] - \hat{\varepsilon}_m[\hat{\lambda}_z]$.
4. $(\mathcal{V} \to \mathcal{P})$ $\mathcal{V}$ randomly selects a set $T = N \setminus \{i^*\}$ of $N-1$ parties to corrupt.
5. $(\mathcal{P} \to \mathcal{V})$ $\mathcal{P}$ reveals the $\log(N)$ seeds from preprocessing step 1 that suffice for $\mathcal{V}$ to recompute σ_p for all corrupted parties $p \in T$, but not the remaining seed σ_{i^*}.

Verification. $\mathcal{V}$ accepts only if all of the following are true:

- The output values (v_w for $w \in O$) provided by $\mathcal{P}$ correspond to logical true.
- The commitments in rounds 1 and 3 are consistent with the opened keys σ_p for corrupted parties, and with the shares of $[Z]$, $[\alpha_m]$, and $[\lambda_w]$ for output wires for *all* parties. $\mathcal{V}$ can compute $N-1$ shares of these from its seeds, and the remaining shares from the revealed α_m values and the known values for Z and output wires.

Figure 4·2: *End of the interactive zero knowledge protocol $\Pi_{TurboIKOS}$ between prover $\mathcal{P}$ and honest verifier $\mathcal{V}$, given a relation represented as an arithmetic circuit with ADD and MUL gates over a field $\mathbb{F}$. See Figure 4·1 for the beginning of this protocol.*

in Baum-Nof [25].

Committing to all wire values. The first step of converting Turbospeedz into an MPC-in-the-head protocol is to replace the step of opening shares of e_z by having the $\mathcal{P}$ simply provide e_z in the clear. However, unlike Turbospeedz, we do not wish to have a costly preprocessing process in which the verifier becomes convinced of the validity of all Beaver triples in the circuit; instead we wish to use ζ_m values as in Eq. 4.1. In order to do this without reusing masks on multiple values, we must make some subtle changes to the original Turbospeedz protocol.

To save space, most shares held by an emulated party will be pseudorandomly derived from a party-specific seed. To generate the party-specific seeds themselves, we follow the same method as Katz et al. [188], namely, to derive the party seeds pseudorandomly from a master seed via a binary tree with the parties' seeds at the leaves. This reduces the cost of sending the seeds from N to $\log N$.

$\mathcal{P}$ will generate all λ_w values for all wires in the circuit the same as in original Turbospeedz, but by generating each party's share pseudorandomly using the party-specific seed. We take advantage of the external value e as an offset: the value on wire w is defined as simply $v_w = e_w - \lambda_w$. Additionally, for each MUL gate m, the prover will generate additional pseudorandom shares $[\hat{\lambda}_{y,m}]$ and $[\hat{\lambda}_z]$. $\mathcal{P}$ computes $\hat{e}_z = \lambda_x \hat{\lambda}_{y,m} + \hat{\lambda}_z$, forming a correlated Beaver triple $\langle \lambda_x, \hat{\lambda}_{y,m}, \hat{e}_z - \hat{\lambda}_z \rangle$ that will be sacrificed. The double index on $[\hat{\lambda}_{y,m}]$ is due to the fact that wire y may be reused in several different MUL gates m, each of which must define their own Beaver triple for the prover's privacy. (For legibility, we sometimes omit this double-subscript when the gate under consideration is clear from context.)

Creating a test for consistency of all gates. The largest change is in how ζ_m is calculated for each MUL gate m. We begin similarly to the Baum-Nof challenge: $\mathcal{V}$ will send random challenges $\varepsilon_m, \hat{\varepsilon}_m \leftarrow \mathbb{F}$. Our α_m values are defined slightly differently, for a reason we will explain shortly. The prover will send $\alpha_m = \varepsilon_m \lambda_y + \hat{\varepsilon}_m \hat{\lambda}_y$. Then, the parties compute:

$$[\zeta_m] = \varepsilon_m e_z - \varepsilon_m e_x e_y + \hat{\varepsilon}_m \hat{e}_z + (\varepsilon_m e_y - \alpha_m)[\lambda_x] + \varepsilon_m e_x [\lambda_y] - \varepsilon_m [\lambda_z] - \hat{\varepsilon}_m [\hat{\lambda}_z]. \quad (4.2)$$

First, we wish to show that this ζ_m serves a similar purpose to Baum-Nof's, assuming (for the moment) that the prover $\mathcal{P}$ honestly computes all α_m values from the parties' shares. For each MUL gate $m \in M$, define:

$$\Delta_{z,m} = (e_z - \lambda_z) - (e_x - \lambda_x)(e_y - \lambda_y) \text{ and } \hat{\Delta}_{z,m} = (\hat{e}_z - \hat{\lambda}_z) - \lambda_x \hat{\lambda}_y.$$

Observe that if $\mathcal{P}$ is honest, then $\langle e_x - \lambda_x, e_y - \lambda_y, e_z - \lambda_z \rangle$ and $\langle \lambda_x, \hat{\lambda}_y, \hat{e}_z - \hat{\lambda}_z \rangle$ are both valid Beaver triples and therefore $\Delta_{z,m} = \hat{\Delta}_{z,m} = 0$.

Lemma 4.4.3.1. *Fix a MUL gate $m \in M$. If ε_m and $\hat{\varepsilon}_m$ are chosen uniformly randomly from $\mathbb{F}$, and if either $\Delta_{z,m} \neq 0$ or $\hat{\Delta}_{z,m} \neq 0$ (or both), then $\zeta_m \neq 0$ with* *probability at least* $1 - 1/|\mathbb{F}|$.

Proof. Observe that

$$\begin{aligned}
\zeta_m &= \varepsilon_m e_z - \varepsilon_m e_x e_y + \hat{\varepsilon}_m \hat{e}_z + (\varepsilon_m e_y - \alpha_m)\lambda_x + \varepsilon_m e_x \lambda_y - \varepsilon_m \lambda_z - \hat{\varepsilon}_m \hat{\lambda}_z \\
&= \varepsilon_m e_z - \varepsilon_m e_x e_y + \hat{\varepsilon}_m(\lambda_x \hat{\lambda}_y + \hat{\Delta}_{z,m}) + (\varepsilon_m e_y - \alpha_m)\lambda_x + \varepsilon_m e_x \lambda_y - \varepsilon_m \lambda_z \\
&= \varepsilon_m e_z - \varepsilon_m e_x e_y + \hat{\varepsilon}_m \hat{\Delta}_{z,m} + (\varepsilon_m e_y - \varepsilon_m \lambda_y)\lambda_x + \varepsilon_m e_x \lambda_y - \varepsilon_m \lambda_z \\
&= \varepsilon_m e_z - \varepsilon_m \lambda_z - \varepsilon_m e_x e_y + \varepsilon_m e_x \lambda_y + \varepsilon_m e_y \lambda_x - \varepsilon_m \lambda_y \lambda_x + \hat{\varepsilon}_m \hat{\Delta}_{z,m} \\
&= \varepsilon_m((e_z - \lambda_z) - (e_x - \lambda_x)(e_y - \lambda_y)) + \hat{\varepsilon}_m \hat{\Delta}_{z,m} \\
&= \varepsilon_m \Delta_{z,m} + \hat{\varepsilon}_m \hat{\Delta}_{z,m}.
\end{aligned}$$

Now, consider the probability that $\zeta_m = 0$ over the uniform choice of ε_m and $\hat{\varepsilon}_m$ from $\mathbb{F}$. The only way for this to occur is if $\varepsilon_m \Delta_{z,m} = -\hat{\varepsilon}_m \hat{\Delta}_{z,m}$. If $\Delta_{z,m} = 0$, this happens if and only if $\hat{\varepsilon}_m = 0$, which occurs with probability $1/|\mathbb{F}|$. If $\Delta_{z,m} \neq 0$, then for any choice of $\hat{\varepsilon}_m$ there exists a single option for $\varepsilon_m = -\hat{\varepsilon}_m \hat{\Delta}_{z,m} \Delta_{z,m}^{-1}$ that makes $\zeta_m = 0$, so again we arrive at a probability of $1/|\mathbb{F}|$. These two cases are mutually exclusive, which yields the desired bound. □

Similar to Baum-Nof, we check the consistency of all these values with one random linear combination Z to test the values in Eq. (4.2) for all multiplication gates at once. However, because we defined ζ_m to already include two different random coefficients on the different Δ values, these coefficients already suffice to serve as challenge coefficients for this linear combination. As we show in Lemma 4.4.3.2, the upshot is that we can test all gates in the circuit with a soundness error of only $1/|\mathbb{F}|$ by merely revealing $[Z] = \sum_{m \in M} [\zeta_m]$.

Lemma 4.4.3.2. *If ε_m and $\hat{\varepsilon}_m$ are chosen uniformly randomly from $\mathbb{F}$ for all multiplication gates in the circuit, and if there exists at least one* MUL *gate $\bar{m} \in M$ such that $\Delta_{z,m} \neq 0$ or $\hat{\Delta}_{z,m} \neq 0$, then $Z \neq 0$ with probability at least $1 - 1/|\mathbb{F}|$.*

Proof. Consider $Z = \sum_{m \in M} \zeta_m = (\varepsilon_{\bar{m}} \Delta_{z,\bar{m}} + \hat{\varepsilon}_{\bar{m}} \hat{\Delta}_{z,\bar{m}}) + Z'$, where Z' is the sum of all other terms in the formula and $\bar{m} \in M$ is the gate where the sum is guaranteed to be nonzero; without loss of generality, suppose that $\Delta_{z,\bar{m}} \neq 0$. Then, $Z = 0$ if and only if $\varepsilon_{\bar{m}} = \Delta_{z,\bar{m}}^{-1} \cdot (-Z' - \hat{\varepsilon}_{\bar{m}} \hat{\Delta}_{z,\bar{m}})$, which occurs with probability $1/|\mathbb{F}|$. □

Completing the consistency test. Rounds 3-5 of the protocol provide a method for the verifier to check whether $Z = 0$, up to $1/N$ soundness error. We will describe two ways to perform this task: a base protocol $\Pi_{\text{TurboIKOS}}$ described in this section (shown also in Figure 4·2) and an improved protocol $\tilde{\Pi}_{\text{TurboIKOS}}$ in Section 4.4.4. Both techniques involve providing some 'advice' in the form of the non-privacy-sensitive α_m value for each MUL gate that assists the verifier in its computation of Z.

In round 3 of the base protocol $\Pi_{\text{TurboIKOS}}$ in this section, the prover provides for each MUL gate $m \in M$. Importantly, the prover also commits to all shares $[\alpha_m] = \varepsilon_m[\lambda_y] + \hat{\varepsilon}_m[\hat{\lambda}_y]$, and analogously for all $[Z]$ shares. There are three claims that the verifier must check:

- The committed $[\alpha_m]$ and $[Z]$ are consistent with the parties' individual views, at least for the $N-1$ emulated parties that the verifier can open.
- The committed $[\alpha_m]$ shares in round 3 collectively sum to the provided α_m value. That is, the prover provided the public α_m 'advice' value correctly.
- Assuming the advice is correct, then the $[Z]$ shares committed in round 3 sum to $Z = 0$. That is, the prover passes the test posed in Lemma 4.4.3.2.

After the prover reveals seeds for $N-1$ parties in round 5, the verifier can check these claims as follows. First, $\mathcal{V}$ can compute the remaining party's $[\alpha_m]$ by subtracting the known shares from the public α_m value, and then check whether these shares together constitute a valid opening of the commitment in round 3. This checks (most of) the first two claims simultaneously. The final claim is verified similarly; the key observation here is that if the prover is honest, then the value $Z = 0$ is publicly known. So, the $\mathcal{V}$ computes $N-1$ shares of Z, calculates what the remaining party's share must be in order for the overall value $Z = 0$ as required, and then checks whether these shares together constitute a valid opening of the commitment.

Putting it all together. Our protocol $\Pi_{\text{TurboIKOS}}$ is described in detail in Figures 4·1-4·2. The prover $\mathcal{P}$ and verifier $\mathcal{V}$ interact in a 5-round protocol, and if all consistency checks pass then the verifier believes that the output wire labels derived from the circuit evaluation is correct.

Completeness is a straightforward consequence of the fact that the honest prover computes the desired circuit (many times, in fact). We prove the privacy and knowledge soundness of our ZK argument of knowledge in Section 4.5.

Compared to Baum-Nof [25], we reduce communication per multiplication gate from 4 to 3 field elements. Concretely, for each multiplication gate, Baum-Nof must send the f and g values described in §4.4.1. Their protocol must also send a Beaver triple offset (analogous to $\hat{e}_z$) as well as the offset for the output wire of the MUL gate (similar to e_z). By using the Turbospeedz approach, we reduce communication to only 3 field elements: e_z, $\hat{e}_z$, and α_m.

Algorithmic optimizations. There are a few optimizations that we can apply to the base protocol $\Pi_{\text{TurboIKOS}}$ to save space even further. Some of these optimizations are deliberately omitted from Figures 4·1 and 4·2 for brevity; they are simple to add, and they are built into our implementation described in §4.6.

Our first optimization saves on the cost of commitments. Recall that we need to commit to values in each of the prover steps (rounds 1, 3, and 5), and also that the entire procedure from Figs. 4·1-4·2 is repeated R times. It suffices to build a single commitment per round across all repetitions: that is, just 2 commitments in total for the entire proof.

Second, we described the SPDZ-style MPC protocol by considering pseudorandom values for each party plus a public offset. Following prior works, we save space by integrating the offset into a single party's value (say, party 1). While this party no longer has pseudorandom value, the upshot is that we only need to reveal the $\hat{e}_z$

values within party 1's view, or in other words we don't need to reveal these values for the $1/N$ fraction of repetitions in which party 1 is the unopened party. (Note that we still need to publish the e_z and α_m values on all repetitions because $\mathcal{V}$ needs this information to perform its consistency check.)

Third, if the circuit has a single known value that represents 'logical true' (say, the value 0), then we can save on the cost of opening the output wire shares $[\lambda_w]$ for all parties (i.e., including the unopened party). Instead, we can follow a similar trick as we described above for $[\alpha_m]$ and $[Z]$. In round 1 of the protocol, the prover $\mathcal{P}$ commits to all output wire mask (λ) shares. Once the verifier $\mathcal{V}$ learns the seeds to reconstruct $N - 1$ of these shares for itself, it assumes that the output wires collectively reconstruct to logical true and calculates the remaining share accordingly as $[\lambda]_{i^*} = e - v_{\text{expected}} - \sum_{i \neq i^*} [\lambda]_i$, where v_{expected} is the expected value on the wire. Finally, $\mathcal{V}$ checks that all shares match $\mathcal{P}$'s commitment.

4.4.4 Constructing smaller consistency tests

In this section, we describe an improved protocol $\tilde{\Pi}_{\text{TurboIKOS}}$ that reduces the cost per multiplication gate from 3 field elements down to 2. Specifically, we show a new method to check the consistency of Z in rounds 3-5 without revealing an α_m element for each MUL gate. The motivation for this change is twofold. The first reason is obvious: reducing the number of field gates required per MUL gate shrinks the proof size. The second and more subtle reason is to improve the performance of TurboIKOS on smaller fields, such as the field GF(256) used in AES, without blowing up the size of the protocol with additional zero-checks.

We will explain this second motivation at a high level here; for more detail see the soundness analysis (Theorems 4.5.0.2 and 4.5.0.3) in §4.5. A cheating prover must have a sufficiently low chance of getting a set of coefficients $\varepsilon_m, \hat{\varepsilon}_m$ where $Z = 0$ even though at least one MUL constraint is violated. For small fields, a malicious prover has a decently-high probability of passing a single zero-test Z by pure chance (one in the size of the field). While one could overcome this issue by increasing the number of repetitions R, there is an alternative solution: run several $Z = 0$ tests using fresh random ε_m and $\hat{\varepsilon}_m$ coefficients for each, but the same wire shares λ_z and offsets e_z and $\hat{e}_z$. Concretely, we create a new parameter U denoting the number of checks to run.

This alternative method doesn't fare well in the original protocol $\Pi_{\text{TurboIKOS}}$ because our method requires revealing an α_m value per MUL gate for each test, so each additional Z value used to improve the soundness error would reveal an additional field element per MUL gate, making the proof size much larger. Thus, our goal in this section is to show how to consistency-check multiple zero-tests $Z_1, \ldots, Z_U$ without transmitting additional information proportional to the number of MUL gates beyond the two field elements e_z and $\hat{e}_z$ we already transmitted in round 1.

Recall that for a single MUL gate m,

$$[\zeta_m] = \varepsilon_m e_z - \varepsilon_m e_x e_y + \hat{\varepsilon}_m \hat{e}_z + \varepsilon_m e_y[\lambda_x] - \alpha_m[\lambda_x] + \varepsilon_m e_x[\lambda_y] - \varepsilon_m[\lambda_z] - \hat{\varepsilon}_m[\hat{\lambda}_z]$$

is a sharing of $\zeta_m = \varepsilon_m \Delta_{z,m} + \hat{\varepsilon}_m \hat{\Delta}_{z,m} = 0$ for an honest prover. Observe that each party can calculate most of the terms in this sum even without receiving α_m. Specifically, for each MUL gate, define $\phi_m \coloneqq \zeta_m + \alpha_m \lambda_x$. Each party has the information to compute its own share $[\phi_m]$ using information already available: the public ε_m and $\hat{\varepsilon}_m$ values, the known offsets e, and the corrupted shares $[\lambda]$.

Also, recall from the original protocol that the parties never test each $[\zeta_m]$ directly, but rather they only test that the sum $Z = \sum_{m \in M} \zeta_m$ equals zero. We can rewrite the shares of this test in the following way. (We add a subscript m to the wire values to make it unambiguous which gate each wire belongs to.)

$$[Z] = \sum_{m\in M} [\zeta_m] = \sum_{m\in M} [\phi_m] - \sum_{m\in M} \alpha_m[\lambda_{m,x}].$$

The corrupted parties can compute their shares for the left sum. However, the sharing of the remaining term, which we will name

$$\beta = \sum_{m\in M} \alpha_m\lambda_{m,x} \tag{4.3}$$

is problematic because it seems to require each α_m to be known in the clear to calculate the sum. This is where the original protocol $\Pi_{\text{TurboIKOS}}$ revealed all α_m, so that each party could compute their share of β as $\sum_{m\in M} \alpha_m[\lambda_{m,x}]$.

We proceed to show a different way that the prover can commit to and provide the shares for $[\beta]$, which we also describe pictorially in Fig. 4·3. Let $[x]_i$ denote the ith share of x. The crucial observation is that all shares $[\alpha_m]$ can be revealed without a loss in privacy; our only objective here is a performance improvement to avoid sending these shares, even though we could safely do so. Furthermore, observe that:

$$\begin{aligned}\beta &= \sum_{m=1}^{M} \alpha_m\lambda_{m,x} = \sum_{m=1}^{M}\left(\sum_{i=1}^{N}[\alpha_m]_i\right)\left(\sum_{j=1}^{N}[\lambda_{m,x}]_j\right) = \sum_{m=1}^{M}\sum_{i=1}^{N}\sum_{j=1}^{N}[\alpha_m]_i[\lambda_{m,x}]_j \\ &= \sum_{j=1}^{N}\sum_{i=1}^{N}\beta_{i,j}, \text{ where } \beta_{i,j} = \sum_{m=1}^{M}[\alpha_m]_i[\lambda_{m,x}]_j \end{aligned} \tag{4.4}$$

We will take advantage of the MPC-in-the-head structure of our proof to create a sharing where each party essentially holds a "column" of these values: that is, party j's share of β is $\sum_{i=1}^{N}\beta_{i,j}$. In a normal MPC protocol, each pair of parties i and j could collaborate to compute $\beta_{i,j}$. In MPC-in-the-head this is mostly unnecessary, because $\mathcal{V}$ has corrupted $N-1$ of the parties, it has $N-1$ of these $[\alpha_m]_i$ shares and therefore can compute $\beta_{i,j}$ for all corrupted parties $i, j \in T$. It is missing only $\beta_{i^*,j}$, where i^* is the remaining, uncorrupted party.

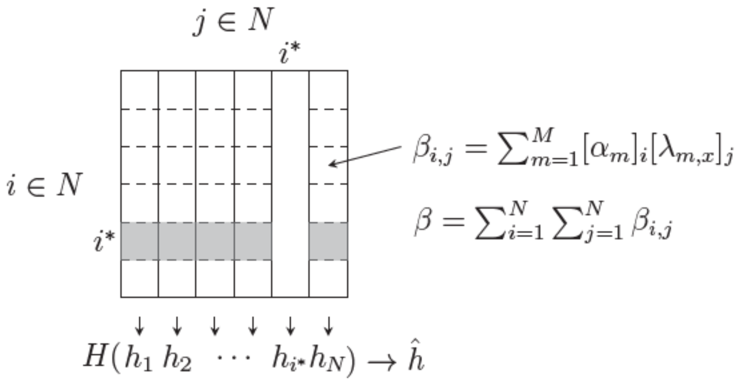
$j \in N$
i^*
$i \in N$
i^*
$\beta_{i,j} = \sum_{m=1}^{M} [\alpha_m]_i [\lambda_{m,x}]_j$
$\beta = \sum_{i=1}^{N} \sum_{j=1}^{N} \beta_{i,j}$
$H(h_1\, h_2\, \cdots\, h_{i^*} h_N) \rightarrow \hat{h}$

Figure 4·3: *Creating a more efficient sharing of β (Eq. 4.3). Each cell holds a single $\beta_{i,j}$ value (Eq. 4.4). A single commitment $\hat{h}$ binds the entire matrix. To open $N-1$ of the columns, $\mathcal{P}$ gives $\mathcal{V}$ the field elements in the shaded region along with h_{i^*}.*

In Fig. 4·3, the parties' shares of β are the sum of the elements in each column. $\mathcal{V}$ has $N-1$ elements (the unshaded regions in each column) out of the N elements needed to compute the full share (column sum), but needs to be sent the remaining (shaded) element to sum the column and compute the share.

For soundness, we require that all $\beta_{i,j}$ be committed to in advance. To do this, we concatenate all elements in each column and commit to them; call these commitments $h_1, \ldots, h_N$. Each commitment h_j must be randomized since the field elements might be small enough not to provide the required min-entropy on their own, but this can be done "for free" by using party j's seed to generate this random value. We then commit to the concatenation of those commitments and give the result $\hat{h}$ to $\mathcal{V}$ in round 3, before the corrupted parties are chosen.

To check the commitments, $\mathcal{V}$ checks two properties: First, the commitment to the $\beta_{i,j}$ values must be consistent with the party views, and second, the shares of Z must sum to 0. To show the first property, $\mathcal{V}$ first recreates all $(N-1)^2$ elements it can from the parties it corrupted. It then receives $(N-1)$ missing elements (the shaded region in Fig. 4·3) from $\mathcal{P}$, which lets it compute the shares $[\beta]_j$ for the parties it corrupted and also the commitment h_j to the concatenation of the elements in each column. $\mathcal{V}$ does not get the missing share $[\beta]_{i^*}$; instead, it is given h_{i^*}, the missing commitment. Using all these, it can check $\hat{h}$ to ensure the corrupted party views are consistent with the commitment to the $\beta_{i,j}$.

Interactive phase, continued from Fig. 4·1:

3. $(\mathcal{P} \to \mathcal{V})$ $\mathcal{P}$ commits to all $\beta_{i,j}$ by sending $\hat{h}$ and commits to all $[Z]$ shares. These variables are computed as follows.
   - For $i \in N$, for $j \in N$, $\beta_{i,j} = \sum_{m=1}^{M} [\alpha_m]_i [\lambda_{m,x}]_j$.
   - For every party $j \in N$, assign $[Z]_j = \sum_{m \in M} \big(\varepsilon_m e_{m,z} - \varepsilon_m e_{m,x} e_{m,y} + \hat{\varepsilon}_m \hat{e}_z + \varepsilon_m e_y [\lambda_{m,x}] + \varepsilon_m e_{m,x} [\lambda_{m,y}] - \varepsilon_m [\lambda_{m,z}] - \hat{\varepsilon}_m [\hat{\lambda}_{m,z}]\big) - \sum_{i \in N} \beta_{i,j}$.
4. $(\mathcal{V} \to \mathcal{P})$ As before, $\mathcal{V}$ samples a set $T = N \setminus \{i^*\}$ of $N-1$ parties to corrupt.
5. $(\mathcal{P} \to \mathcal{V})$ $\mathcal{P}$ opens commitments to $\mathcal{V}$ in the following way:
   - As before, reveal $\log(N)$ seeds from preprocessing step 1 that suffice for $\mathcal{V}$ to recompute σ_p for all corrupted parties $p \in T$, but not the remaining seed σ_{i^*}.
   - Reveal $\beta_{i^*,j}$ for all $j \neq i^*$ and reveal h_{i^*}.

Verification. $\mathcal{V}$ accepts only if all of the following are true for all executions:

- The output values (v_w for $w \in O$) provided by $\mathcal{P}$ correspond to logical true.
- The commitments in rounds 1 and 3 are consistent with the opened keys σ_p for corrupted parties, with the $\beta_{i,j}$ opened, and with the shares of $[Z]$ and $[\lambda_w]$ for output wires for *all* parties. $\mathcal{V}$ computes these using the party seeds, the remaining $\beta_{i^*,j}$ values it is given, and the known Z and output values.

Figure 4·4: *Ending of the improved zero knowledge protocol $\tilde{\Pi}_{TurboIKOS}$ between prover $\mathcal{P}$ and honest verifier $\mathcal{V}$, given a relation represented as an arithmetic circuit with ADD and MUL gates over a field $\mathbb{F}$. See Figure 4·1 for the beginning of this protocol. Compared to Fig. 4·2, step 3 is new, and this change affects the opening and checking of commitments in step 5 and verification; the remainder of the protocol is unchanged from before. If multiple zero tests are desired, then steps 2, 3, and 5 are repeated with independent ε_m and $\hat{\varepsilon}_m$, Z, and $\beta_{i,j}$ values.*

Second, as in the other version of this protocol, the shares to Z are also committed to, but where the β component of Z is shared using this method. $\mathcal{V}$ checks that $Z = 0$ by recomputing the $N-1$ shares it corrupted, and then computes the last share by subtracting those shares from 0. The recomputed shares are checked against the commitment to the $[Z]$ shares from round 3.

This portion of the proof sends $(N-1)$ field elements (the shaded elements) and one commitment (h_{i^*}) per repetition; the additional commitments to the shares of Z

and the $\hat{h}$ need only use one commitment for the entire proof, across all repetitions. It bears repeating that this method was able to check the consistency of all multiplication gates without revealing an additional α_m value per MUL gate. This allows us to add additional fresh zero-tests independently of the number of MUL gates as well. If there are U of these tests, our communication per repetition becomes about $(2M+UN)$ field elements, which can outperform Katz et al. [188] when the number of parties N is small. The description of this version of the protocol is given in Fig. 4·4.

4.5 Security analysis

In this section we prove the honest-verifier zero-knowledge and knowledge soundness properties of the two protocols constructed in §4.4. For each property, we first analyze the base protocol $\Pi_{\text{TurboIKOS}}$, and then we describe how the analysis changes for the improved protocol $\tilde{\Pi}_{\text{TurboIKOS}}$.

Theorem 4.5.0.1. *When instantiated with a pseudorandom function PRF and a computationally hiding commitment scheme Com, both the base protocol $\Pi_{TurboIKOS}$ and improved protocol $\tilde{\Pi}_{TurboIKOS}$ run with N parties and R repetitions is honest-verifier computational zero knowledge with distinguishing bound at most $R\cdot(Adv_{\mathsf{Com}}+Adv_{\mathsf{PRF}})$.*

Proof. We focus here on the privacy argument for the base protocol $\Pi_{\text{TurboIKOS}}$. The privacy of the improved protocol $\tilde{\Pi}_{\text{TurboIKOS}}$ then follows immediately from the fact that it provides strictly less information to the verifier than the base protocol $\Pi_{\text{TurboIKOS}}$ does.

Additionally, we prove the statement for a single repetition of the base protocol $\Pi_{\text{TurboIKOS}}$, from which the theorem follows by a union bound over the independent repetitions.

Let I and M' represent the set of input wires and MUL-gate-output wires respectively. Consider a simulator that follows the following steps during the interactive protocol.

1. The simulator samples all verifier challenges uniformly at random: all $\varepsilon_m, \hat{\varepsilon}_m$ in round 2, and a party $i^* \in N$ in round 4 to be the "uncorrupted party" whose key will not be revealed to $\mathcal{V}$.

2. Choose keys σ_p for all parties $p \in N$ uniformly at random, honestly following step 1 of preprocessing.

3. For all corrupted parties, derive all shares $[\lambda_w]$ for all wires w from the keys honestly, as in step 3 of the preprocessing.

4. For the output wires $w \in O$, choose the e_w values uniformly at random, and set party i^*'s share of λ_w such that $v_w = e_w - \lambda_w$ represents logical true.

5. Now, work backward through the circuit in reverse topological order.

   (a) For ADD gates, choose a random setting of the e_x and e_y values on the input wires to the ADD gate, conditioned on meeting the linear constraints induced by all ADD gates at this layer.

   (b) For a MUL gate with input wires x and y and output wire z, the values of e_x, e_y, and the corrupted shares of $\hat{\lambda}_y$, $\hat{\lambda}_z$, and $\hat{e}_z$ must all be simulated. Note that e_x, e_y, or both may already be set by an existing constraint (e.g. if the wire was reused in a later layer or used in multiple gates).

      i. Generate $\hat{\lambda}_y$ and initialize the $\hat{\lambda}_z$ shares honestly for the corrupted parties from the party keys (leaving it unspecified for party i^*).

      ii. Generate $\hat{e}_z$ uniformly (and also e_x or e_y, if they are unspecified).

      iii. Let Δ_z be the difference between the value on the output wire z and the product of the value on the input wires xy. That is, $\Delta_z = v_z - v_x v_y = e_z - e_x e_y - \lambda_z + e_x \lambda_y + e_y \lambda_x - \lambda_x \lambda_y$. Let $\hat{\Delta}_z$ be the difference between $(\hat{e}_z - \hat{\lambda}_z)$ and $\lambda_x \hat{\lambda}_y$; that is, $\hat{\Delta}_z = \hat{e}_z - \hat{\lambda}_z - \lambda_x \hat{\lambda}_y$. For an honest prover, $\Delta_z = \hat{\Delta}_z = 0$, but the simulator is not honest so these values are likely to be non-zero. Using the foreknowledge of ε_m and $\hat{\varepsilon}_m$, alter party i^*'s share of $\hat{\lambda}_z$ so that $\zeta = \varepsilon_m \Delta_z + \hat{\varepsilon}_m \hat{\Delta}_z$ equals 0 (or alter λ_z if $\hat{\Delta}_z = 0$ but $\Delta_z \neq 0$).

6. In round 1, commit honestly to all parties' keys σ_p and all parties' shares $[\lambda_z]$ for all output wires $z \in O$. Also, send the offsets e_z for all $z \in I \cup M'$.

7. In round 3, commit to all shares $[\alpha_m]$ and $[Z]$, where $[Z]$ is computed correctly from the already-manipulated ζ shares from above.

8. In round 5, honestly open the key commitments for all parties except i^*. Also, honestly reveal party i^*'s shares $[Z]$, $[\alpha_m]$, and $[\lambda_w]$ for output wires.

It is straightforward to confirm that the simulated proof passes all verification checks. It remains to show that the simulated proof is computationally indistinguishable from a real one. As a stepping stone, we consider a hybrid proof H that is constructed like the real one, except using an ideal commitment scheme Com (where it is impossible to recover an un-opened key) and a truly random function in place of PRF. The distinguishing probability between the real game and H is at most $\mathrm{Adv}_{\mathsf{Com}} + \mathrm{Adv}_{\mathsf{PRF}}$.

In the hybrid world, we claim that all of the information provided by $\mathcal{P}$ to $\mathcal{V}$ throughout the proof is meaningless. The commitment to σ_{i^*} is now useless, values like the output wires or Z are publicly known to $\mathcal{V}$ beforehand, and the remaining information (e_w for all wires $w \in W$, $\hat{e}_z$ and α_m values for all MUL gates, and the shares $[Z]$) contains masks that hide the real values from $\mathcal{V}$.

- On each wire w, the revealed $e_w = v_w + \lambda_w$ does not reveal anything about the value on the wire v_w because it is masked by party i^*'s share of λ_w.
- For each MUL gate $m \in M$, information in α_m is masked by i^*'s share of $\hat{\lambda}_y$.
- For each MUL gate, $\hat{e}_z$ hides info about $\lambda_x \hat{\lambda}_y$ by masking it with party i^*'s share of $\hat{\lambda}_z$.
- Party i^*'s share of Z reveals no information because it can be computed as $-\sum_{p \in T}[Z]$ (leveraging the fact that $Z = 0$ is public knowledge), and the corrupted parties' shares of Z are only a function of their own data.

All of these masks are truly random in the hybrid world. Observe that e_w, $\hat{e}_z$ and α_m all have the uniform distribution in the simulated world as well. Therefore, the distance between the hybrid and simulated games is 0, which completes the proof for the base protocol $\Pi_{\text{TurboIKOS}}$. □

Next, we examine the soundness of the base protocol $\Pi_{\text{TurboIKOS}}$. We focus on its security for the non-interactive version of the MPC-in-the-head construction using the Fiat-Shamir transform using a random oracle H with 2κ bits of output, so that finding a collision has 2^κ cost. (The interactive version of the protocol has even better soundness because the prover cannot rewind the verifier.)

Theorem 4.5.0.2. *Consider the non-interactive version of the base protocol $\Pi_{\mathit{TurboIKOS}}$ over a large field $|\mathbb{F}| = 2^{\kappa}$ and instantiated with a random oracle H with 2κ output length. Then, Protocol $\Pi_{\mathit{TurboIKOS}}$ with $R = \frac{\kappa}{\log_2(N)} + 1$ repetitions provides knowledge soundness with error at most $1/2^{\kappa}$.*

Proof. We focus here on proving the traditional soundness property. The stronger knowledge soundness claim immediately follows by applying the analysis from Katz et al. [188, §3.1] in order to build an extractor that recovers a witness by observing the inputs to the random oracle-based commitment scheme on a single execution. The reduction is tight because the extractor never needs to rewind.

To prove soundness, consider a malicious prover $\mathcal{P}^*$ that is attempting to prove a false statement. Since an honest execution of the circuit would return logical false, the prover must deviate from the protocol on each repetition. There are effectively four different places where the malicious prover $\mathcal{P}^*$ can deviate from the protocol in order to gain an advantage:

1. In round 1 of the protocol, $\mathcal{P}^*$ can change the offsets for one or more MUL gates, so that the Δ_z or $\hat{\Delta}_z$ values on these gate(s) are non-zero. (We presume it is simple for the prover to determine a sufficient set of gates to tamper in order to cause the circuit to return logical true.) $\mathcal{P}^*$ will learn in round 2 whether $\mathcal{V}$ catches this deviation or whether $\mathcal{P}^*$ has successfully evaded detection. By Lemma 4.4.3.2, the prover is successful with probability $1/|\mathbb{F}|$.

2. In round 3, $\mathcal{P}^*$ can change one party's $[\alpha_m]$ and $[Z]$ shares so that the verifier reconstructs a Z value of 0. The success of this attack is revealed in round 4, and $\mathcal{P}^*$ evades detection with probability $1/N$.

3. In rounds 1 or 3, $\mathcal{P}^*$ can attempt to break the binding property of the commitment scheme and open it later to different values.

4. In round 1, $\mathcal{P}^*$ can change one party's share of the final output so the result becomes logical true. $\mathcal{P}^*$ will learn in round 4 whether $\mathcal{V}$ catches this deviation or whether $\mathcal{P}^*$ has successfully evaded detection (this occurs with probability $1/N$).

(Observe that the pseudorandom function has no impact on soundness. It only exists to 'compress' each party's share of each wire label $[\lambda_w]$, and any tampering of these wire labels is equivalent to tampering the corresponding offset e_w.)

The key observation in this proof is that item 1 is strictly better for $\mathcal{P}^*$ than item 3 and that item 2 is strictly better than item 4. The first part of this claim follows from the observation that items 1 and 3 both require the prover to deviate in round 1 and the first has better probability of success since the commitment scheme has soundness κ. The second part of this claim is true because the attacks in items 4 and 2 both give the same probability of success and are both revealed in round 4, yet the alteration of $[\alpha_m]$ and $[Z]$ occurs later. Delaying the start of the attack is strictly better for $\mathcal{P}^*$ because it can wait to see if the attack on wire offsets in round 1 was successful, and only attempt this attack if necessary.

Concretely, we use the same proof technique as several recent analyses of non-interactive zero knowledge proofs that apply the Fiat-Shamir transform to protocols with more than three rounds [23, 41, 185]. We consider all *attacker strategies* (r_1, r_2) in which the prover $\mathcal{P}^*$ changes wire offsets in round 1 until r_1 repetitions happen to have $Z = 0$ anyway, and then $\mathcal{P}^*$ attacks the remaining $r_2 = R - r_1$ repetitions in round 3 by altering α_m and Z. In general, the cost of any multi-round attack strategy is given by $C = 1/p_1 + 1/p_2$, where p_1 and p_2 denote the probability that the first and second parts of the attack succeed, respectively. To achieve κ soundness, we must choose a sufficiently large number of repetitions R so that any attacker strategy $(r_1, R - r_1)$ has a total cost $C \geq 2^\kappa$.

In this case, one attacker strategy dominates the rest: $r_1 = 1$ and $r_2 = R - 1$. In more detail, the malicious prover $\mathcal{P}^*$ changes the wire offsets for all repetitions in round 1, and rewinds until finding an input that evades detection from the verifier's round 2 challenge on $r_1 = 1$ instance. Because each instance evades detection only with probability $1/|\mathbb{F}| = 1/2^\kappa$, the malicious prover $\mathcal{P}^*$ can only expect to evade detection on $r_1 = 1$ instance in time less than 2^κ. Even succeeding on 2 instances is exceedingly unlikely in the adversary's runtime, and the $\mathcal{P}^*$ gains no benefit by foregoing an attack on round 1 altogether. Thereafter, $\mathcal{P}^*$ must complete the attack on the remaining $r_2 = R - 1$ repetitions by altering $[\alpha_m]$ and $[Z]$. This change is undetected by the verifier with probability $1/N$ independently for each repetition. Hence, the overall probability of success for the second repetition is $(1/N)^{r_2}$, and thus its cost exceeds 2^κ when $R = \frac{\kappa}{\log_2(N)} + 1$. □

Theorem 4.5.0.3. *Consider the improved protocol $\tilde{\Pi}_{TurboIKOS}$ that is instantiated with a random oracle H with 2κ output length and executed over the field $\mathbb{F}$ with N parties, R repetitions, and U tests per repetition. Then, the protocol satisfies* *knowledge soundness with security parameter κ if:*

$$\min_{0\leq r_1\leq R}\left\{\left[\sum_{i=r_1}^{R}\binom{R}{i}\cdot\left[\frac{1}{|\mathbb{F}|}\right]^{Ui}\cdot\left[1-\frac{1}{|\mathbb{F}|}\right]^{U(R-i)}\right]^{-1}+N^{(R-r_1)}\right\}>2^{\kappa}. \tag{4.5}$$

Proof. Once again, we focus on proving traditional soundness, after which we can build an extractor for knowledge soundness using the same technique as before. Additionally, the analysis from Theorem 4.5.0.2 about the options for a malicious prover $\mathcal{P}^*$ to deviate from the protocol holds here too. The prover's dominant strategy remains to change the offsets for one or more MUL gates in a way that causes the remainder of the circuit (computed honestly) to return logical true; note that this will also cause the corresponding Δ_z or $\hat{\Delta_z}$ values to be nonzero. For each repetition independently, there are two ways for the malicious prover $\mathcal{P}^*$ to evade detection by the verifier:

1. Based on the verifier's U independent random choices of ε_m and $\hat{\varepsilon}_m$ in round 2, there is a $(1/|\mathbb{F}|)^U$ probability that all of the Z tests happen to equal 0 (by Lemma 4.4.3.2).

2. If even a single Z value is non-zero, then $\mathcal{P}^*$ can commit to erroneous $\beta_{i,j}$ values to 'fix' this error. Note that $\mathcal{P}^*$ can only inject erroneous data in one row of the table in Fig. 4·3 because the verifier can check the remaining $N-1$ rows directly. This attack evades detection only if the verifier chooses in round 4 to leave this row as the uncorrupted party, which happens with probability $1/N$.

As before, we can analyze the malicious prover $\mathcal{P}^*$'s probability of success by analyzing all attacker strategies (r_1, r_2) that operate as follows. First, the prover $\mathcal{P}^*$ rewinds round 1 until at least r_1 repetitions have the property that the verifier's choice of ε_m and $\hat{\varepsilon}_m$ are such that *all* of the Z tests within these repetitions equal 0. Second, $\mathcal{P}^*$ rewinds round 3 until the remaining $r_2 = R - r_1$ have been tampered in the locations chosen by the verifier in round 4.

To achieve κ soundness, we must select enough repetitions R so that any attacker strategy $(r_1, R-r_1)$ has a total cost $C \geq 2^{\kappa}$. Here, the *cost* of this multi-round attack strategy is given by $C = 1/p_1 + 1/p_2$, and p_1 and p_2 denote the probability that the first and second parts of the attack succeed, respectively.

Because we consider an arbitrary field size in this theorem, the cost analysis here is more complicated than in Theorem 4.5.0.2. Our argument is very similar to that of Banquet [23]. The first probability boils down to the chance that the attacker has r_1 successes out of R trials, where each trial succeeds with probability $(1/|\mathbb{F}|)^U$. This is the right tail of a binomial distribution:

$$p_1 = \sum_{i=r_1}^{R} \binom{R}{i} \cdot (1/|\mathbb{F}|)^{Ui} \cdot (1 - 1/|\mathbb{F}|)^{U(R-i)}. \tag{4.6}$$

The second probability is simply $p_2 = (1/N)^{r_2}$ because the malicious prover must succeed on all remaining repetitions of the protocol. Combining the costs of both parts of the attack results in Eq. (4.5), completing the proof of the theorem. □

κ	N	U				
		1	2	4	6	9
128	8	66	53	47	45	44
	16	54	41	36	34	33
	31	47	35	30	28	27
192	8	99	79	70	68	66
	16	81	62	54	52	50
	31	71	53	45	43	41
256	8	132	106	95	91	89
	16	108	83	73	69	67
	31	95	71	61	57	55

Table 4.2: *Valid parameter settings for $\mathbb{F}_{2^8}$. The body of the table shows the number of repetitions R based on the soundness parameter κ, number of parties N, and number of tests per repetition U.*

The goal here is to find the "minimum" choices of N, R, and U that yield a desired soundness parameter κ for a circuit with a given field size $|\mathbb{F}|$. While it is challenging to write a closed-form version of Eq. 4.5 that connects the five parameters, it is easy to find satisfying tuples empirically. In Table 4.2, we show several such choices for the field $\mathbb{F}_{2^8}$. We include a computer program that calculates valid parameter settings within our open source repository [156].

4.6 Performance and prototype implementation

In this section, we compare our $\tilde{\Pi}_{\text{TurboIKOS}}$ system's performance against other MPC-in-the-head based systems, and we describe our prototype Python implementation of $\Pi_{\text{TurboIKOS}}$.

4.6.1 Performance analysis

To evaluate our performance, we measure our signature size when computing a variant of the Picnic signature scheme [224]. Picnic uses MPC-in-the-head (specifically, a variant of the Katz et al. protocol [188]) and LowMC [6], a block cipher with few multiplications that is designed to be efficient in secure computation. Picnic is currently an "alternate candidate" in round 3 of the NIST post-quantum crypto competition [5]. BBQ [104] introduced the idea of using AES in Picnic-like signatures and showed that the signature sizes could be competitive with those using LowMC. To achieve this, rather than evaluating the binary circuit for AES, they used an arithmetic MPC-in-the-head system over $\mathbb{F}_{2^8}$; this facilitates proving constraints about inverses in $\mathbb{F}_{2^8}$, the non-linear component of the AES S-box. They also show how each field inversion can be reduced to a single multiplication gate (without testing for the case in which the input and output are both equal to 0) with very small reduction in the soundness of the resulting system (less than 3 bits of security). We follow their approach in this section.

Table 4.3 shows the proof sizes of $\tilde{\Pi}_{\text{TurboIKOS}}$ when computing signatures using AES at different security levels. We compare against the following systems:

- BBQ [104]: The figures are taken directly from their paper, except the number of parties is not listed. We make an educated guess that they use approximately $N = 64$ parties; for lower N their proof size will be larger.
- Katz et al. [188]: We calculate the proof size using the formula in [188], assuming

that a field inversion constraint can be verified with two field elements per MUL gate like in this work. Also, our calculations use 32 parties rather than 31; this result should be a conservative smaller estimate of the actual proof size when $N = 31$.

- Baum-Nof [25]: We calculate a conservative underestimate of the proof size using the formula in [25], assuming that only a single zero-check is needed per repetition. However, this is unlikely to be the case in $\mathbb{F}_{2^8}$ for the same reason as in our work.

- Banquet [23]: The figures are taken directly from their paper.

Scheme	**N**	**Protocol** *(all sizes in KB)*				
		BBQ [104]	Katz et al. [188]	Baum-Nof [25]	$\tilde{\Pi}_{\text{TurboIKOS}}$	Banquet [23]
AES-128 (L1)	8	|	26.7	37.3	23.8	–
	16	31.6	22.4	29.4	20.6	19.8
	31	|	19.8	24.8	19.8	17.5
AES-192x2 (L3)	8	|	76.3	110.8	66.7	–
	16	86.9	63.1	87.4	56.3	51.2
	31	|	54.9	73.4	52.2	45.1
AES-256x2 (L5)	8	|	122.6	179.9	109.3	–
	16	133.7	101.9	140.7	90.4	83.5
	31	|	89.1	118.0	82.8	73.9

Table 4.3: *Signature size comparison for the Picnic signature scheme at different security levels for different systems. All signature sizes are shown in kilobytes (KB). AES-128 has* $(M, I) = (200, 128)$*, AES-192x2 has* $(M, I) = (416, 192)$*, and AES-256x2 has* $(M, I) = (500, 256)$*.*

Banquet [23] is independent recent work that reduce the size of AES-based Picnic signatures using very different techniques based on polynomial interpolation in extension fields of $\mathbb{F}_{2^8}$ [54, 93] that are not directly applicable to general fields. They achieve a smaller proof size than all competitors, including us. However, this advantage comes with two downsides relative to our scheme and prior ones. First, performing polynomial arithmetic in extension fields would be costly on embedded devices whose CPU architectures typically have a small word size. Second, the polynomials are proportional to the entire circuit size, making their system more memory-costly than traditional MPC-in-the-head based methods like ours where the memory is only proportional to the circuit width (i.e., the memory required to compute the circuit).

4.6.2 Prototype implementation

We created a prototype implementation for $\Pi_{\text{TurboIKOS}}$ in Python. We did not optimize the runtime or memory usage of our code, thus, we will likely not win on prover runtime with other MPC-in-the-head approaches written in C or C++, e.g. [25, 188]. That having been said, our protocol is amenable to all of the optimizations made by the recent Reverie software [288] implementing the protocol of Katz et al. Our code currently achieves runtimes about twice as long as Reverie for the same size circuit. In this section we briefly describe our implementation.

Our Python implementation [156] supports both Bristol fashion circuits[1] and Prover Worksheet (PWS) format[2] as input. The circuit is parsed into a list of Gate objects, found in gate.py, and initializes a Wire data structure, found in wire.py, that takes in a list of dictionaries containing the values on each wire.

Dictionaries are used extensively to manage information. Circuit information such as the number of various types of gates are stored in a dictionary. Values on each wire such as e and λ are also stored in a dictionary. Each wire has a dictionary of values, resulting in a list of dictionaries with length of the number of wires. This list of dictionaries is later used as the input to the Wire object, found in wire.py, which defines functions to access values on the wire.

On the Prover side, $\mathcal{P}$ calculates all the parties as she parses through the circuit, so $\mathcal{P}$ uses the Wire objects to access a party's value. On the Verifier side, $\mathcal{V}$ picks

[1] https://homes.esat.kuleuven.be/~nsmart/MPC/
[2] https://github.com/hyraxZK/pws

one party to leave unopened and reconstructs the other $N-1$ views from the seeds given by $\mathcal{P}$. We generate the shares λ, $\hat{\lambda}_y$, and $\hat{\lambda}_z$ pseudorandomly using AES as a PRF with the party's seed as the key and the concatenation of the (fixed-length) wire index and type of value as the message.

The prover is required to send a commitment in Round 1 and Round 3. As discussed in §4.3, when committing to values with sufficient min-entropy, we simply use $\mathsf{Com}(m) := H(m)$, thus decommitments are "free" aside from the cost of m itself. This remains computationally hiding as long as m has sufficient min-entropy and H is a random oracle. We use $\mathsf{SHA2}$ from `hashlib` for our instantiation of H. To achieve non-interactivity via the Fiat-Shamir Transform [120], the random messages sent by the Verifier in Round 2 and Round 4 are replaced by a call to H on the info sent by $\mathcal{P}$ in all previous rounds.

4.7 Introduction to BooLigero

In this work, we focus on and improve Ligero [10], a protocol that achieves a balance between proof size and prover runtime.

4.7.1 Related work

As we described in §4.1, zero knowledge proofs are evaluated for performance on three metrics: proof/argument size, prover runtime, and verifier runtime. There is a spectrum of zero-knowledge proof/argument systems.

On one extreme of the spectrum, large, fast proofs construct ZK proofs from various flavors of MPC: the garbled-circuit based approach of ZKGC [182] (with improvements from [201]) or approaches that use the GMW [142] paradigm (e.g. [177], improved in [138] and [76]). All of these are fairly quick to compute, but they incur a linear proof size (except the very recent work of [316], which cannot be made non-interactive, and is therefore not usable in most blockchain scenarios).

On the other extreme, we have "succinct" sublinear-size arguments. The smallest arguments are constant size, but generally suffer from two problems – assumptions and trusted setup. Many of these arguments use unfalsifiable assumptions (e.g., [34,37,94,136,153,214,243]) and this is inherent at a certain level [137]. Others require a trusted setup step performed by a central authority or a trusted committee operating a costly multiparty computation (e.g. [33, 34, 44, 80, 131, 136, 153, 154, 218, 243, 324]), both being undesirable or even unacceptable in many financial use cases.

In the middle, there exist transparent protocols that achieve sublinear (but not constant) size without the need for trusted setup. A number of these protocols use assumptions that render them vulnerable to quantum attacks (e.g. [55, 169, 267, 312]). There are three different approaches to sublinear transparent protocols without trusted setup that are plausibly post-quantum secure: Ligero [10], Stark [32], and Aurora [35].

Compared to Ligero and BooLigero, Stark's proof size is asymptotically smaller ($O(\log^2 s)$ instead of $O(\sqrt{s})$ for circuit size s), but concretely larger for circuits smaller than approximately 10^6 gates, as shown in [312]. Its prover runtime is more expensive than Ligero's both asymptotically by a $\log s$ factor, and is also concretely longer. For circuits with repeated sub-circuits, Stark has significantly improved verifier runtime, but there is no asymptotic difference for circuits without this property.

Aurora [35] also has a significantly smaller proof size than Ligero and BooLigero ($O(\log^2 s)$ instead of $O(\sqrt{s})$) and the same asymptotic prover and verifier runtime. However, its interactive version has a $O(\log s)$ round complexity compared to Ligero and BooLigero's $O(1)$, and its prover runtime is concretely higher than Ligero's. Moreover, without a certain unproven conjecture involving Reed-Solomon codes, it becomes much less efficient (see discussion in [267]).

Finally, the lattice-based ZK argument of Baum et al. [21] has proof size

$O(\sqrt{s\lambda \log^3 s})$ asymptotically, but its concrete performance remains unknown (see discussion on p. 6 of Ben-Sasson et al. [35]).

4.8 Preliminaries

Notation. We use $\mathbb{F}$ to refer to a finite field, and $\mathrm{GF}(2^w)$ to refer to a finite field with order 2^w. We also often use w to refer to the "word size" and refer to elements of $\mathrm{GF}(2^w)$ as "w-words" when we use their w-bit representations.

For operations, we use $\oplus$ for bitwise XOR and & for bitwise AND, over bits or w-words depending on context. We use $*$ to denote Galois field multiplication, and $\cdot$ for element-wise multiplication of vectors.

Bit indexing, denoted with square brackets, always begins at 1. Bitstrings are always shown in big endian. Thus, if $x = 0001$, then $x[1] = 1$ is the least significant bit of x.

Zero knowledge IOPs. A ZKIOP is an interactive oracle proof (IOP) [36] that is additionally zero-knowledge. Let $\mathcal{P}$ and $\mathcal{V}$ be probabilistic polynomial-time interactive Turing machines. An interactive oracle protocol between $\mathcal{P}$ and $\mathcal{V}$ occurs over several rounds. $\mathcal{P}$ reads messages sent by $\mathcal{V}$ fully, but $\mathcal{V}$ queries random parts of $\mathcal{P}$'s message rather than reading them entirely. At the end, $\mathcal{V}$ either accepts or rejects. Let $\langle \mathcal{P}(x, w), \mathcal{V}(x) \rangle$ refer to the output of $\mathcal{V}(x)$ when executing an interactive oracle protocol with $\mathcal{P}(x, w)$. Let R be a relation for language L so that $(x, w) \in R$ if w is a witness for x's membership in L.

Definition 4.8.0.1 (Zero knowledge interactive oracle proof)**.** $\langle \mathcal{P}, \mathcal{V} \rangle$ is a *zero knowledge interactive oracle proof system* for R with soundness error δ if:

- *Completeness*: For any $(x, w) \in R$, $\langle \mathcal{P}(x, w), \mathcal{V}(x) \rangle = 1$.
- *Soundness*: If $x \notin L$, then for all $\mathcal{P}^*$, $\Pr[\langle \mathcal{P}^*, \mathcal{V}(x) \rangle = 1] \leq \delta$

- *Perfect honest-verifier zero knowledge*: Let $\mathsf{View}_{\mathcal{V}}(\mathcal{P}, \mathcal{V}, x, w)$ be the view of $\mathcal{V}$ upon completion of $\langle \mathcal{P}(x, w), \mathcal{V}(x) \rangle$. The protocol is perfect honest-verifier zero knowledge if there exists a probabilistic poly time simulator $\mathcal{S}$ such that for all (x, w), the distribution of $\mathcal{S}(x)$ equals the distribution of $\mathsf{View}_{\mathcal{V}}(\mathcal{P}, \mathcal{V}, x, w)$.

The IOPs we deal with in this paper are also *public-coin*, meaning that $\mathcal{V}$'s messages to $\mathcal{P}$ are always chosen randomly from a known distribution, and $\mathcal{V}$'s queries to $\mathcal{P}$ depend only on messages that have already occurred and that $\mathcal{P}$ has seen. Zero knowledge IOPs can be converted to zero knowledge arguments in a standard way using the Fiat-Shamir transform [36].

4.9 Ligero background

In this section we provide relevant background from [10]. We start with some background on Interleaved Reed-Solomon codes.

Definition 4.9.0.1 (Reed-Solomon Code, [10] Defn. 4.1)**.** For positive integers n and k, finite field $\mathbb{F}$, and a vector $\eta = (\eta_1, \ldots, \eta_n) \in \mathbb{F}^n$ of distinct field elements, the code $L = \mathsf{RS}_{\mathbb{F},n,k,\eta}$ is the $[n, k, n-k+1]$ linear code over $\mathbb{F}$ that consists of all n-tuples $(p(\eta_1), \ldots, p(\eta_n))$ where p is a polynomial of degree $< k$ over $\mathbb{F}$.

Definition 4.9.0.2 (Interleaved Code, [10] Defn. 4.2)**.** Let $L \subset \mathbb{F}^n$ be a $[n, k, d]$ linear code over $\mathbb{F}$. We let L^m denote the $[n, mk, d]$ (interleaved) code over $\mathbb{F}^m$ whose codewords are all $m \times n$ matrices U such that every row U_i of U satisfies $U_i \in L$.

Definition 4.9.0.3 (Encoded message, [10] Defn. 4.5)**.** Let $L = \mathsf{RS}_{\mathbb{F},n,k\eta}$ be an RS code and $\zeta = (\zeta_1, \ldots, \zeta_\ell)$ be a sequence of distinct elements of $\mathbb{F}$ for $\ell \leq k$. For $u \in L$ we define the message $\mathsf{Dec}_\zeta(u)$ to be $(p_u(\zeta_1), \ldots, p_u(\zeta_\ell))$ where p_u is the polynomial (of degree $< k$) corresponding to u. For $U \in L^m$ with rows $u^1, \ldots, u^m \in L$, we let $\mathsf{Dec}_\zeta(U)$ be the length-$m\ell$ vector $x = (x_{11}, \ldots, x_{1\ell}, \ldots, x_{m1}, \ldots, x_{m\ell})$ such that $(x_{i1}, \ldots, x_{i\ell}) = \mathsf{Dec}_\zeta(u^i)$ for $i \in [m]$. Finally when ζ is clear from context, we say the U encodes x if $\mathsf{Dec}_\zeta(U)$.

4.9.1 Proof size of Ligero.

Ligero [10] is a zero-knowledge argument that achieves $O(\sqrt{s})$ proof size, where s is the size of the verification circuit.

Ligero encodes the witness using an Interleaved Reed-Solomon code, which can be considered an m-vector of Reed-Solomon (RS) codewords. Each RS codeword can itself be considered a vector of n elements which encode ℓ unencoded elements, for $n = O(\ell)$. Thus, the overall interleaved Reed-Solomon code can be considered an $m \times n$ matrix encoding $m \times \ell$ variables.

Ligero achieves $O(\sqrt{s})$ proof size by being clever about how the verifier checks constraints on this matrix. Roughly speaking, the communication will consist of some (linear combinations of) rows and some columns of the matrix, with simplified complexity $O(n + m)$. Thus, one can balance m against ℓ and set both to $O(\sqrt{s})$ to achieve a proof size of $O(\sqrt{s})$. Specifically, m is set to $O(\sqrt{s/\kappa})$ and ℓ is set to $O(\sqrt{s\kappa})$, where κ is a security parameter.

We let $L = \mathsf{RS}_{\mathbb{F},n,k,\eta}$ be a Reed-Solomon code with minimal distance. L^m refers to the interleaved code, which has codewords that are simply m codewords of L. L^m is best understood as a matrix where the m rows are L-codewords.

Tests in Ligero. As a zero-knowledge IPCP between the prover $\mathcal{P}$ and verifier $\mathcal{V}$, the prover begins by encoding its witness as a L^m codeword – an $m \times n$ matrix encoding $m \times \ell$ variables in the witness. Ligero creates three tests for constraints over this matrix: **Test-Interleaved** ([10] §4.1), **Test-Linear-Constraints-IRS** ([10] §4.2), and **Test-Quadratic-Constraints-IRS** ([10] §4.3). Each of these tests consists of two phases:

1. **Oracle phase:** $\mathcal{P}$ creates an oracle to the L^m-encoded witness (possibly with some additional info).

2. **Interactive testing phase:** $\mathcal{P}$ and $\mathcal{V}$ interact with each other. $\mathcal{P}$ sends some linear combinations of rows of the matrix. $\mathcal{V}$ makes queries to the oracle to obtain columns of the L^m codeword (without receiving any L codeword "rows" fully). After the interaction, $\mathcal{V}$ checks whether the linear combinations given to it by $\mathcal{P}$ match the columns it queried, and either accepts or rejects.

When used as a zero-knowledge argument (instead of a ZKIPCP), the oracle is replaced with a commitment. Before the interactive testing phase, $\mathcal{P}$ commits to all columns of its encoded witness as the leaves of a Merkle tree that uses a statistically hiding commitment scheme. To make the proof non-interactive, the verifier's messages can be replaced with a random oracle call on the prover's messages up to that point.

Boolean circuits in Ligero. Ligero is presented for arithmetic circuits over a prime field. It is possible to use Ligero for a Boolean circuit as well, but this has two downsides.

The first downside is that one must use an entire field element to represent a single bit. This causes a blowup of $\log |\mathbb{F}|$ in the number of witness elements, which causes a blowup of $O(\sqrt{\log |\mathbb{F}|})$ in the proof size.

How small of a field can we use? There is a minimum requirement that $|\mathbb{F}| \geq \ell + n$ ([10] §5.3), which is required so that there are sufficient evaluation points for L. Furthermore, if the field gets too small, one must repeat the protocol several times in order to achieve the desired soundness. At very small field sizes, the costs of the commitments ($\log s$ times a constant hash output length) also start growing in comparison to the rest of the proof. Concretely, testing out different field sizes for Boolean circuits on the order of 10^6 to 10^9 gates tends to yield optimal field sizes of about 14-20 bits. This suggests that there is approximately a 3.7-4.4x gain to be had by packing the bits efficiently.

The second downside is that this costs additional constraints. First, each extended witness element e must be proven to be 0 or 1 by adding a quadratic constraint that $e^2 - e = 0$. Second, XOR and AND are also both quadratic constraints: the constraint $e_1 + e_2 = a_0 + 2 \cdot a_1$, along with bit constraints on all variables, enforce that a_0 is the XOR of e_1 and e_2, and that a_1 is the AND of e_1 and e_2. Computing only one or the other necessitates the creation of a dummy variable for the other, and enforcing bit constraints on all. Hence, the number of constraints is twice the maximum number of AND and XOR gates combined.

Unlike linear constraints, which can be evaluated using only an encoding of the witness itself, evaluating quadratic constraints like $x * y = z$ requires providing encodings of x, y, and z, separately from (but related to) the encoding of the witness itself. Although the number of quadratic constraints will asymptotically be $O(s)$, this suggests that there may be concrete room for improvement by reducing the number of quadratic constraints.

4.10 BooLigero techniques

We make one minor change and two major changes to Ligero [10], which we described in §4.9.

The minor change is that we use GF(2^w) instead of the prime field GF(p). Ligero's methods work for any finite field, where addition and multiplication now use operations in the new field. Since we are still using Interleaved Reed-Solomon codes, we can directly reuse Ligero's **Test-Interleaved**, **Test-Linear-Constraints-IRS**, and **Test-Quadratic-Constraints-IRS**. The latter two now test bitwise XOR/NOT constraints and GF(2^w) multiplication rather than arithmetic addition and multiplication. We lose the ability to natively check linear arithmetic constraints in mod 2^w, but we gain the ability to cheaply check XORs. We can still check linear arithmetic constraints in power-of-two moduli by building an adder out of the constraint tests we have.

The following two larger changes to Ligero are the focus of our work:

Change 1: Additional constraint tests that reveal variables directly. We add a number of tests for additional constraints. These new tests operate differently than the Ligero tests, and in fact the new tests rely on the Ligero tests in order to check linear and quadratic constraints. In the new tests, the prover modifies and extends the witness with additional variables, some of which are based on a "challenge" sent by the verifier. As part of the proof oracle, the prover sends some (masked) elements of the witness to the verifier directly, and the verifier must check to see whether the revealed elements have a certain property. These tests can be nested inside other tests – e.g., our **Test-And-Constraints** procedure involves invoking **Test-Pattern-Zeros-Constraints**, as described in §4.10.3.

Most of our tests use only linear constraints and cost $O(\kappa)$ (a security parameter) in the proof size, independent of the circuit size and the number of constraints. Our **Test-And-Constraints** involves adding approximately $3\sqrt{w}N$ hidden variables, where N is the number of AND gates. This is still an improvement over the approximately wN added elements that are required to represent wN Boolean wires in plain Ligero for $w \geq 9$. We describe our constraint tests in §4.10.3.

Change 2: Two oracles/rounds of commitment. Unlike original Ligero, many of the tests we add require verifier input in order to choose which constraints we will check – generally, the verifier will pick a random linear combination of the variables to use in constraints. However, for this to be sound, the original variables must already have been available in an oracle (or been committed to). This necessitates splitting the proof oracle in two: one that presents an encoding of the "original" witness, and one that is parameterized by the verifier's random choices and returns an encoding of the added variables. We call the first oracle the "initial oracle" and the second the "response oracle". Thus, whereas Ligero had two phases of procedures – the oracle phase and the interactive testing phase – we have four: initial phase (creation of initial witness to be provided as oracle or commitment), challenge phase (verifier sends random bits as challenge), response phase (creation of witness extension to be provided as a second oracle or commitment), and the interactive testing phase We describe each of these phases in §4.10.1. This process is based on the circuit sampling idea of Baum and Nof [25].

4.10.1 Test procedures

In original Ligero, each test consists of an oracle and an interactive test procedure which will ensure that the oracle is valid. In our protocol, each test consists of two oracles, separated by a verifier challenge, and followed by an interactive test procedure. The second oracle is the response to the challenge. We describe each of our constraint test procedures in four phases:

1. **Initial phase:** $\mathcal{P}$ adds elements to the witness, encodes it, and provides the columns of the encoding as the first proof oracle.

2. **Challenge phase:** $\mathcal{V}$ sends random bits to $\mathcal{P}$, which will be required to generate the second proof oracle.

3. **Response phase:** Based on the bits received in the challenge, $\mathcal{P}$ adds more elements to the witness, and adds additional constraints. $\mathcal{P}$ encodes the extensions to the witness, and provides the encoding (which can be combined with the first oracle's output) as well as the revealed variables.

4. **Interactive testing phase:** $\mathcal{P}$ and $\mathcal{V}$ run an interactive testing protocol. At the end, $\mathcal{V}$ has acceptance criteria for determining whether to accept or

reject the proof. Our tests augment the original Ligero acceptance criteria with additional checks on properties of the revealed variables.

In slightly more detail, the variables in the witness consist of:

- v_0 original variables
- v_1 added hidden variables in the initial phase
- v_2 added hidden variables in the response phase
- v_3 added revealed variables in the response phase

$\mathcal{P}$ and $\mathcal{V}$ first set ℓ, m_1, m_2, and m_3 so that $\ell m_1 \geq v_0 + v_1$, $\ell m_2 \geq v_2$, and $\ell m_3 \geq v_3$. $\mathcal{P}$ creates the initial witness encoding $U^{\mathbf{w}_1} \in L^{m_1}$ from the v_0 original variables and v_1 added hidden variables in the initial phase, and sets this as the initial oracle. After receiving $\mathcal{V}$'s challenge, it creates the response witness encoding $U^{\mathbf{w}_2} \in L^{m_2}$ from the v_2 newly added hidden variables. As in original Ligero, it also creates encodings U^x, U^y, and $U^z \in L^{m'}$ needed for testing quadratic constraints (where m' is set so that $m'\ell$ is at least the number of quadratic constraints). $\mathcal{P}$ sets the response oracle as the vertical concatenation of $U^{\mathbf{w}_2}$, U^x, U^y, U^z, along with all revealed variables in the clear.

When doing the interactive testing phase, $\mathcal{P}$ also creates $U^{\mathbf{w}_3} \in L^{m_3}$ which contains the revealed variables added in the response phase. During this phase, $\mathcal{P}$ treats its witness encoding $U^{\mathbf{w}}$ as the vertical concatenation of $U^{\mathbf{w}_1}$, $U^{\mathbf{w}_2}$, and $U^{\mathbf{w}_3}$. The verifier will do the same with the revealed variables.

Our new BooLigero tests rely on executing the interactive testing phase of Ligero tests on $U^{\mathbf{w}}$. (The encodings of x, y, and z needed for **Test-Quadratic-Constraints-IRS** are also built relative to the full $\mathbf{w}$.) They also add additional linear and quadratic constraints to be tested in this way. These tests may be useful in frameworks outside BooLigero as well.

Adding linear constraints. Ligero's **Test-Linear-Constraints-IRS** checks whether an encoding of a secret vector x is a solution to linear equation $Ax = b$, where A is a public matrix and b is a public vector. In the context of testing the protocol in a full circuit, x is the witness vector, b is the all 0s vector, and A is set so that the jth row of Ax equals $in_1 + in_2 - out$, where the jth addition gate in the circuit computes $out = in_1 + in_2$. To add an additional linear constraint, we simply add an additional row to A along with an additional element to b. Doing so does not affect the proof size.

Adding quadratic constraints. Ligero's **Test-Quadratic-Constraints-IRS** tests whether encodings of vectors x, y, z meet the condition that $x \cdot y + a \cdot z = b$, where $\cdot$ represents element-wise multiplication in $\mathbb{F}$. When using the protocol for testing a circuit, the x, y, z vectors are built so that their jth entries are in_1, in_2, out, where the jth multiplication gate in the circuit computes $out = in_1 * in_2$. These vectors are constructed in a public way from the witness, i.e. $\mathcal{P}$ and $\mathcal{V}$ both construct P_x such that $x = P_x \mathbf{w}$. Unlike the linear constraint test, separate encodings of x, y, and z must be provided to the verifier; thus, increasing the number of quadratic constraints increases the proof size.

4.10.2 Testing linear operations that yield zero over bits

We first define a useful class of tests that can be batched very efficiently.

Let ℓ_1 and ℓ_2 be positive integers, and let $t_1 = \ell_1 w$ and $t_2 = \ell_2 w$. Let $T \in \{0,1\}^{t_2 \times t_1}$ be a public $t_2 \times t_1$ binary matrix. Then T defines a test on $x \in \{0,1\}^{t_1}$ which checks whether $Tx = \vec{0}$, where $\vec{0}$ is of length t_2.

To incorporate this into Ligero, we observe that one can represent a vector of Ligero variables $x \in \mathrm{GF}(2^w)^{\ell_1}$ as a vector in $\{0,1\}^{t_1}$ of $t_1 = \ell_1 w$ bits. Adding two variables in one of these representations exactly corresponds to adding the variables in the other representation. So, we abuse notation and treat the vector $x \in \mathrm{GF}(2^w)^{\ell_1}$ as vector in $\mathrm{GF}(2)^{t_1}$.

Observe that, given $a \in \{0,1\}^{t_1}$ (which can also be represented by a vector in $\mathrm{GF}(2^w)^{\ell_1}$) such that $Ta = \vec{0}$, this implies that $T(x+a) = \vec{0}$ if and only if $Tx = \vec{0}$. In the full protocol, rather than guaranteeing that Ta will be 0, we will write a test that will check whether *both* Ta and Tx are 0 simultaneously. This is the same idea as sacrificing in multi-party computation.

To achieve privacy, we blind any Ligero variables we wish to test with T. $\mathcal{P}$ will generate a random a subject to the constraint that $Ta = \vec{0}$ and then open the variable $(x+a)$ directly to the verifier, who can independently check that $T(x+a) = \vec{0}$. Figure 4·5 shows a construction for a perfect zero-knowledge protocol between $\mathcal{P}$ and $\mathcal{V}$ to test T for a batch of N variables with low soundness error. Observe that the communication complexity for this batched test of N ℓ_1-tuples is only the size of one tuple: ℓ_1 elements of $\mathrm{GF}(2^w)$. Note that it is also independent of t_2; it depends only on t_1 and w.

We will embed this construction into BooLigero to test properties discussed in §4.10.3, such as rearranging bits in a "pattern," checking whether certain bits are zero, or both at the same time.

Note that the test itself is not sound without the additional tests provided by Ligero – the soundness of the main part of the test depends on the revealed variables being well-formed. We ensure that the all variables are well-formed by using **Test-Linear-Constraints-IRS** and **Test-Interleaved**, and ensuring that the initial elements are provided in an oracle (or committed to) before receiving the challenge. Note also that sometimes T itself will reveal certain information about x – for example, that x is 0 at certain bit locations. But the protocol will not reveal anything about x other than the fact that $Tx = \vec{0}$, which is already true if $\mathcal{P}$ is honest.

Lemma 4.10.2.1 (Security of **Test-T**). *The protocol described in Fig. 4·5 is com-*

Test-$T(\kappa, \mathbb{F}; x_1 = (x_{11}, \ldots, x_{1t}), \ldots, x_N = (x_{N1}, \ldots, x_{Nt}))$

Auxiliary input: Soundness parameter κ. Field GF(2^w) Positive integers ℓ_1 and ℓ_2; let $t_1 = w\ell_1$ and $t_2 = w\ell_2$. Binary matrix $T \in \{0,1\}^{t_2 \times t_1}$.
Inputs: A batch of N secret variables in GF$(2^w)^{\ell_1}$ held by $\mathcal{P}$, $x_1 = (x_{11}, \ldots, x_{1\ell_1})$, ..., $x_N = (x_{N1}, \ldots, x_{N\ell_1}) \in$ GF$(2^w)^{\ell_1}$. where $\mathcal{P}$ claims that (abusing notation and treating each x_i as a binary t_1-vector) $Tx_i = \vec{0}$ for all $i \in [N]$.
Protocol:

1. Initial phase: For $j \in [\kappa]$, $\mathcal{P}$ picks and adds hidden variable $a^{(j)} \in$ GF$(2^w)^{\ell_1}$ such that $Ta^{(j)} = \vec{0}$ to the witness.

2. Challenge phase: $\mathcal{V}$ sends $N\kappa$ bits: $r_i^{(j)}$ for $i \in [N]$ for $j \in [\kappa]$.

3. Response phase: $\mathcal{P}$ adds

$$u^{(j)} = a^{(j)} \oplus \bigoplus_{\substack{i \in [N] \\ \text{s.t.} \\ r_i^{(j)} = 1}} x_i \tag{4.7}$$

   for $j \in [\kappa]$ (a total of $\ell_1\kappa$ elements) as revealed elements to the witness. $\mathcal{P}$ and $\mathcal{V}$ add the κ instances of Eqn. 4.7 to the list of linear constraints to check.

4. Interactive testing phase: $\mathcal{P}$ and $\mathcal{V}$ run **Test-Linear-Constraints-IRS** and **Test-Interleaved**. $\mathcal{V}$ accepts if both tests pass and additionally (abusing notation and treating each $u^{(j)}$ as a binary t_1-vector) $Tu^{(j)} = \vec{0}$ for all $j \in [\kappa]$.

Figure 4·5: *Test construction for any binary matrix T*

plete, perfect zero-knowledge, and has soundness error $1/2^\kappa + \delta_1 + \delta_2$*, where* δ_1 *is the soundness error of* ***Test-Linear-Constraints-IRS*** *and* δ_2 *is the soundness error of* ***Test-Interleaved****.*

Proof. We prove each property in turn:

Completeness: If the prover is honest, then $Tx_i = 0$ for all $i \in [N]$, and $Ta^{(j)} = 0$ for all $j \in [\kappa]$. Thus, clearly $Tu = T(a^{(j)} \oplus \bigoplus_{i \in [N] \text{s.t.} r_i^{(j)} = 1} x_i) = 0$ for all $j \in [\kappa]$ as desired.

Honest verifier zero-knowledge: We know by [10] Lemma 4.13 that aside from the added revealed variables u, the remainder of the protocol is honest-verifier perfect zero knowledge. Consider the additional u variables. Each of these is "masked" by a random element a which is known only to $\mathcal{P}$. Thus, u will have the same distribution as a fresh random binary vector for which $Tu = 0$. Thus, a simulated version of the protocol which returns a random v for which $Tv = 0$, which accurately simulates the real execution which returns u.

Soundness: Suppose the prover is lying and there exists at least one $i \in [N]$ for which $Tx_i \neq \vec{0}$. Without loss of generality, let $i = 1$ be one such index. If the Ligero matrix itself is malformed, this will be caught with probability $1 - \delta_2$ by **Test-Interleaved**. Assuming it is not malformed, if any of the $u^{(j)}$ values are not equal to those given in Equation 4.7, then a linear constraint has been violated and this will be caught with probability $1 - \delta_1$ by **Test-Linear-Constraints-IRS**. We assume $u^{(j)}$ is correct for now. Let $c^{(j)} = a^{(j)} \oplus \bigoplus_{\substack{i \in \{2,\ldots,N\} \\ \text{s.t.} \\ r_i^{(j)}=1}} x_i$ (note the indexing beginning at 2). Thus, for the remainder of this proof, we assume u is well-formed. If $r_1^{(j)} = 0$, then $c^{(j)} = u^{(j)}$. If $r_1^{(j)} = 1$, then $c^{(j)} = u^{(j)} \oplus x_1$. Consider $Tc^{(j)}$. There are two cases:

> *Case 1.* $Tc^{(j)} = \vec{0}$. If $r_1^{(j)} = 0$, then $Tu^{(j)} = Tc^{(j)} = \vec{0}$ and the prover successfully cheats. If, however, $r_1^{(j)} = 1$, then $Tu^{(j)} = T(c^{(j)} + x_1) = \vec{0} + Tx_1 \neq \vec{0}$. So $\mathcal{V}$ correctly rejects in this case.
>
> *Case 2.* $Tc^{(j)} \neq \vec{0}$. If $r_1^{(j)} = 0$, then $Tu^{(j)} = Tc^{(j)} \neq \vec{0}$ so $\mathcal{V}$ correctly rejects. If $r_1^{(j)} = 1$, then $Tu^{(j)} = T(c^{(j)} + x_1)$. This may equal 0, so the prover may successfully cheat.

In each case, the prover is caught cheating with probability $1/2$ over the choice of $r_1^{(j)}$. Note that this continues to hold regardless of how many indices within $[N]$ the prover chooses to cheat on. Repeating this for κ choices of r_1, the chance that the prover successfully cheats in all of them is $1/2^\kappa$. Union bounding this with the chance of failure of **Test-Linear-Constraints-IRS** and **Test-Interleaved**, the soundness error of this protocol is $1/2^\kappa + \delta_1 + \delta_2$, as desired. □ □

4.10.3 New constraint tests

In this section, we describe our added tests for BooLigero. Each of these calls one of the original Ligero tests **Test-Linear-Constraints-IRS** or **Test-Quadratic-Constraints-IRS**. The later tests additionally call on the earlier BooLigero tests as well.

Properties tested by Test-T. We proceed to name two useful properties (and their conjunction) that can be tested using the construction from the previous section. As described in the previous section, they can test the property on arbitrarily many input variables for only a constant overhead over the cost of the variables themselves. Since we will reuse them later, we name each of these special cases of **Test-T**.

- **Test-Zeros-Constraints**: This tests whether particular bit locations in the input are 0. Let $Z \subseteq [t_2]$ be a set of indices to be zero-tested. Formally, let T_Z be a square matrix with 1s on the diagonal for indices in Z, and 0 for all other elements. Observe that $T_Z x = \vec{0}$ if and only if x is 0 at the Z indices.

- **Test-Pattern-Constraints**: We informally define a "pattern" as a relationship between $(t_i = t_1 - t_2)$ "input bits" and t_2 "output bits." The pattern property enforces that each "output bit" is an XOR of some subset of the input bits. In general, pattern matrices T_π are defined as a matrix that is a concatenation between a matrix $\pi \in \{0,1\}^{t_2 \times t_i}$ and a $t_2 \times t_2$ identity matrix. Several useful functions can be defined as patterns:

  - Masking. Suppose we wished to show in Ligero that $x \& \mu = y$ for some public mask μ, for Ligero variables $x, y \in \mathrm{GF}(2^w)^{\ell_2}$ and mask $\mu \in \{0,1\}^{t_2}$. Let M be the t_2-square matrix with μ comprising the diagonal and zeros elsewhere. Then we can test whether $x \& \mu = y$ using the pattern $T_\mu = [\ M \mid I\ ]$, because:

    $$\left[\ \ M \ \mid\ I\ \ \right] \left[\begin{array}{c} x \\ \hline y \end{array}\right] = \vec{0}$$

    which, for diagonal matrix M, implies that $Mx = y$.

  - Even parity. Suppose we wished to show that $y \in \mathrm{GF}(2^w)$ is the parity of

$x \in \mathrm{GF}(2^w)^{\ell_1 - 1}$. This can be tested by checking that

$$\left[\begin{array}{ccc|c} \ddots & & \iddots & \\ \cdots & 0 & \cdots & I \\ 1 & \cdots & 1 & \end{array}\right] \left[\begin{array}{c} x \\ \hline y \end{array}\right] = \vec{0}$$

which will check that the least significant bit of y equals a sum of all bits of x. Even-parity can be batch-tested by using Zeros, testing the parity of many variables simultaneously.

- **Test-Pattern-Zeros-Constraints**: Notice that a Zeros test can be performed on the same revealed values as a pattern test, if the blinding variables are chosen to meet both constraints. We will often perform these tests on the same revealed variables to save space.

Bitwise AND test. Next, we describe our test for bitwise AND. Note that AND cannot be tested using the method in the previous section, since it is not a linear operation. Instead, we write a new test that calls **Test-Pattern-Zeros-Constraints**.

As a first step, one way to test AND would be to fully bit-decompose our single w-bit element into w elements each representing a single bit. This would let us use quadratic constraints directly to show AND constraints. However, doing so is expensive. This method would yield roughly the same proof size as original Ligero, since it uses an entire w-bit element to represent a single bit. Instead, we exploit the nature of Galois field arithmetic to compute the AND of $w_0 = \lfloor \sqrt{w} \rfloor$ bits simultaneously in a w-bit element using a GF multiplication. We then use our Pattern test to convert between the original variables and the w_1 decomposed variables, where w_1 is the minimum integer such that $w_0 w_1 \geq w$. Each of these w_1 "split" variables contains w_0 bits of the original element (except the last, which may contain fewer if $w_0 \not| w$).

Suppose we have elements x, $y \in \mathrm{GF}(2^w)$, and want to find $z = x \& y$. We start by using a Pattern to split x into w_1 variables $\hat{x}_1, \ldots, \hat{x}_{w_1}$, and to split y into w_1 variables $\hat{y}_1, \ldots, \hat{y}_{w_1}$. First, consider the x variables. Each split variable $\hat{x}_h$ will consist of w_0 chunks of w_0 bits each. (Recall that by construction $w_0^2 \leq w$.) The least significant bit of each chunk will be a bit of x, and all other bits will be 0. This is illustrated in Equation 4.8. The y variables will be split differently: each $\hat{y}_h$ will consist of a single chunk of w_0 bits from y, starting with the least significant bit. This is shown in Equation 4.9.

We then set $\hat{z}_h = \hat{x}_h * \hat{y}_h$. The effect of multiplying $\hat{x}_h$ by $\hat{y}_h$ is that the chunk of w_0 bits in $\hat{y}_h$ is "copied" to each of the w_0 chunks of output for which the corresponding chunk of $\hat{x}_h$ was 1. Thus, in order to figure out which bits were shared between x and y, we go to the kth bit of the kth chunk of $\hat{z}_h$. This will equal the kth bit of $\hat{y}_h$ times the LSB of the kth chunk of $\hat{x}_h$. This is shown in Equation 4.10. Recomposing from the $\hat{z}_h$ variables back to z by using Pattern once again, we have exactly computed the bitwise AND of x and y.

Figure 4·6 shows an example of how to split (x, y, z), where $z = x \& y$. The full **Test-And-Constraints** procedure is shown in Figure 4·7. The patterns π_x, π_y, and π_z, described formally in Figure 4·7 step 1(d).

Lemma 4.10.3.1 (Security of **Test-And-Constraints**). *The protocol described in Fig. 4·7 is complete, perfect zero-knowledge, and has soundness error* $3(1/2^\kappa) + \delta_1 + \delta_2 + \delta_3$*, where* δ_1 *is the soundness error of* ***Test-Quadratic-Constraints-IRS****,* δ_2 *is the soundness error of* ***Test-Linear-Constraints-IRS****, and* δ_3 *is the soundness error of* ***Test-Interleaved****.*

Proof. We must show that **Test-And-Constraints** is complete, zero-knowledge, and sound up to error $3(1/2^\kappa) + \delta_1 + \delta_2 + \delta_3$, where δ_1 is the soundness error of **Test-Quadratic-Constraints-IRS**, δ_2 is the soundness error of **Test-Linear-Constraints-IRS**, and δ_3 is the soundness error of **Test-Interleaved**.

Completeness: If $\mathcal{P}$ is honest, then all variables are well-formed. We must show that following the process described in step 1(b) of Fig. 4·7 will lead to computing bitwise AND. Elements in GF(2^w) are polynomials over GF(2) of degree at most $(w-$

1), and multiplication in GF(2^w) is polynomial multiplication modulo an irreversible polynomial. As in step 1(a), let $w_0 = \lfloor \sqrt{w} \rfloor$. Fix $i \in [N]$.

By construction, the polynomial representations of all $\hat{y}_{i,h}$ variables (for $h \in [w_1]$) have degree at most $w_0 - 1$. They can be written as $\sum_{k=0}^{w_0-1} c_k v^k$, where v is the polynomial variable and c is the coefficient (either 0 or 1).

Further, the $\hat{x}_{i,h}$ variables are of the form $\sum_{k=0}^{w_0-1} d_k v^{kw_0}$ (now using d as the coefficient).

Thus, if we multiply $\hat{x}_{i,h} * \hat{y}_{i,h}$, the result can be written as:

$$\begin{aligned}
\hat{z}_{i,h} = \hat{x}_{i,h} * \hat{y}_{i,h} &= \left(\sum_{k=0}^{w_0-1} d_k v^{kw_0}\right)\left(\sum_{k=0}^{w_0-1} c_k v^k\right) \\
&= d_0\left(\sum_{k=0}^{w_0-1} c_k v^k\right) + d_1\left(\sum_{k=0}^{w_0-1} c_k v^{w_0+k}\right) + \ldots + d_{w_0-1}\left(\sum_{k=0}^{w_0-1} c_k v^{(w_0-1)w_0+k}\right) \\
&= \left(d_0 c_0 v^0 + \ldots + d_0 c_{w_0-1} v^{w_0-1}\right) \\
&\qquad\qquad + \left(d_1 c_0 v^{w_0} + \ldots + d_1 c_{w_0-1} v^{2w_0-1}\right) \\
&\qquad\qquad + \ldots + \left(d_{w_0-1} c_0 v^{(w_0-1)w_0} + \ldots + d_{w_0-1} c_{w_0-1} v^{w_0^2-1}\right) \\
&= \sum_{k=0}^{w_0^2-1} d_{\lfloor k/w_0 \rfloor} c_{(k \bmod w_0)} v^k
\end{aligned}$$

First, notice that the degree of this polynomial is at most $w_0^2 - 1$, so by construction, this polynomial will not need to be reduced modulo the irreducible polynomial. Next, notice that the coefficient e_k of v^k can be written as $e_k = d_{\lfloor k/w_0 \rfloor} c_{(k \bmod w_0)}$. But remember that the c and d coefficients correspond to the bits of $\hat{x}_{i,h}$ and $\hat{y}_{i,h}$, which in turn correspond to the bits of x_i and y_i. So if we wish to know the AND of $c_{k'}$ and $d_{k'}$, we can look at the coefficient of v^k, for the k for which $k' = \lfloor k/w_0 \rfloor = (k \bmod w_0)$, This will occur at $k = k'w_0 + k'$. Thus, each $\hat{z}_{i,h}$ can be used to find the AND of w_0 bits. For $k' \in \{0, \ldots, w_0 - 1\}$, bit $\hat{z}_{i,h}[1 + k' + k'w_0]$ is the AND of $\hat{x}_{i,h}[1 + w_0 k']$ and $\hat{y}_{i,h}[1 + k']$.

Zooming back out to z_i, we find that each bit of z_i can be found as $z_i[k] = \hat{z}_{i,\lfloor \frac{k+1}{w_0} \rfloor}[1 + ((k-1) \bmod w_0) + w_0((k-1) \bmod w_0)]$. Since the $\hat{z}_{i,h}$ variables were formed correctly from the $\hat{x}_{i,h}$ and $\hat{y}_{i,h}$ variables, which were formed correctly from x_i and y_i, z_i will be the AND of x_i and y_i for all $i \in [N]$, as desired.

Zero-knowledge: The hidden variables are zero-knowledge for the same reason the original witness variables are. The revealed variables added as part of **Test-Pattern-**

Zeros-Constraints in step 3 can be simulated perfectly with random sets of elements that meet R_x, R_y, and R_z, as described in **Test-Pattern-Zeros-Constraints**.

Soundness: Suppose $\mathcal{P}$ is cheating, that is, there is at least one (x_i, y_i, z_i) triple for which $z_i \neq x_i \& y_i$. Without loss of generality, let $i = 1$ be an index on which the prover cheats.

If the Ligero matrix is not well-formed, **Test-Interleaved** will fail with probability at least $1 - \delta_3$; we assume this is not the case for the remainder of the proof.

If $z_1 \neq x_1 \& y_1$, then one of the following must be true:

1. There exists an $h \in [w_1]$ for which $\hat{z}_{1,h} \neq \hat{x}_{1,h} * \hat{y}_{1,h}$.

2. The $\hat{x}_{1,h}$ variables were not properly formed from x_1. In other words,

$$T_{\pi_x}[x_1, \hat{x}_{1,1}, \ldots, \hat{x}_{1,w_1}, x_1]^{\perp} \neq \vec{0}.$$

   The same may be true for T_{π_y} on the y variables, or T_{π_z} on the z variables.

If the former is true, then **Test-Quadratic-Constraints-IRS** will fail with probability at least $1 - \delta_1$. If the latter is true, then either **Test-Linear-Constraints-IRS** will fail with probability at least $1 - \delta_2$, or the pattern-checking part of **Test-Pattern-Zeros-Constraints** for R_x will fail with probability at most $1/2^{\kappa}$. Similarly for R_y and R_z.

Thus, by a Union bound, the overall protocol has soundness error $3(1/2^{\kappa}) + \delta_1 + \delta_2 + \delta_3$ over the verifier's coins. □

Additionally, the $\hat{z}$ variables can be used to compute bitwise outer product if desired.

4.10.4 Range tests

We now show how to do cheap range tests for power-of-two ranges, and how to build an adder and use it to perform non-power-of-two range tests.

Power-of-two range tests Our Zeros test can be used to very cheaply test range tests where the range is a power of two, and can be used combined with an Add-with-

$$x = 10110 \to (\hat{x}_{w_1} = 000\,\overbrace{0\underline{1}}^{w_0}, \hat{x}_2 = 0\,\overbrace{0\underline{0}}^{w_0}\,\overbrace{0\underline{1}}^{w_0}, \hat{x}_1 = 0\,\overbrace{0\underline{1}}^{w_0}\,\overbrace{0\underline{0}}^{w_0}) \tag{4.8}$$

$$y = 11101 \to (\hat{y}_{w_1} = 0000\,\overbrace{1}, \hat{y}_2 = 000\,\overbrace{1}\,\overbrace{1}, \hat{y}_1 = 000\,\overbrace{0}\,\overbrace{1}) \tag{4.9}$$

$$\updownarrow \qquad\qquad \updownarrow \qquad\qquad \updownarrow \quad \hat{z}_h = \hat{x}_h * \hat{y}_h$$

$$z = 10100 \leftarrow (\hat{z}_{w_1} = 000\,\overbrace{0\underline{1}}^{w_0}, \hat{z}_2 = 0\,\overbrace{\underline{0}0}^{w_0}\,\overbrace{1\underline{1}}^{w_0}, \hat{z}_1 = 0\,\overbrace{\underline{0}1}^{w_0}\,\overbrace{0\underline{0}}^{w_0}) \tag{4.10}$$

Figure 4·6: *Example variable splits for **Test-And-Constraints** for $w = 5$, $w_0 = 2$, $w_1 = 3$. Pattern constraints enforce the relationship between x and $\hat{x}$, and similar for y and z. The $\hat{z}$ variables are related to $\hat{x}$ and $\hat{y}$ via a quadratic constraint.*

modulus gadget (which makes use of AND and Pattern) to make a non-power-of-two range test.

To check whether a variable $x < 2^a$ for some $a \leq w$, we simply use the Zeros test with $Z = \{k : k > a\}$, to see whether the higher order bits of x are 0. This scales very efficiently – to check whether N variables are all less than the same power of 2, the Zeros test can be applied to all N variables and costs only κ hidden and κ revealed variables (independent of N).

Note that power-of-two range tests in $\mathbb{F}_{2^w}$ are especially useful in financial use cases that require proving numbers are non-negative, i.e. the arithmetic addition or multiplication of two such numbers never wraps around. The addition case can be done by showing that the two numbers are each less than 2^{w-1}, and the multiplication case can be done by showing that the two numbers are each less than $2^{\lfloor w/2 \rfloor}$ (assuming this is an acceptably loose upper bound).

Non-power-of-two range tests. Suppose we wish to show that $x < n$ for some non-power-of-two $n < 2^w - 1$. Let a be the integer such that 2^a is the smallest power of 2 greater than n. A range proof to ensure $x < n$ can be done by showing that both x and $(x - n + 2^a)$ have 0 for their $w - a$ MSBs, in other words, they are less than 2^a. Here, $+$ and $-$ denote arithmetic addition and subtraction.

Test-And-Constraints$(\mathbb{F} = \mathrm{GF}(2^w), \kappa; x_1, \ldots, x_N, y_1, \ldots, y_N, z_1, \ldots, z_N)$

Inputs: Soundness parameter κ. Secret variables $x_1, \ldots, x_N, y_1, \ldots, y_N, z_1, \ldots, z_N \in \mathbb{F}$ where $\mathcal{P}$ claims that $x_i \& y_i = z_i$ for all $i \in [N]$.

Constraint enforced: $x_i \& y_i = z_i$ for all $i \in [N]$.

Procedure:

1. Initial phase:
   (a) Let $w_0 = \lfloor \sqrt{w} \rfloor$. Let w_1 be the minimum integer such that $w_0 w_1 \geq w$. (Note that $w_1 \leq w_0 + 2$.)
   (b) Add $3Nw_1$ new variables to the witness as described below.
      i. For $i \in [N]$, for $h \in [w_1]$, add new variable $\hat{x}_{i,h}$ where $\hat{x}_{i,h}[(k-1)w_0 + 1] = x_i[((h-1)w_0) + k]$ for $k \in [w_0]$ (if the index is defined), and all other bits are 0. An example is shown in Equation 4.8.
      ii. For $i \in [N]$, for $h \in [w_1]$, add new variable $\hat{y}_{i,h}$ where $\hat{y}_{i,h}[k] = x_i[((h-1)w_0) + k]$ for $k \in [w_0]$ (if the index is defined), and all other bits are 0. An example is shown in Equation 4.9.
      iii. For $i \in [N]$, for $h \in [w_1]$, add new variable $\hat{z}_{i,h} = \hat{x}_{i,h} * \hat{y}_{i,h}$, where $*$ denotes multiplication in $\mathrm{GF}(2^w)$. Observe that within $\hat{z}_{i,h}$, each of w_0 "chunks" of w_0 bits, the kth bit of the kth chunk is 1 if and only if the corresponding bits in x_i and y_i are 1. An example is shown in Equation 4.10.
   (c) For all $i \in [N], h \in [w_1]$, add a quadratic constraint that $\hat{x}_{i,h} * \hat{y}_{i,h} = \hat{z}_{i,h}$.

Continued in Fig. 4·7b.

(a)

Figure 4·7: *Witness modification procedure and costs for* ***Test-And-Constraints*** *(continued in Fig. 4·7b)*

This motivates the creation of an adder. A gadget to add x and y in a power-of-two modulus of at most 2^{w-1} can be created by writing constraints on an additional "carry" variable c based on a ripple-carry adder. AND and Pattern can be combined to form a carry variable c such that $c[i] = \mathsf{Majority}(c[i-1], x[i-1], y[i-1])$ for $i \in [w]$, and $c[0] = 0$ by convention. The XOR of c with x and y create the z variable. This can be used to set up the $(x - n + 2^a - 1)$ equation above, and then the Zeros test can be used to ensure that both that and x are less than 2^a.

Our full modifications to the protocol from [10] are shown in Figure 4·8.

4.11 Performance

We primarily evaluate our proof on its size compared to original Ligero, since our asymptotic prover and verifier runtime should be the same as Ligero. Recall that a

Continued from Fig. 4·7a

1. Initial phase, continued from Fig. 4·7a
   (d) We define patterns π_x, π_y, π_z which describe the relationship between the variables and their "hatted" versions described in 1(b) and with examples in Equations 4.8, 4.9, and 4.10.
      i. T_{π_x} enforces the following on column-vector $[x_i, \hat{x}_{i,1}, \ldots, \hat{x}_{i,w_1}]$:
         A. For all $h \in [w_1]$, $\hat{x}_{i,h}$ is 0 everywhere except at indices $(1 + kw_0)$ for all valid k. Additionally, $\hat{x}_{i,w_1}$ is also 0 at indices where $k + 1 > w \bmod w_0$.
         B. For $k \in [w]$, $x_i[k] = \hat{x}_{i,\lfloor \frac{k+1}{w_0} \rfloor}[1 + w_0((k-1) \bmod w_0)]$
      ii. T_{π_y} enforces the following on column-vector $[y_i, \hat{y}_{i,1}, \ldots, \hat{y}_{i,w_1}]$:
         A. For all $h \in [w_1]$, $\hat{y}_{i,h}$ is 0 everywhere except at indices $1, \ldots, w_0$. Additionally, $\hat{y}_{i,w_1}$ is also 0 at indices greater than $w \bmod w_0$.
         B. For $k \in [w]$, $y_i[k] = \hat{y}_{i,\lfloor \frac{k+1}{w_0} \rfloor}[k \bmod w_0]$
      iii. T_{π_z} enforces the following on column-vector $[z_i, \hat{z}_{i,1}, \ldots, \hat{z}_{i,w_1}]$:
         A. For all $h \in [w_1]$, $\hat{z}_{i,h}$ is 0 at all indices greater than w_0^2. Additionally, $\hat{z}_{i,w_1}$ is also 0 at indices greater than $w_0(w \bmod w_0)$.
         B. For $k \in [w]$, $z_i[k] = \hat{z}_{i,\lfloor \frac{k+1}{w_0} \rfloor}[1 + ((k-1) \bmod w_0) + w_0((k-1) \bmod w_0)]$

      Run the initial phase of 3 **Test-Pattern-Zeros-Constraints** tests, one for each of these predicates, using all of the corresponding variables from 1(b) as input. That is, do a batch test for T_{π_x}, T_{π_y}, and T_{π_z} for all $i \in [N]$. Each test's initial phase adds hidden $(w_1 + 1)\kappa$ elements, for a total of $3(w1 + 1)\kappa$.
2. Challenge phase: Run the challenge phase of each of the three **Test-Pattern-Zeros-Constraints**, which involves picking $N\kappa$ bits for each test.
3. Response phase: Run the response phase of each of the three **Test-Pattern-Zeros-Constraints**, which will add $(w_1 + 1)$ revealed variables and some linear constraints on them.
4. Interactive testing phase: Run **Test-Quadratic-Constraints-IRS** on the constraints described in 1(c), and run the interactive testing phase of **Test-Pattern-Zeros-Constraints**, which involves running **Test-Linear-Constraints-IRS** and ensuring that **Test-Pattern-Zeros-Constraints** passes using the revealed variables.

(b)

Figure 4·7: *Witness modification procedure and costs for* ***Test-And-Constraints*** *(continued from Fig. 4·7a)*

proof for a Boolean circuit in original Ligero requires using an entire field element to represent a single bit value on a wire. Like original Ligero, the parameters for BooLigero can be set so that the proof size is $O(\sqrt{s})$ elements, where s is the circuit size. If the field size in Ligero is $b = \lceil \log \mathbb{F} \rceil$ bits, then we would expect BooLigero to save a factor of $O(\sqrt{b})$ in the proof size. For example, if a Ligero proof used a field with 18 bits, we would expect a $\sqrt{18} \approx 4.2\times$ improvement in the BooLigero proof size. For AND gates we require $w_1 \approx \sqrt{w}$ variables to compute the AND of a single

Protocol ZKIOP$(C, \mathbb{F} = \mathrm{GF}(2^w))$

- **Input:** The prover $\mathcal{P}$ and the verifier $\mathcal{V}$ share a common input circuit $C : \mathrm{GF}(2^w)^{n_i} \to \mathrm{GF}(2^w)$ and input statement x. $\mathcal{P}$ additionally has input $\overline{\alpha} = (\alpha_1, \ldots, \alpha_{n_i})$ such that $C(\overline{\alpha}) = 1$.
- **Initial oracle:** Let v_1 be the total number of variables added by $\mathcal{P}$ in the initial phase all BooLigero tests. Let v_2 and v_3 be the number of hidden and revealed variables added by $\mathcal{P}$ in the response phase all BooLigero tests. The variables themselves cannot be known until the challenge, but the number is fixed. Let m_1, m_2, m_3, ℓ be integers such that $m_1 \cdot \ell > n_i + s + v_1$, $m_2 \cdot \ell > v_2$, and $m_3 \cdot \ell > v_3$, where s is the number of gates in the circuit. $\mathcal{P}$ generates an extended witness $\mathbf{w}_1 \in \mathbb{F}^{m_1 \ell}$ where the first $n_i + s$ entries of $\mathbf{w}$ are $(\alpha_1, \ldots, \alpha_{n_i}, \beta_1, \ldots, \beta_s)$ where β_i is the output of the ith gate when evaluating $C(\overline{\alpha})$. The next v_1 variables are those for the **initial phase** section of all BooLigero tests. Let $L = \mathsf{RS}_{\mathrm{GF}(2^w),n,k,\eta}$, and let $\zeta = (\zeta_1, \ldots, \zeta_\ell)$ be a sequence of distinct elements disjoint from $\eta_1, \ldots, \eta_n$. The prover samples random codeword $U^{\mathbf{w}_1} \in L^{m_1}$ subject to $\mathbf{w}_1 = \mathsf{Dec}_\zeta(U^{\mathbf{w}_1})$.
- **Challenge:** $\mathcal{V}$ chooses and sends random bits as described in the **challenge phase** section of all BooLigero tests.
- **Response oracle:** $\mathcal{P}$ generates witness extension $\mathbf{w}_2 \in \mathbb{F}^{m_2 \ell}$ where the first v_1 variables in $\mathbf{w}_2$ are the hidden variables for the **response phase** section of all BooLigero tests. $\mathcal{P}$ also generates witness extension $\mathbf{w}_3 \in \mathbb{F}^{m_3 \ell}$, where the first variables in $\mathbf{w}_3$ are the revealed variables for the **response phase** section of all BooLigero tests. Let $m = m_1 + m_2$ and let $m' = m_1 + m_2 + m_3$. Let $\mathbf{w} \in \mathbb{F}^{\ell}$ be the vertical concatenation of $\mathbf{w}_1$ and $\mathbf{w}_2$. Let $\mathbf{w}' \in \mathbb{F}^{(m_1+m_2+m_3)\ell}$ be the vertical concatenation of $\mathbf{w}$ and all revealed variables. $\mathcal{P}$ deterministically chooses codeword $U^{\mathbf{w}_3} \in L^{m_3}$. $\mathcal{P}$ samples random codeword $U^{\mathbf{w}_2} \in L^{m_2}$ subject to $\mathbf{w}_2 = \mathsf{Dec}_\zeta(U^{\mathbf{w}_2})$ and sets $U^{\mathbf{w}} \in L^{m}$ to be the vertical concatenation of $U^{\mathbf{w}_1}$ and $U^{\mathbf{w}_2}$.

Continued in Fig. 4·8b

(a)

Figure 4·8: *ZKIOP of [10] with our modifications shown in blue (continued in Fig. 4·8b)*

w-bit variable. If the Ligero and BooLigero field sizes require the same number of bits to represent, BooLigero will use $\sqrt{b}$ fewer variables, a proof size improvement of $O(b^{1/4})$ for AND-heavy circuits. We also add a small constant up-front cost for the revealed variables.

Determining the size of the extended witness and proof. If the verification circuit C consists of only XOR gates, NOT gates, and Galois field multiplications, then the size of the witness $\mathbf{w}$ is simply the number of wires in C, which we call v_0 as described in §4.10.1. If C contains ANDs, or uses any other BooLigero test (e.g. using

Test-Pattern-Constraints to perform a bit shift), then $\mathbf{w}$ is augmented with the ($v_1 + v_2$ hidden and v_3 revealed) variables described in §4.10.3. The combined proof oracle becomes an L^m encoding of the $v_0 + v_1 + v_2$ hidden variables in the witness, plus the v_3 revealed variables in the clear. Once the extended witness is created, the process of choosing the parameters proceeds in the same way as original Ligero: the number of rows m is balanced against the number of variables per row ℓ to achieve sublinear proof size. For more details, see [10] §5. In the non-interactive version of BooLigero, the Merkle path part of the proof is doubled since the initial and response variables were committed to separately. We computed the parameters using our own optimizer written in SciPy and validated them with an optimizer obtained from [308].

4.11.1 Concrete results

For both SHA-2 and SHA-3, we evaluate our proof sizes compared to Ligero on proving membership in the list captured by a Merkle tree. This has become a common benchmark for evaluating the scalability of zero-knowledge proofs to larger predicates. We compare the proof sizes of Ligero and BooLigero for Merkle trees of increasing size. For a Merkle tree with M leaves, $(2M - 1)$ hash computations are done.

SHA-3

SHA-3 only uses bit operations, so there is no special benefit from using an arithmetic system. Both BooLigero and Ligero may do the wordwise rotations for free; they can be achieved by re-indexing constraints for the next step. Ligero can do the bitwise rotations for free (since each variable represents only a single bit), but in BooLigero we must write the additional variable and use **Test-Pattern-Constraints** to enforce the constraint.

Using SHA-3 as the hash function in a Merkle tree, each invocation of the hash function consists of a single call to the f-function.

SHA-2

SHA-2 contains a mixture of Boolean operations and mod-2^{32} addition. Although the SHA-2 circuit used in [10] was not provided, we reconstruct a similar circuit using the same techniques. As described in [10], Ligero computes modular addition by using a dummy variable. Our SHA-2 circuit for original Ligero tracks 16 32-bit variables (11 main variables plus 5 dummy variables) throughout 64 iterations of the SHA-2 loop.

BooLigero prefers a different strategy. Although we can compute mod-2^{32} addition in BooLigero by implementing an adder, it turns out that a standard 135840-wire Boolean circuit for SHA-2 leads to a smaller proof size since it uses far fewer ANDs.

Continued from Fig. 4·8a

- **Response oracle** (continued from Fig. 4·8a): Let m'' be an integer such that $m''\ell$ is greater than the number of multiplication gates plus additional quadratic constraints in BooLigero tests. $\mathcal{P}$ constructs vectors $x, y, z \in \mathbb{F}^{m''\ell}$ where the jth entry of x, y, z contains the values $\beta_a, \beta_b, \beta_c$ corresponding to the jth multiplication gate in $\mathbf{w}$. The following entries of x, y, z contain the values for additional constraints added in BooLigero tests. $\mathcal{P}$ and $\mathcal{V}$ construct matrices $P_x, P_y, P_z \in \mathbb{F}^{m''\ell \times m''\ell}$ such that $x = P_x\mathbf{w}', y = P_x\mathbf{w}', z = P_z\mathbf{w}'$. $\mathcal{P}$ constructs matrix $P_{\mathbf{add}} \in \mathbb{F}^{m''\ell \times m''\ell}$ such that the jth row of $P_{\mathbf{add}}\mathbf{w}$ equals $\beta_a + \beta_b - \beta_c$ where β_a, β_b, and β_c correspond to the jth addition gate of the circuit in $\mathbf{w}$, and the subsequent rows correspond to additional linear constraints added in BooLigero tests. It also samples $U^x, U^y, U^z \in L^{m''}$ subject to $x = \mathsf{Dec}_\zeta(U^x)$, $y = \mathsf{Dec}_\zeta(U^y)$, and $z = \mathsf{Dec}_\zeta(U^z)$. Let $u'_h, u^x_h, u^y_h, u^z_h, u^0_h, u^{\mathbf{add}}_h$ be auxiliary rows sampled randomly from L for every $h \in [\sigma]$ where each of $u^x_h, u^y_h, u^z_h, u^{\mathbf{add}}_h$ encodes an independently sampled random ℓ messages $(\gamma_1, \ldots, \gamma_\ell)$ subject to $\sum_{c \in [\ell]} \gamma_c = 0$ and u^0_h encodes 0^ℓ. $\mathcal{P}$ sets the combined oracle as $(U \in L^{m+3m''}, R)$ where U is set as the vertical juxtaposition of the matrices $U^{\mathbf{w}} \in L^m$, $U^x, U^y, U^z \in L^{m''}$, and R is the set of all v_3 revealed variables. When the combined oracle is queried on $Q \subset [n]$, the response will be the columns of U that are in Q, as well as R sent in the clear.
- **The interactive protocol:**
  1. For every $h \in [\sigma]$, $\mathcal{V}$ sends the first verifier message of the testing process for **Test-Interleaved**, **Test-Linear-Constraints-IRS** applied to $A = P_{\mathbf{add}}, b = \vec{0}$ on $U^{\mathbf{w}}$, and **Test-Quadratic-Constraints-IRS** applied to U^x, U^y, U^z.
  2. For every $h \in [\sigma]$, $\mathcal{P}$ responds with the appropriate next step of the testing process for **Test-Interleaved**, **Test-Linear-Constraints-IRS**, and **Test-Quadratic-Constraints-IRS**.
  3. $\mathcal{V}$ picks a random set $Q \subset [n]$ of size t, and queries $U[j]$ that is the vertical juxtaposition of $U^x_h[j], U^y_h[j], U^z_h[j], U^{\mathbf{w}}_h[j], u^x_h[j], u^y_h[j], u^z_h[j], u^{\mathbf{add}}_h[j], u'_h[j]$, $j \in Q$. It also receives R, the list of revealed variables. It uses the same deterministic process as $\mathcal{P}$ to generate $U^{\mathbf{w}_3}$ and appends this to the bottom of the queried columns for testing. It accepts if all acceptance criteria for **Test-Interleaved**, **Test-Linear-Constraints-IRS**, **Test-Quadratic-Constraints-IRS**, **Test-Pattern-Zeros-Constraints** and **Test-And-Constraints** are met.

(b)

Figure 4·8: *ZKIOP of [10] with our modifications shown in blue (continued from Fig. 4·8a)*

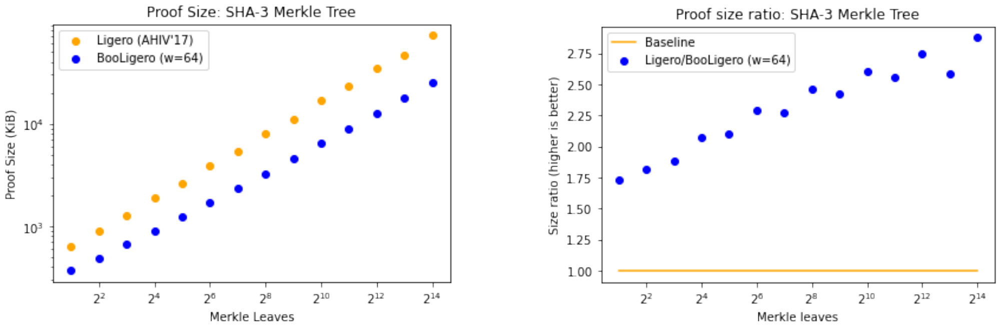

Figure 4·9: *BooLigero and Ligero absolute and relative proof sizes for SHA-3 Merkle trees*

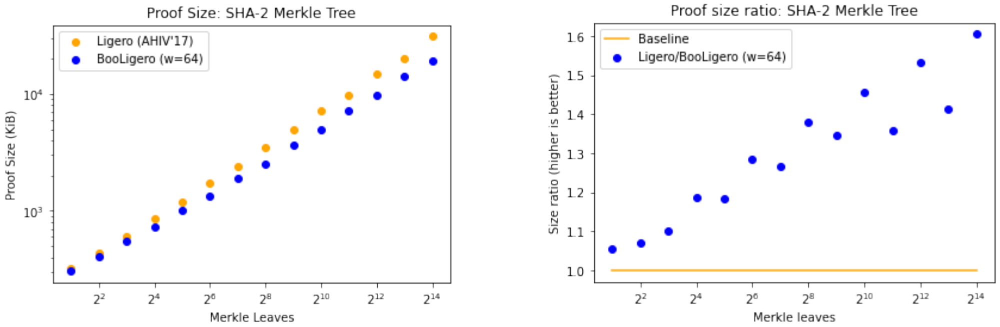

Figure 4·10: *BooLigero and Ligero absolute and relative proof sizes for SHA-2 Merkle trees*

Chapter 5

Interdisciplinary cryptographic and legal research: challenges, opportunities, and future work

It is my privilege to study at the intersection of two fields that each individually have enough of their own exciting challenges to last multiple lifetimes. Law and cryptography share several key qualities: an emphasis on adversarial thinking, a notion of "convincing" a third party via a proof or logical argument, and a desire for rigor and consistency. Moreso than most other computer science fields, cryptography pays more than a little attention to social principles: privacy, autonomy, non-non-repudiation, deniability, and so on.

Although codes and ciphers have been used for over 3,000 years [184, p. 72], this practice only began to modernize with the advent of radio in World War I and intensive code making and breaking efforts on both sides of the war in World War II [109]. Subsequently, cryptography grew into the field we know it as today in the late 1970s and early 1980s, after information theory and computational complexity had matured. Several highly influential developments emerged which would set the tone for the next 50 years: the call for proposals for (and eventual invention of) the Data Encryption Standard [235], Diffie and Hellman's *New Directions in Cryptography* [108] and the RSA algorithm [253] which introduced public key cryptography, Chaum's work on mix-nets [78] that emphasized practical privacy concerns about not only the content of electronic communications but also the metadata, Goldwasser and Micali's *Probabilistic Encryption* [145] which brought the rigor of computational complexity to cryptography by basing all cryptography on problems proven to be computationally "hard," and the birth of zero-knowledge proofs due to Goldreich, Micali, and Wigderson [143]. Diffie and Landau describe an additional revolution in cryptography in the new millennium, with deregulation that led not to major adoption of cryptography in every natural setting, but in specific areas such as protecting intellectual property and the introduction of Trusted Platform Modules to ensure the integrity of the software on computers [109]. We can add HTTPS to that list; the majority of web browsing has been encrypted on Mozilla Firefox and Google Chrome since 2017 [118]. In any case, as old as cryptography's goals are, many of its methods and impacts on society are still new.

Law, on the other hand, has always been a central part of human society, even though the mechanisms and rules change drastically from culture to culture, from time to time, and from place to place. Modern democratic legal systems include methods for resolving disputes between members of the society and to address problems with the governing body. In America, law has always been a key component in ensuring rights and freedoms; as the Constitution was being drafted, it was recognized that without an independent judiciary, "all the reservations of particular rights or privileges would amount to nothing" [160]. The American legal system, for all its flaws, has kept America afloat for almost 250 years.

However, new technologies bring new challenges and shine new light on older challenges in the design, application, and interpretation of the law. Modern cryptography in particular poses many new legal challenges, second only to perhaps the rise of machine learning and automated decision making more broadly. Cryptography, instead of being used only by militaries and for secret correspondence of state leaders, is now deployed by most companies with a digital product, and now touches every aspect of life on the internet or on a device. The new capabilities enabled by cryptography directly lead to new questions of law: How should old rules apply to new technologies? What new actions can we take with these new capabilities? How, if at all, must we restrict the use of these new methods? What principles should guide us toward answers?

Modern data analysis is also inextricably tied to cryptography at this point, via debates over online privacy. Cryptography is one of the most reliable methods for obscuring information about oneself. For the last several years we have been in a perpetual debate over what information about the average person will be revealed, and to whom. It is telling that "statistics," a word originally referring to "the science dealing with data about the condition of a state or community" such as a state would collect [275, p. 1128] is a science now performed routinely not just by governments, but by companies, academics, and anyone who can get their hands on large datasets.

In this chapter, I argue for the benefits of joint technology-law research and describe why I believe it will lead to better and more timely outcomes than performing siloed research in both fields. I first describe the challenges in bringing cryptographic methods to bear on legal issues, and respond to those challenges and describe why I believe joint research is important despite the barriers (§5.1). I then describe specific laws that are heavily affected by aspects of cryptography and statistics and how research in the two areas can inform each other §5.2. I end the section with a discussion of what I see as some promising paths for interdisciplinary research in law and computer science both generally (§5.3) and in specific cryptographic research areas (§5.4).

5.1 In favor of joint research

Feigenbaum and Weitzner [117] describe common problems when seeking to perform socio-technical research. Citing prominent legal scholar Ronald Dworkin [113], they describe the difference in law between *rules* and *principles*: rules "can be understood as logical propositions that are expected to yield answers about what is and is not permitted using formal reasoning capabilities." Principles, on the other hand, are values that guide decisions, but will likely not yield an "unambiguous outcome." Principles are the only guide when rules fail us – indeed, many cases that rise to the level of the U.S. Supreme Court do so because the topic is at an edge case of the rules, and so are decided on principles alone.

A common criticism that technologists face when venturing into the realm of the socio-technical is that it is difficult to logically formalize the deliberately-vague principles and policies that guide policymakers, lawmakers, and judges. The vagueness of these principles is valuable to the long-term stability of society, in concert with the equal application of any hard-and-fast rules. I agree with this criticism – computer scientists who try to transmute principles into rules do so at their peril.

However, even beyond attempts to convert principles into rules, some remaining aversions to interdisciplinary research remain. Technologists are often wary of entering the legal sphere for fear of losing neutrality, although in my opinion this fear is exaggerated: there is plenty of space for descriptive research to measure or predict the effects of a technology. Even when venturing into making normative claims, there is an important need for recommendations backed by legitimate technical evidence.

There is resistance to considering "tech law" or "cyberlaw" its own field on the legal side as well. Most famously, Judge Frank Easterbrook, a Seventh Circuit judge, deemed cyberlaw similar to "Law of the Horse" and claimed that law was at its best when it studied general rules and institutions that could be *applied* to specific cases like cyberspace [114]. His criticism was specifically aimed at lawyers who were likely to make claims about new technologies that were either false outright, or would be outdated in five years. As he said, “put together two fields about which you know little and get the worst of both worlds.”

There have been numerous responses to this critique, most notably from Lawrence Lessig [209] who essentially claims that some general principles can be unearthed from the study of the specific law of cyberspace. Lessig was talking about questions like whether law should adapt in response to specific landscapes, or if it should try to alter the landscape so as to be more regulable under existing law, or to what extent the code itself supplants law in certain scenarios.

Although I respect Easterbrook’s note of caution, I find myself agreeing more with Lessig’s point that sometimes new general points are unearthed by the study of specifics. Although Lessig’s examples of this principle in the early days of the Internet included things like authenticating age (often easy enough in the real world, difficult over the Internet) and the invisibility of tracking, I see similar occurrences in algorithmic fairness. Algorithmic fairness brought to light issues that are not new. In some sense, though the field’s name is “algorithmic” fairness, there is nothing “algorithmic” about the issues studied there at all – human decision makers would be subject to the exact same statistical constraints. It is only because of the scale of algorithms, general discomfort with passing control over a person’s life and well-being over to an algorithm, and a suspicion of the naive promise that algorithms could not reinforce bias, that we noticed these problems existed in the first place. New technologies, sometimes by virtue of their power or scale, illuminate broader issues that now *must* be addressed.

Thus, I reject the separation of cryptographic research from law. For one thing, the separation has become less maintainable as technology has become more and more widespread. Laws must deal with technology, whether specifying legally punishable behavior on the internet or establishing rules about autonomous weapons. So too must technology deal with laws, though this is truer when creating products than when performing academic research. And though technology rarely offers a complete solution to a non-technical problem, it would be ridiculous to suggest that it can never help. Furthermore, even the most die-hard technophobe would surely recommend researching what negative impacts technology could have on the situation at hand, so as to know how best to establish good practices.

5.1.1 Laws, policies, ethical codes, or societal goals?

Before moving on, let us briefly pause to describe six different sets of potential goals for interdisciplinary research: will the research inform laws, policies, standards, best practices, or ethical codes, or merely improve society in a broad way? We expect this style of research to be beneficial to all six. However, the interdisciplinary researcher benefits from understanding the differing effects and strengths of each of these objects, so as to better determine which type is most in need of what information.

By *laws*, I mean stated rules which govern a populace under penalty of punishment if not obeyed, and the processes by which those laws are enforced and challenged. Laws in America, at least those that stand the test of time, are generally narrower, and stronger, than policies and ethical codes. As a result, conducting research that has impact on a law has strong consequences for a narrow set of circumstances. Good examples include the fight over the census and the Appropriations Act mentioned in the previous section, research on the GDPR in Europe, and the analysis on the foregone conclusion doctrine presented in the next chapter of this work. As far as research goes, a primary benefit of examining laws and lawsuits are that they are generally explicitly written, and in America have an explicit goal of self-consistency, with a hierarchic error-correction structure built into the court system to resolve conflicts.

These works have a good starting point in the form of reading the relevant statutes and case law, then attempting to form formal definitions and "test" the definition on existing legal cases. A reasonable ending point would be a recommendation to amend or create a law, an amicus brief on a court case describing the finding, and in some cases, a technical finding. In some cases, the legal finding may lead to additional technical research, such as in §2 which introduces a new security definition and constructs protocols to meet it, where the security definition was a response to the legal finding made earlier.

Policies are formal procedures by which decisions are made. These could be created and used by a government, or by a company or other organization. A policy can be created in order for a company to determine how it is to comply with a law, or set up a default set of allowable actions, or any other formal procedure short of a law. Modifications to policies can be farther reaching, and in many cases just as impactful as modifications to laws themselves. Interdisciplinary research in this area can measure the general landscape of policies, whether by scraping privacy policies or by intensely investigating one particular organization. Statistical research such as that described in §3 should guide policies (or laws) by showing how to achieve various statistical goals, or by showing that some different goals are incompatible with each other.

Somewhat like a cryptographic "ideal functionality," *standards* describe the design, operation, and requirements of protocols or devices. Standards tend to be maintained by a specific organization, especially the National Institute of Standards and Technology, or the Internet Engineering Task Force. Technical research is heavily involved in the creation and improvement of standards, as it should be. My prior work on the Sender Policy Framework for email, for instance, showed a mismatch between the clear intention of the standard, and its actual implementation which allowed a potential denial of service vector by sending an email to a target mail server [260]. We see the primary roles of interdisciplinary research toward standards as being measurement (i.e. measuring the extent to which a standard is adhered to) and finding ways to push the state of the art forward in areas where the standard seems to be in tension with a societal need (for example algorithmic fairness research).

Best practices are weaker than policies in that they are not generally binding. However, they can be legally important: by adhering to an explicit shared list of best practices, an entity can claim that it “did the best it could” and should therefore not be held liable for any problems that occurred while following it. Within cryptography, best practices are relevant in the context of content moderation, security, and end-to-end encryption. Like standards, technical research already makes many rolling changes to best practices lists, especially in security. Interdisciplinary research may wish to go beyond the technical role in places where the best practices are having unintended adverse effects on whatever they are governing, or to see the extent to which various parties are following the best practices, or to tease out gray areas in the best practices where the principles did not align properly with the rules, similar to standards.

Ethical codes are adopted by many professional societies or companies. They are sometimes binding (under the punishment of exile from the organization) and sometimes not. Cryptography research itself has few intersections with ethical codes, aside from ethical codes that set general boundaries of what is considered acceptable research practices. In security research, this primarily consists of guidelines for responsible disclosure, and what systems are considered valid targets for researching attacks. However, in the other direction, ethical codes are beginning to encode privacy, security, and algorithmic fairness requirements. Google’s “Responsible AI practices” contain not only general recommendations but also specific recommendations for fairness, interpretability, privacy, and security [148].

Finally, some research has the explicit goal of pursuing some *societal improvement*. Some researchers, especially those who distrust the motives of powerful governments or corporations laying down laws or policies, prefer to conduct research that they believe will directly improve society, rather than focusing on improving specific codes or laws. Of course, I would hope all research would be carried out with this ethic of responsibility, as Rogaway would put it [254], which has been an important part of science since at least the Russell-Einstein manifesto [255]. Moreover, if one takes a very liberal definition of "improving society," then nearly all research improves society in some way. It is not the case that seemingly-theoretical research will never lead to societal improvement. Although much theoretical research is never directly implemented or used, some theoretical research has tremendous societal impact decades after its original publication, and one cannot often tell which is which for years after its publication. However, in this case, I am referring to research whose results would create some immediate form of societal change. These research problems are a good candidate for interdisciplinary research, and I expect all the same criticisms and advice for successful law-CS research would apply to social-CS research as well.

5.2 Examples of cryptography in the legal regime

In this section, I describe several recent laws which have overlap with cryptography, privacy, or algorithmic fairness. The purpose of this section is not to be an exhaustive list of every law that mentions cryptography, but rather to demonstrate the fact that such laws are now fairly numerous, and show that a joint technical-legal understanding of these laws is beneficial. Urs Gasser has described three general patterns of legal responses to technological change [135]: *subsumption*, in which old rules are applied to new technologies; *innovation*, in which new laws or doctrines are created to deal with the new technology; or *gradual adaptation* of society and law to deal with the new problem without any specific intervention. We will categorize these recent laws accordingly.

The key aspect shared among all these examples is that there is some pressure to come to a timely decision on matters at the intersection of technology and law. Sometimes a lawsuit comes up, and it must be decided where a technology fits into the landscape of the law. Sometimes a new technology is invented that is so impactful that its negative affects must be negated or mitigated, or its positive affects must be adopted and mandated. Oftentimes, the issue is forced before an understanding of the technical and legal issues at hand can be gained naturally by policymakers, and interdisciplinary research becomes an invaluable tool to resolve the quandary in a timely manner.

5.2.1 Examples of subsumption

In this section I describe cases of *subsumption*, where existing laws were adapted to fit new technological challenges.

Federal privacy law The United States' laws on privacy tend to be centered on specific applications. Most notably, the Family Education Rights and Privacy Act of 1974 (FERPA) protects the privacy of educational records [63], the Privacy Act of 1974 describes allowed uses of records about individuals by federal agencies [183], Title 13 provides additional requirements on Census data [293], the Health Insurance Portability and Accountability Act of 1996 creates a privacy rule for health information [14], and the Financial Services Modernization Act of 1999 (GLBA) requires financial institutions to create and adhere to privacy policies [150]. These regulations have inspired a fair amount of research and development, especially in the health care setting (e.g. [8, 202, 249]) though these works tend to apply existing techniques rather than invent new ones. In order to create such techniques, one must have an understanding of both the legal restrictions and the technical security goals.

An 1807 lawsuit: Aaron Burr's secretary. Against the odds, there is actually a very early example of an American legal case involving a cipher. Orin Kerr provides an in-depth modern take on the case in a recent paper [195]. In the facts of the case, Aaron Burr was under trial for treason in 1807. Some letters, written in code, came to light; the court wished to know if Burr was sending encrypted messages to partners in crime. At the time, having knowledge of a treasonous plot (not necessarily your own), and concealing that plot, was itself a crime, known as "misprison of treason." Burr's secretary, Charles Willie, was eventually asked whether he understood the contents of the paper. He refused to answer, citing his 5th Amendment privilege against self-incrimination. His right to do so was more complicated than it sounds. As examined in much greater detail in §2, the modern interpretation of the 5th Amendment only provides a privilege against *self*-incrimination, and only if the requested response is *compelled*, *incriminating*, and *testimonial*. In this case, the request was clearly compelled (the government was forcing Willie to answer) and testimonial (the government sought an oral response to a yes/no question regarding Willie's knowledge); Willie's lawyers argued that Willie's response would incriminate himself for misprison of treason. A lengthy and complex debate ensued among the lawyers and judge of the case. Eventually, it was determined that since only his *current* knowledge of the cipher was being asked, the government would have no way of knowing whether Willie would have been capable of reading the letter earlier, and therefore there was no risk of Willie incriminating himself for misprison of treason. This is a reassuring case, as we study the 5th Amendment in a different encryption scenario, as we have a test that shows that both the general 5th Amendment principles, and specific court doctrines, can be subsumed to apply to encryption technologies as well. We explore technical and legal issues of compelled decryption in much greater detail in §2; these issues remain challenging in both fields to this day.

Privacy in the census. For the 2020 census, the Census Bureau adopted differential privacy in order to prevent attacks that would allow reconstruction of many individuals' census responses as a result of the numerous statistics published by the Census. This poses a question of how to determine how new technological methods should be treated under existing laws. The Census is a particularly interesting example of a function that has strict requirements that tug in different directions: on the one hand, it must maintain the confidentiality of individual responses for 72 years; on the other hand they provide the most accurate count of people in the country, which is most notably used to draw districts and apportion seats in the House of Representatives. These goals in tension with each other require a high level of technical sophistication for the Census Bureau. Moreover, previous attempts it has made at changing their method have been explicitly prohibited by law: section 195 of the Census Act prohibited the "statistical sampling" method the Census sought to use in 2020 [240]. Now, in 2021, after the adoption of differential privacy, a key lawsuit [285] was brought by the state of Alabama (representing several states) challenging the use of differential privacy. Among other claims, it calls differential privacy a "statistical method" that can be challenged; in this context, the 1998 Appropriations Act defines statistical method as a "statistical procedure... to add or subtract counts to or from the enumeration of the population as a result of statistical inference." This then passes the buck down to asking what exactly the definition of statistical inference is. This question of law has clearly become a question of mathematics. This case, along with many others, demonstrates the need for timely specialized research by computer scientists that is relevant to questions of law and policy.

5.2.2 Examples of innovation

This section describes new laws which were *innovated* to deal with new technologies. Determining the boundary of this section with the gradual adaptation category is challenging – many new laws propose a small change, especially to limit the use of a new technology, however these changes are usually narrow changes rather than sweeping new innovations. For now, our criterion for being an "innovation" rather than a "gradual adaptation" is a relative measure rather than an absolute one: I have called more drastic changes "innovations" and small adjustments "gradual adaptations."

In Europe: the GDPR The most obvious recent example of a cryptography-adjacent law is the European General Data Protection Regulation (GDPR). Not only did GDPR bring in sweeping new privacy laws that companies all over the world scrambled to adhere to, it also brought specific (non-technical) definitions of several types of deanonymization, including singling out, linking, and inferring. In addition, a working party created an influential list of guidelines describing whether several anonymization techniques such as pseudonymisation, noise addition, substitution, k-anonymity, l-diversity, hashing, and differential privacy, provided adequate defense against each of the three types of attacks. Each of these techniques requires a mathematical analysis to determine whether or not it meets the new security definitions. These analyses were later further refined in a technical paper by Cohen and Nissim [84]. The GDPR also stipulates that "the controller shall have the obligation to erase personal data without undue delay" leading to technical works meant to determine how, exactly, to erase data that has already been incorporated into some kind of framework like a machine learning model [134]. Finally, the GDPR has also encouraged people to apply known techniques such as multi-party computation and homomorphic encryption to new problems, especially in health care, e.g. [128, 262].

Biometric Information Privacy Act (BIPA). BIPA is an Illinois law enacted in 2008. It is most famous for requiring companies to collect consent *each time* a biometric (fingerprint, iris/retina scan, voiceprint, facial/hand scan) is disclosed. Additionally, it also grants individuals the right to sue in the State of Illinois if this is violated. This law rose to prominence again in the late 2010s as Facebook began performing facial recognition and "Face ID" was developed as an option for logging into smartphones. Facebook settled a lawsuit for $650 million for violating BIPA; there is currently a similar ongoing case against TikTok (in TikTok's case, the lawsuit alleges that the app conducts a full facial scan so as to be able to display face filters in real time video; TikTok's lawyers say that the app does not capture any biometric information) [251]. This law inspires interdisciplinary questions about the definition of "biometric" and the purposes for which they should and should not be used.

Facial recognition bans. Several city and state governments have banned law-enforcement use of facial recognition. As summarized by the ACLU in 2021, these include "San Francisco, Berkeley, and Oakland, California; Boston, Brookline, Cambridge, Easthampton, Northampton, Springfield, and Somerville, Massachusetts; New Orleans, Louisiana; Jackson, Mississippi; Portland, Maine; Minneapolis, Minnesota; [and] Portland, Oregon" [9]. Similar bans exist at the state level for Virginia and Vermont. California prohibited facial recognition specifically in police-worn body cameras, and New York state prohibited facial recognition in schools [9]. Two key issues arise with facial recognition: First, current facial recognition tools developed in the US are famously less accurate on darker-skinned faces [66]; inaccuracy in facial recognition has now led to at least one mistaken arrest (of a black man) [129]. Second, even if facial recognition worked perfectly accurately on everyone regardless of appearance, the thought of applying facial recognition and tracking to omnipresent cameras inspires an Orwellian image of mass surveillance. Several factors are currently pushing the federal government to enact some kind of federal regulation for facial recognition software; many are pushing for a full moratorium of government use of facial recognition. This is likely to be an area with continuing need for research to address a myriad of technical and legal questions for years to come.

5.2.3 Examples of gradual adaptation

This section describes proposed legislation which represents a *gradual adaptation* to new technology – although they propose new changes and could therefore be considered innovations, the innovations are relatively small.

Student Know Before You Go Act. The "Student Know Before You Go Act," was originally proposed in 2012 by Senators Wyden and Rubio, and has been reintroduced several times with new modifications, most recently in 2019 by Senators Wyden, Rubio, and Warner. It is meant to allow potential college students to learn information about various colleges' graduates' debt, future job situations, and so on. The bill specifically requires that the students' data be protected by "secure multiparty computation technologies; or (2) may utilize technology other than secure multiparty computation technologies if the other technology – (A) fully complies with [security requirements given earlier in the bill]; and (B) delivers greater student privacy and security than secure multiparty computation" [323, 3(c)]. Of particular note are the security requirements described: the system must "use[] technical protection measures that reasonably ensure that – (A) a reporting entity's raw data, including personally identifiable information, shall not be accessible through the system to the Department or any party other than the reporting entity; (B) no information about the data components used in the system is revealed by the system to the Department or any other party, except as incorporated into the outcome metrics described in Section 5; and (C) no data or information that can identify an individual is revealed by the system to the Department or any other party" [323, 3(b)]. This proposed law thus addresses a use case for cryptography that still remains underutilized in industry: the ability for multiple data-holders to provide an obfuscated version of their data to some central computing server, and receive some analysis of their data, without revealing the data itself. The ability to state this goal in these terms is the fruit of effective communication between cryptographers and policymakers.

Algorithmic Accountability Act. In a different technical regime, the "Algorithmic Accountability Act of 2019" proposed by Senators Wyden and Booker would have required data-holding companies and organizations to conduct regular assessments of automated decision-making systems, with extra requirements for "high-risk" systems that pose significant risk of privacy or security violation, or discrimination [322]. The list of what must be present in the assessment includes ideas that might already be considered best practices for data-holders: performing data minimization, setting a time period after which the data and results might be destroyed, determining how much of the system is visible for public review, describing how to appeal results, and who receives the results. This is a reasonable list of things to assess, but the bill still begs the question of how to determine what "significant risk" means in the classification of determining a "high-risk" system. In practice, this would likely either turn to an existing technical standard, or create a new standard. Once again, setting this standard requires an understanding of both the legal desiderata and the technical capabilities of the systems analyzed.

EARN IT Act. The "Eliminating Abusive and Rampant Neglect of Interactive Technologies Act of 2020" (or "EARN IT Act"), introduced by Senator Lindsey Graham and 16 cosponsors including Senator Dianne Feinstein, has a long and complex history. In summary, it was meant to encourage online platforms to adopt practices that would make it more difficult to send child sexual abuse material (CSAM) without detection. Primarily it does this by amending Section 230 of the Communications Decency Act. Currently, Section 230 broadly grants online providers immunity against civil and criminal lawsuits: they are not “treated as the publisher or speaker” of information [125]. This creates a “safe harbor” for websites and other providers to be shielded from liability for what their users post. There has been a lot of discussion recently about amending Section 230, and EARN IT is a part of that discussion. Essentially, the EARN IT act would specify “best practices” that would allow the platforms to keep that liability shield. (The companies could do something other than the best practices, but they would then be open to litigation and would have to justify their choices to the court.) The best practices would include retaining CSAM and associated location data [149, 4(3)(C)] and using content moderators to review potential CSAM [149, 4(3)(F)]. While EARN IT was originally understood to be an “anti-encryption bill,” in response to criticism [246] the most recent draft explicitly carves out an exception for the use of end-to-end encryption, stating [149, 5(7)]:

> (7) CYBERSECURITY PROTECTIONS DO NOT GIVE RISE TO LIABILITY.–Notwithstanding paragraph (6), a provider of an interactive computer service shall not be deemed to be in violation of section 2252 or 2252A of title 18, United States Code, for the purposes of subparagraph (A) of such paragraph (6), and shall not otherwise be subject to any charge in a criminal prosecution under State law under subparagraph (B) of such paragraph (6), or any claim in a civil action under State law under subparagraph (C) of such paragraph (6), because the provider– (A) utilizes full end-to-end encrypted messaging services, device encryption, or other encryption services;

> (B) does not possess the information necessary to decrypt a communication; or
>
> (C) fails to take an action that would otherwise undermine the ability of the provider to offer full end-to-end encrypted messaging services, device encryption, or other encryption services.

Clearly, this is a law that must interact meaningfully with several different cryptographic and privacy notions in order to determine what best practices to recommend, what technologies to carve out as an exception, and other similar concerns.

5.2.4 Interdisciplinary research accelerates the process of improving the law

In all of these specific laws and lawsuits, there is a certain need for interoperability between technical and legal concepts. There is a clear public interest in ensuring that proper technological safeguards are adopted and mandated in situations that warrant (for example) privacy or accountability tools, and in banning technologies that pose an unacceptable risk. To some extent this interoperability happens naturally: as a technology becomes more widespread, the concepts become more approachable even to non-experts.

However, waiting for that natural understanding is often not an option. Few lawmakers are experts in facial recognition technology or encryption, nor should they need to be. Without research showing disproportionate inaccuracy of facial recognition on specific demographics [66], the benefits of facial recognition in law enforcement use would likely have been recognized as more "legible" than the costs – which have been determined to be so severe that several municipalities are banning the technology outright. The original draft of the EARN IT Act would have heavily discouraged end-to-end encryption (see §5.2.3), whereas the new draft specifically carves out an exception for end-to-end encryption, in part due to widespread criticism from researchers in law, cryptography, and the intersection [246]. Arguments over the algorithms used in the Census require an understanding of both the legal requirements and the technical algorithm being deployed in order to determine the right path forward; since an active lawsuit is occurring over the 2020 Census process, we cannot wait for a natural understanding of the methods involved to percolate through society.

In all of these areas described above, interdisciplinary researchers are in the ideal position to make timely recommendations for these systems.

5.3 A path forward

The stock criticism of interdisciplinary tech-law work is correct: the weakest link is usually interface between the model and reality. Cryptographers will be familiar with this concept already – a theoretical framework can have air-tight security yet still be thwarted by side channels or incorrect assumptions. The same is true of the interface between a legal concept and a technical modeling thereof. However, this is not to say that the technical models have no use whatsoever. As the saying goes, all models are broken, some models are useful. And, as another saying goes, if you're doing anything, you're probably using a model, so we have a desire for high-quality models. This is exactly the problem that this area of interdisciplinary tech-law research seeks to address: making a stronger interface between the model and reality.

At the end of their paper describing common problems when seeking to perform socio-technical research [117], Feigenbaum and Weitzner describe two areas where they believe socio-technical research can be especially fruitful: First, clarifying in what situations rules and principles are the main guide, often by amending the rules so that they are either more in line with the principles, or have fewer gray areas in which they must be abandoned for principles. They provide location privacy as an example: the rules are few and are either old laws that were subsumed into applying to new technology, or very strict constitutional limits; the called-upon principles in the location privacy debate are conflicting as well, and are at times in conflict with the rules. Second, socio-technical research also helps bring transparency and accountability to decision-making processes that are governed by principles, in the areas where rules do not reach. The authors describe surveillance policy as an example; the authors recommend both research into the state of affairs (rather than relying on conflicting reports from various agencies and advocacy organizations), as well as ensuring that any technical intervention remain accountable.

We agree that these two areas of socio-technical research are excellent opportunities where technology can help inform new and existing rules and principles. However, I would like to bring this further and describe a handful of other opportunities I see in the field where cryptography can decrypt legal dilemmas. As described in the last section, it is usually principles, and not cryptography, that will help guide you to a choice between two dilemma options. However, cryptography can often help to illustrate what the dilemma options are. It can inform you when two different goals are compatible, and when they are not. It can illustrate gaps between the principles and the rules. It can improve the benefits, or reduce the costs, of one or both options in the dilemma. Sometimes, it can innovate a third option, or a different resolution. And finally, interdisciplinary tech-law research can guide us to new principles and rules. We conclude this section with a description of these research types in a little more detail.

Illustrating the dilemma options and revealing gray areas. Probably the most common form of interdisciplinary tech-law research has to do with clarifying the state of what I have dubbed "legal dilemmas." This research either analyzes specific technologies, determining whether they meet various legal criteria, or they formally model laws so as to be able to apply them in new situations in the absence of case law. As an example, Nissim et al. [239] formally model some privacy laws for education in the U.S. Garg et al. [134] provide a simulation-based definition of the right to be forgotten. In addition to modeling law, determining whether certain technologies adhere to legal criteria also falls under this category. For example, in addition to formalizing one aspect of European privacy law into a security goal, Cohen and Nissim [84] prove that differential privacy achieves this security goal but k-anonymity does not. There are several works analyzing the use of multi-party computation in various privacy law contexts, for example in Estonia [53] and under GDPR generally [262]. We find this kind of work helpful; analogies of various cryptographic technologies, like all analogies, inevitably fail at the edges. I see the main portion of §2 as an example of this category. More broadly, most interdisciplinary has some results in this category even if the main thrust of the work sought a different goal. All the statistical research in algorithmic fairness, for instance, helps illustrate the fact that the problem is not merely "algorithmic" as mentioned earlier: replacing algorithms with humans will not rely underlying statistical invariants. This category also includes technical researchers communicating clearer information about new technologies to legal and social scholars and resolve discrepancies where they arise. This is especially important with technologies that are growing quickly and have far-reaching consequences. This touches everything from internet governance in the earlier days, to blockchain and autonomous weapon systems today.

Compatibility of different goals. Some technical research also yields impossibility results. It is clear why this is helpful from a policy standpoint: if a policy wants to achieve two different goals, it helps to know that they are both statistically achievable. Our work in §3 addressed some questions of this ilk. Even if not a strict impossibility, technical research can also reveal situations where different goals are at odds with each other. The morass of work surrounding the exceptional access debate is a good example here, for example [1].

Improving opportunity cost of one choice or developing a third option. Cryptography in particular has great potential for improving the benefits (or reducing the costs) of a dilemma option itself. It can add accountability or enforceability to existing rules. It may improve the privacy of individuals by replacing privacy-invasive implementations of rule-enforcement with ones that limit the information collected. Frankle et al [126] develop a method for ensuring that data requests by government agencies adhere to formal policies, based on work by Goldwasser and Park [147]. Some various attempts at creating exceptional access fall into this category, attempting to preserve some level of privacy or defense against mass surveillance while allowing a small number of encrypted devices to be decrypted by law enforcement (e.g. [62,77,107,204,258,278]), though unsurprisingly these do not resolve the tension in the principles themselves. We see a similar set of research beginning in the context of moderation in end-to-end encrypted chat [205].

Zero-knowledge proofs in particular are valuable tools that open new doors for accountability and privacy. They are natural choices for enforcing properties that would have previously been checked non-cryptographically, improving accountability and enforcement. They will also potentially improve privacy by using zero-knowledge proofs where previously any non-privacy-preserving proof would have done. Our work in §4 provides two zero-knowledge proofs that can be used in this manner. However, it must be said that zero-knowledge proofs can be a double-edged sword. In addition to adding privacy to existing surveillance mechanisms, they may also enable surveillance opportunities that did not exist before. Information that was previously unqueriable because it was tied to sensitive information might not reveal the sensitive information anymore. The recent Supreme Court case *Carpenter v. U.S.* [74] is illustrative: the warrantless collection of 127 days' worth of Carpenter's cell phone location information (12,898 individual location points) was deemed to be a Fourth Amendment search. However, envision a world in which law enforcement had been able to conduct a search query that revealed only one bit of information: whether or not Carpenter's cell phone was at each of the nine robbery locations (and was the only cell phone that met this criterion). In this alternate world, it is not at all clear whether or not this would still be regarded as a search subject to Fourth Amendment protections.

Toward new principles and rules. Last, interdisciplinary research can help create new principles and rules for emerging technologies. The very idea that "code" can guide people's behavior along with laws, norms, and markets [209] is a new principle, or at least something more abstract than a rule. The idea of "statistical fairness" as an overall goal takes on new importance and meaning with widespread homogeneous decision-making systems deployed on tens or hundreds of millions of people, even if those ideas had already been applied decades previously in the context of racial bias in standardized tests [95]. Interdisciplinary research rarely yields these types of new broad principles, and technical research even less so, but when they do, the results take on great significance.

5.4 Key areas for future work in joint cryptography-law research

I wish to conclude this thesis by describing several questions that I believe will be of great importance in the near to medium-term future. In particular, I identify five key areas of interdisciplinary research that I predict will require significant attention and research over the next several years and decades.

Election verifiability, integrity, and auditability Election security is a prominent public issue due to both real and perceived concerns about the security of all components of the vote counting and tallying process, which became an especially hot issue in 2020 [45,46,99,233,241]. Two issues are outstanding that cryptographers can help address and improve the state of affairs in 2022 and beyond: First, current electronic voting systems remain vulnerable to known attacks [48–51]. Most security researchers flatly recommend against adopting electronic voting due to these problems, including issues with specific proposed systems and difficulties verifying their integrity (e.g. [12,48,233,242,242,272]). However, some of the proposed security and verifiability notions for electronic voting can still apply to more easily verified paper ballots. At the absolute least, we should widely implement the current best advice of election security researchers, including voter-marked paper ballots and risk-limiting audits [47], and move to end-to-end verifiable elections [233]. This *end-to-end verifiability* guarantee brings cryptography-style formal guarantees of integrity, counting accuracy, public verifiability, and transparency [132,233]. The second outstanding issue is one of public trust. Regardless of what actually happened in the 2020 election, it is clear that existing systems lack the ability to "convince" someone that the election results were legitimate in the sense that a cryptographic proof "convinces" the verifier that a statement is true. Cryptographically-verifiable voting would not solve the entire problem – as always, the weakest link will be in connecting the real world to whatever cryptographic object is used to verify – but any steps that can be made in the direction of more public verifiability in the voting process and outcome will be helpful. One silver lining in the wake of the 2020 election is that companies that build election infrastructure are now interested in ways they can verify their results to the public (e.g. [216]). Cryptographers should take advantage of this renewed interest to improve the security and verifiability of these systems more generally, beyond merely addressing the current public disputes.

Weapons. Militaries have had no shortage of interest in cryptography since its invention, and this is not likely to change. What will change, however, are the technologies and methods used to wage war, and these changes will impact the types of cryptography that is of military interest. If autonomous weapons are not banned by treaty, they will almost certainly be used extensively, and possibly this will be the case even if they are banned by treaty. While autonomous (or mere unmanned) weapons started out quite expensive, with improving drone technology they are now becoming quite cheap: numerous small cheap machines are becoming the norm rather than large expensive ones [161, 230, 310]. My own research into autonomous weapon systems indicates several issues with the modern legal understanding of these technologies [259]. My past research focused on the artificial intelligence aspect of the weapons, however this technology also raises a host of key management problems and other cryptographic questions. In the military context, the question of a device being captured and its keys being stolen is not an *if*, but a *when, and how many.* Communications networks among many devices are likely to be varied and intermittent, unlike the standard fully-connected graph assumed by most cryptographic multi-party protocols: there will be a greater emphasis on unusual communication topologies and topology-hiding multi-party computation [4, 18, 167, 208]. Lightweight hardware will require lightweight cryptography while maintaining the security against the compromise of individual devices – this is also true in the burgeoning civilian drone market [211, 279, 280]. These problems are reminiscent of problems posed by the Internet of Things (IoT) paradigm, but unlike IoT, the military drone setting has more incentive to address security issues. Military research is likely to advance the field of cryptography in these areas; cryptographers ignore this at their peril.

Responsible and verifiable corporate and government data use. In 2021, there is no shortage of guidelines on how to use data responsibly. We discussed various data regulations in §5.2, but even aside from official regulations, many major companies have their own internal guidelines for ensuring that data use is private, secure, and fair. These solutions remain tenable, if somewhat unsatisfying. However, the data held by these entities over the years will only grow in both quantity and type. Envision holding millions of 50-year longitudinal browsing histories, location data, biometrics, genome data, email, chat history, health information, social media, augmented reality, shopping habits, food ordered, books read and videos watched, questions searched for, and people talked to. There is already public unease at the breadth of this data held by both companies and government agencies, but this data will only grow unless policy is drastically changed. Many of the important upcoming changes in responsible data use will be in machine learning and algorithmic fairness. However, cryptography also has an extremely important place at the table. This is true not only for data-minimization and privacy purposes, but also for verifiability and accountability purposes: With such a wealth of information at their disposal, it seems that the least we can do is verify that they're actually using the data in the way they say they are. Zero knowledge proofs such as those described in §4 are among the best tools to implement these methods. Moreover, since no company or government agency is incentivized to so of its own accord, law must also play a part in requiring such verifiable actions.

Anonymity, identity, and accountability. Long-simmering arguments about anonymity are boiling hotter now. At the application layer especially, the American is debating the merits of anonymity online. This argument is intertwined with arguments about content moderation also touched upon in §3. To summarize the situation, two schools of thought stand in tension: On the one hand, by requiring identity to be revealed, there are benefits to social cohesion and a discouragement to be antisocial; moreover it may be a legal requirement if the service involves some kind of commerce or other transaction [140, 192, 226, 276]. On the other hand, anonymity brings positive aspects such as the ability to ask advice about private affairs without reputational impact, a wider ability to experiment with new ideas, the autonomy to control your own information, and being freed from the "burden of social markers" of various identity aspects such as race and gender [39, 58, 140, 290]. There are also practical problems associated with enforcing people's real names, such as issues with stage names or lack of ID [158] or people who shield their identity due to domestic abuse or stalking [140]. Both these schools of thought have some merit to them; we are faced with a tradeoff between the benefits of anonymity and the accountability that comes with identity. I have stated all these problems at the application layer, but they can be extended down to the network layer – many use Tor for the benefits of traffic anonymization [250, 287], but others point to the extent to which that anonymity enables criminal activity (e.g. [180, 315]).

We wish to improve the tradeoff between these two poles. We will need both social and technical innovations to retain the benefits of anonymity while reducing the costs [179]. A good step in this direction is Cloudflare's Privacy Pass [103], which improves the browsing experience of human Tor users by limiting the number of CAPTCHA challenges they face, while still forcing bot users of Tor to go through these CAPTCHAs. We should investigate other improvements, both technical and social, that improve the Pareto frontier between the benefits of anonymity and the accountability of identity.

Deniability. As discussed in §2, the crypto wars show no sign of letting up anytime soon. Especially in the Fifth Amendment context, and also to a lesser extent in the Fourth Amendment context, *deniability* is likely to become a key aspect of many upcoming cryptosystems. A *deniable* encryption scheme [71] allows a party to maintain some security even if coerced to reveal some state after the communication is complete. Depending on what direction the Supreme Court goes, it may become the case that the only way to ensure information is hidden in an uncorruptible way is to use deniable encryption. Indeed, this has already come up in a Circuit Court case [175]. The technical state of deniability remains primarily focused on the problem of encryption, and there is good theoretical [72] and practical [309] background for deniable encryption. Deniable authentication is already somewhat addressed in the symmetric setting in the form of Message Authentication Codes. However, as cryptography progresses, so too will attempts at bypassing it, and I foresee a world in which the entire stack must be deniable. Also, on the legal side, a gap remains between the cryptographic notion of deniability and the ability to challenge evidence in a courtroom. A better understanding of deniability from both the legal and technical side will be key in the coming years.

Bibliography

[1] Harold Abelson, Ross Anderson, Steven M Bellovin, Josh Benaloh, Matt Blaze, Whitfield Diffie, John Gilmore, Matthew Green, Susan Landau, Peter G Neumann, et al. Keys under doormats: mandating insecurity by requiring government access to all data and communications. *Journal of Cybersecurity*, 1(1):69–79, 2015.

[2] André Adelsbach, Stefan Katzenbeisser, and Ahmad-Reza Sadeghi. Watermark detection with zero-knowledge disclosure. *Multimedia Systems*, 9(3):266–278, 2003.

[3] André Adelsbach and Ahmad-Reza Sadeghi. Zero-knowledge watermark detection and proof of ownership. In *International Workshop on Information Hiding*, pages 273–288. Springer, 2001.

[4] Adi Akavia, Rio LaVigne, and Tal Moran. Topology-hiding computation on all graphs. *Journal of Cryptology*, 33(1):176–227, January 2020.

[5] Gorjan Alagic, Jacob Alperin-Sheriff, Daniel Apon, David Cooper, Quynh Dang, John Kelsey, Yi-Kai Liu, Carl Miller, Dustin Moody, Rene Peralta, Ray Perlner, Angela Robinson, and Daniel Smith-Tone. Status report on the second round of the NIST post-quantum cryptography standardization process. https://csrc.nist.gov/publications/detail/nistir/8309/final, 2020.

[6] Martin R. Albrecht, Christian Rechberger, Thomas Schneider, Tyge Tiessen, and Michael Zohner. Ciphers for MPC and FHE. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology – EUROCRYPT 2015, Part I*, volume 9056 of *Lecture Notes in Computer Science*, pages 430–454. Springer, Heidelberg, April 2015.

[7] Alcorta v. Texas, 355 US 28 - Supreme Court 1957.

[8] Tariq Alshugran and Julius Dichter. Toward a privacy preserving hipaa-compliant access control model for web services. In *IEEE International Conference on Electro/Information Technology*, pages 163–167. IEEE, 2014.

[9] American Civil Liberties Union. ACLU Statement on Extended Amazon Face Recognition Moratorium, 2021. https://www.aclu.org/press-releases/aclu-statement-extended-amazon-face-recognition-moratorium.

[10] Scott Ames, Carmit Hazay, Yuval Ishai, and Muthuramakrishnan Venkitasubramaniam. Ligero: Lightweight sublinear arguments without a trusted setup. In Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu, editors, *ACM CCS 2017: 24th Conference on Computer and Communications Security*, pages 2087–2104. ACM Press, October / November 2017.

[11] Julia Angwin, Jeff Larson, Surya Mattu, and Lauren Kirchner. Machine bias: There's software used across the country to predict future criminals. And it's biased against blacks. *ProPublica*, May 23, 2016.

[12] Andrew Appel, Richard DeMillo, and Philip Stark. Ballot-marking devices (BMDs) cannot assure the will of the voters. *Available at SSRN 3375755*, April 2019.

[13] Toshinori Araki, Jun Furukawa, Yehuda Lindell, Ariel Nof, and Kazuma Ohara. High-throughput semi-honest secure three-party computation with an honest majority. In Edgar R. Weippl, Stefan Katzenbeisser, Christopher Kruegel, Andrew C. Myers, and Shai Halevi, editors, *ACM CCS 2016: 23rd Conference on Computer and Communications Security*, pages 805–817. ACM Press, October 2016.

[14] Rep. Bill Archer. Health Insurance Portability and Accountability Act, 1996.

[15] Giuseppe Ateniese, Bernardo Magri, and Daniele Venturi. Subversion-resilient signature schemes. In Indrajit Ray, Ninghui Li, and Christopher Kruegel, editors, *ACM CCS 2015: 22nd Conference on Computer and Communications Security*, pages 364–375. ACM Press, October 2015.

[16] Nicola Atzei, Massimo Bartoletti, Tiziana Cimoli, Stefano Lande, and Roberto Zunino. SoK: unraveling bitcoin smart contracts. Cryptology ePrint Archive, Report 2018/192, 2018. https://eprint.iacr.org/2018/192.

[17] Michael Backes, Dario Fiore, and Raphael M. Reischuk. Verifiable delegation of computation on outsourced data. In Ahmad-Reza Sadeghi, Virgil D. Gligor, and Moti Yung, editors, *ACM CCS 2013: 20th Conference on Computer and Communications Security*, pages 863–874. ACM Press, November 2013.

[18] Marshall Ball, Elette Boyle, Tal Malkin, and Tal Moran. Exploring the boundaries of topology-hiding computation. In Jesper Buus Nielsen and Vincent Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2018, Part III*, volume 10822 of *Lecture Notes in Computer Science*, pages 294–325. Springer, Heidelberg, April / May 2018.

[19] Kenneth A Bamberger, Ran Canetti, Shafi Goldwasser, Rebecca Wexler, and Evan Joseph Zimmerman. Verification dilemmas, law, and the promise of zero-knowledge proofs. *Law, and the Promise of Zero-Knowledge Proofs*, February 2021.
https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3781082.

[20] Balthazar Bauer, Pooya Farshim, and Sogol Mazaheri. Combiners for backdoored random oracles. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology – CRYPTO 2018, Part II*, volume 10992 of *Lecture Notes in Computer Science*, pages 272–302. Springer, Heidelberg, August 2018.

[21] Carsten Baum, Jonathan Bootle, Andrea Cerulli, Rafaël del Pino, Jens Groth, and Vadim Lyubashevsky. Sub-linear lattice-based zero-knowledge arguments for arithmetic circuits. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology – CRYPTO 2018, Part II*, volume 10992 of *Lecture Notes in Computer Science*, pages 669–699. Springer, Heidelberg, August 2018.

[22] Carsten Baum, Daniele Cozzo, and Nigel P. Smart. Using TopGear in overdrive: A more efficient ZKPoK for SPDZ. In Kenneth G. Paterson and Douglas Stebila, editors, *SAC 2019: 26th Annual International Workshop on Selected Areas in Cryptography*, volume 11959 of *Lecture Notes in Computer Science*, pages 274–302. Springer, Heidelberg, August 2019.

[23] Carsten Baum, Cyprien Delpech de Saint Guilhem, Daniel Kales, Emmanuela Orsini, Peter Scholl, and Greg Zaverucha. Banquet: Short and fast signatures from AES. In *Public Key Cryptography*, Lecture Notes in Computer Science. Springer, 2021.

[24] Carsten Baum, Alex J. Malozemoff, Marc Rosen, and Peter Scholl. Mac'n'cheese: Zero-knowledge proofs for arithmetic circuits with nested disjunctions. Cryptology ePrint Archive, Report 2020/1410, 2020.
https://eprint.iacr.org/2020/1410.

[25] Carsten Baum and Ariel Nof. Concretely-efficient zero-knowledge arguments for arithmetic circuits and their application to lattice-based cryptography. In Aggelos Kiayias, Markulf Kohlweiss, Petros Wallden, and Vassilis Zikas, editors, *PKC 2020: 23rd International Conference on Theory and Practice of Public Key Cryptography, Part I*, volume 12110 of *Lecture Notes in Computer Science*, pages 495–526. Springer, Heidelberg, May 2020.

[26] Donald Beaver, Silvio Micali, and Phillip Rogaway. The round complexity of secure protocols (extended abstract). In *22nd Annual ACM Symposium on Theory of Computing*, pages 503–513. ACM Press, May 1990.

[27] Mira Belenkiy, Jan Camenisch, Melissa Chase, Markulf Kohlweiss, Anna Lysyanskaya, and Hovav Shacham. Randomizable proofs and delegatable anonymous credentials. In Shai Halevi, editor, *Advances in Cryptology – CRYPTO 2009*, volume 5677 of *Lecture Notes in Computer Science*, pages 108–125. Springer, Heidelberg, August 2009.

[28] Mihir Bellare and Shafi Goldwasser. New paradigms for digital signatures and message authentication based on non-interactive zero knowledge proofs. In Gilles Brassard, editor, *Advances in Cryptology – CRYPTO'89*, volume 435 of *Lecture Notes in Computer Science*, pages 194–211. Springer, Heidelberg, August 1990.

[29] Steven M. Bellovin, Matt Blaze, Sandy Clark, and Susan Landau. Going bright: Wiretapping without weakening communications infrastructure. *IEEE Security & Privacy*, 11(1):62–72, 2013.

[30] Steven M. Bellovin, Matt Blaze, Sandy Clark, and Susan Landau. Lawful hacking: Using existing vulnerabilities for wiretapping on the internet. In *Northwestern Journal of Technology & Intellectual Property*, 2013. https://scholarlycommons.law.northwestern.edu/njtip/vol12/iss1/1/.

[31] Aner Ben-Efraim, Michael Nielsen, and Eran Omri. Turbospeedz: Double your online SPDZ! Improving SPDZ using function dependent preprocessing. In Robert H. Deng, Valérie Gauthier-Umaña, Martín Ochoa, and Moti Yung, editors, *ACNS 19: 17th International Conference on Applied Cryptography and Network Security*, volume 11464 of *Lecture Notes in Computer Science*, pages 530–549. Springer, Heidelberg, June 2019.

[32] Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, and Michael Riabzev. Scalable, transparent, and post-quantum secure computational integrity. Cryptology ePrint Archive, Report 2018/046, 2018. https://eprint.iacr.org/2018/046.

[33] Eli Ben-Sasson, Alessandro Chiesa, Christina Garman, Matthew Green, Ian Miers, Eran Tromer, and Madars Virza. Zerocash: Decentralized anonymous payments from bitcoin. In *2014 IEEE Symposium on Security and Privacy*, pages 459–474. IEEE Computer Society Press, May 2014.

[34] Eli Ben-Sasson, Alessandro Chiesa, Daniel Genkin, Eran Tromer, and Madars Virza. SNARKs for C: Verifying program executions succinctly and in zero knowledge. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology – CRYPTO 2013, Part II*, volume 8043 of *Lecture Notes in Computer Science*, pages 90–108. Springer, Heidelberg, August 2013.

[35] Eli Ben-Sasson, Alessandro Chiesa, Michael Riabzev, Nicholas Spooner, Madars Virza, and Nicholas P. Ward. Aurora: Transparent succinct

arguments for R1CS. In Yuval Ishai and Vincent Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2019, Part I*, volume 11476 of *Lecture Notes in Computer Science*, pages 103–128. Springer, Heidelberg, May 2019.

[36] Eli Ben-Sasson, Alessandro Chiesa, and Nicholas Spooner. Interactive oracle proofs. In Martin Hirt and Adam D. Smith, editors, *TCC 2016-B: 14th Theory of Cryptography Conference, Part II*, volume 9986 of *Lecture Notes in Computer Science*, pages 31–60. Springer, Heidelberg, October / November 2016.

[37] Eli Ben-Sasson, Alessandro Chiesa, Eran Tromer, and Madars Virza. Succinct non-interactive zero knowledge for a von Neumann architecture. In Kevin Fu and Jaeyeon Jung, editors, *USENIX Security 2014: 23rd USENIX Security Symposium*, pages 781–796. USENIX Association, August 2014.

[38] Rikke Bendlin, Ivan Damgård, Claudio Orlandi, and Sarah Zakarias. Semi-homomorphic encryption and multiparty computation. In Kenneth G. Paterson, editor, *Advances in Cryptology – EUROCRYPT 2011*, volume 6632 of *Lecture Notes in Computer Science*, pages 169–188. Springer, Heidelberg, May 2011.

[39] Michael Bernstein, Andrés Monroy-Hernández, Drew Harry, Paul André, Katrina Panovich, and Greg Vargas. 4chan and /b/: An analysis of anonymity and ephemerality in a large online community. *Proceedings of the International AAAI Conference on Web and Social Media*, 5(1):50–57, Jul. 2011.

[40] Bernstein v. US Dept. of State, 922 F. Supp. 1426 - Northern District of California 1996.

[41] Ward Beullens and Cyprien de Saint Guilhem. LegRoast: Efficient post-quantum signatures from the Legendre PRF. In Jintai Ding and Jean-Pierre Tillich, editors, *Post-Quantum Cryptography - 11th International Conference, PQCrypto 2020*, pages 130–150. Springer, Heidelberg, 2020.

[42] Rishabh Bhadauria, Zhiyong Fang, Carmit Hazay, Muthuramakrishnan Venkitasubramaniam, Tiancheng Xie, and Yupeng Zhang. Ligero++: A new optimized sublinear IOP. In Jay Ligatti, Xinming Ou, Jonathan Katz, and Giovanni Vigna, editors, *ACM CCS 20: 27th Conference on Computer and Communications Security*, pages 2025–2038. ACM Press, November 2020.

[43] Alex Biryukov, Daniel Dinu, and Dmitry Khovratovich. Argon2: new generation of memory-hard functions for password hashing and other applications. In *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 292–302. IEEE, 2016.

[44] Nir Bitansky, Alessandro Chiesa, Yuval Ishai, Rafail Ostrovsky, and Omer Paneth. Succinct non-interactive arguments via linear interactive proofs. In Amit Sahai, editor, *TCC 2013: 10th Theory of Cryptography Conference*, volume 7785 of *Lecture Notes in Computer Science*, pages 315–333. Springer, Heidelberg, March 2013.

[45] Matt Blaze. Scientists say no credible evidence of computer fraud in the 2020 election outcome, but policymakers must work with experts to improve confidence. https://www.mattblaze.org/papers/election2020.pdf.

[46] Matt Blaze, 12:28 AM EST June 4, 2021. https://twitter.com/mattblaze/status/1400670767453462530.

[47] Matt Blaze. US house of representatives committee on oversight and government reform subcommittee on information technology and subcommittee on intergovernmental affairs hear on cybersecurity of voting machines, 2017. https://www.mattblaze.org/papers/blaze-govtreform-20171129.pdf.

[48] Matt Blaze. Election integrity and technology: Vulnerabilities and solutions. In *Georgetown Law Technology Review*, volume 4, pages 505–522, 2020.

[49] Matt Blaze, Jake Braun, Harri Hursti, Joseph Lorenzo Hall, Margaret MacAlpine, and Jeff Moss. Defcon 25 voting machine hacking village. *Proceedings of DEFCON, Washington DC*, pages 1–18, 2017.

[50] Matt Blaze, Jake Braun, Harri Hursti, David Jefferson, Margaret MacAlpine, and Jeff Moss. Defcon 26 voting machine hacking village. *Proceedings of DEFCON, Washington DC*, pages 1–50, 2018.

[51] Matt Blaze, Harri Hursti, Margaret MacAlpine, Mary Hanley, Jeff Moss, Rachel Wehr, Kendall Spencer, and Christopher Ferris. Defcon 27 voting machine hacking village. *Proceedings of DEFCON, Washington DC*, pages 1–47, 2019.

[52] Manuel Blum, Paul Feldman, and Silvio Micali. Non-interactive zero-knowledge and its applications (extended abstract). In *20th Annual ACM Symposium on Theory of Computing*, pages 103–112. ACM Press, May 1988.

[53] Dan Bogdanov, Liina Kamm, Baldur Kubo, Reimo Rebane, Ville Sokk, and Riivo Talviste. Students and taxes: a privacy-preserving study using secure computation. *Proceedings on Privacy Enhancing Technologies*, 2016(3):117–135, July 2016.

[54] Dan Boneh, Elette Boyle, Henry Corrigan-Gibbs, Niv Gilboa, and Yuval Ishai. Zero-knowledge proofs on secret-shared data via fully linear PCPs. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology – CRYPTO 2019, Part III*, volume 11694 of *Lecture Notes in Computer Science*, pages 67–97. Springer, Heidelberg, August 2019.

[55] Jonathan Bootle, Andrea Cerulli, Pyrros Chaidos, Jens Groth, and Christophe Petit. Efficient zero-knowledge arguments for arithmetic circuits in the discrete log setting. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology – EUROCRYPT 2016, Part II*, volume 9666 of *Lecture Notes in Computer Science*, pages 327–357. Springer, Heidelberg, May 2016.

[56] Jonathan Bootle, Andrea Cerulli, Essam Ghadafi, Jens Groth, Mohammad Hajiabadi, and Sune K. Jakobsen. Linear-time zero-knowledge proofs for arithmetic circuit satisfiability. In Tsuyoshi Takagi and Thomas Peyrin, editors, *Advances in Cryptology – ASIACRYPT 2017, Part III*, volume 10626 of *Lecture Notes in Computer Science*, pages 336–365. Springer, Heidelberg, December 2017.

[57] Amanda Bower, Sarah N Kitchen, Laura Niss, Martin J Strauss, Alexander Vargas, and Suresh Venkatasubramanian. Fair pipelines. *arXiv preprint arXiv:1707.00391*, 2017.

[58] Danah Boyd. The politics of "real names". *Communications of the ACM*, 55(8):29–31, 2012.

[59] Boyd v. United States, 116 US 616 - Supreme Court 1886.

[60] Elette Boyle, Geoffroy Couteau, Niv Gilboa, and Yuval Ishai. Compressing vector OLE. In David Lie, Mohammad Mannan, Michael Backes, and XiaoFeng Wang, editors, *ACM CCS 2018: 25th Conference on Computer and Communications Security*, pages 896–912. ACM Press, October 2018.

[61] Elette Boyle, Geoffroy Couteau, Niv Gilboa, Yuval Ishai, Lisa Kohl, and Peter Scholl. Efficient pseudorandom correlation generators: Silent OT extension and more. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology – CRYPTO 2019, Part III*, volume 11694 of *Lecture Notes in Computer Science*, pages 489–518. Springer, Heidelberg, August 2019.

[62] Ernie Brickell. A proposal for balancing access and protection requirements from law enforcement, corporations, and individuals. In *Encryption and Surveillance workshop, affiliated with Crypto 2018*, August 2018. https://crypto.iacr.org/2018/affevents/legal/medias/Ernie_Brickell.pdf.

[63] Rep. James L. Buckley. Family Educational Rights and Privacy Act of 1974, 1974.

[64] Benedikt Bünz, Jonathan Bootle, Dan Boneh, Andrew Poelstra, Pieter Wuille, and Greg Maxwell. Bulletproofs: Short proofs for confidential transactions and more. In *2018 IEEE Symposium on Security and Privacy*, pages 315–334. IEEE Computer Society Press, May 2018.

[65] Benedikt Bünz, Ben Fisch, and Alan Szepieniec. Transparent SNARKs from DARK compilers. In Anne Canteaut and Yuval Ishai, editors, *Advances in Cryptology – EUROCRYPT 2020, Part I*, volume 12105 of *Lecture Notes in Computer Science*, pages 677–706. Springer, Heidelberg, May 2020.

[66] Joy Buolamwini and Timnit Gebru. Gender shades: Intersectional accuracy disparities in commercial gender classification. In *Conference on fairness, accountability and transparency*, pages 77–91. PMLR, 2018.

[67] Jan Camenisch and Anna Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In Birgit Pfitzmann, editor, *Advances in Cryptology – EUROCRYPT 2001*, volume 2045 of *Lecture Notes in Computer Science*, pages 93–118. Springer, Heidelberg, May 2001.

[68] Jan Camenisch and Anna Lysyanskaya. Signature schemes and anonymous credentials from bilinear maps. In Matthew Franklin, editor, *Advances in Cryptology – CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 56–72. Springer, Heidelberg, August 2004.

[69] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *42nd Annual Symposium on Foundations of Computer Science*, pages 136–145. IEEE Computer Society Press, October 2001.

[70] Ran Canetti, Aloni Cohen, Nishanth Dikkala, Govind Ramnarayan, Sarah Scheffler, and Adam Smith. From soft classifiers to hard decisions: How fair can we be? In *Proceedings of the ACM conference on fairness, accountability, and transparency*, pages 309–318, 2019. https://doi.org/10.1145/3287560.3287561.

[71] Ran Canetti, Cynthia Dwork, Moni Naor, and Rafail Ostrovsky. Deniable encryption. In Burton S. Kaliski Jr., editor, *Advances in Cryptology – CRYPTO'97*, volume 1294 of *Lecture Notes in Computer Science*, pages 90–104. Springer, Heidelberg, August 1997.

[72] Ran Canetti, Sunoo Park, and Oxana Poburinnaya. Fully deniable interactive encryption. In Daniele Micciancio and Thomas Ristenpart, editors, *Advances in Cryptology – CRYPTO 2020, Part I*, volume 12170 of *Lecture Notes in Computer Science*, pages 807–835. Springer, Heidelberg, August 2020.

[73] Ran Canetti, Ben Riva, and Guy N. Rothblum. Practical delegation of computation using multiple servers. In Yan Chen, George Danezis, and Vitaly Shmatikov, editors, *ACM CCS 2011: 18th Conference on Computer and Communications Security*, pages 445–454. ACM Press, October 2011.

[74] Carpenter v. US, 138 S. Ct. 2206 - Supreme Court 2018.

[75] Aravindan Chandrabose, Bharathi Raja Chakravarthi, et al. An overview of fairness in data–illuminating the bias in data pipeline. In *Proceedings of the First Workshop on Language Technology for Equality, Diversity and Inclusion*, pages 34–45, 2021.

[76] Melissa Chase, David Derler, Steven Goldfeder, Claudio Orlandi, Sebastian Ramacher, Christian Rechberger, Daniel Slamanig, and Greg Zaverucha. Post-quantum zero-knowledge and signatures from symmetric-key primitives. In Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu, editors, *ACM CCS 2017: 24th Conference on Computer and Communications Security*, pages 1825–1842. ACM Press, October / November 2017.

[77] David Chaum. Privategrity: online communication with strong privacy, talk at Real World Cryptography, 2016. https://rwc.iacr.org/2016/program.html.

[78] David L Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 24(2):84–90, 1981.

[79] Alessandro Chiesa, Michael A. Forbes, and Nicholas Spooner. A zero knowledge sumcheck and its applications. Cryptology ePrint Archive, Report 2017/305, 2017. https://eprint.iacr.org/2017/305.

[80] Alessandro Chiesa, Yuncong Hu, Mary Maller, Pratyush Mishra, Noah Vesely, and Nicholas P. Ward. Marlin: Preprocessing zkSNARKs with universal and updatable SRS. In Anne Canteaut and Yuval Ishai, editors, *Advances in Cryptology – EUROCRYPT 2020, Part I*, volume 12105 of *Lecture Notes in Computer Science*, pages 738–768. Springer, Heidelberg, May 2020.

[81] Alexandra Chouldechova. Fair prediction with disparate impact: A study of bias in recidivism prediction instruments. *Big Data*, 5(2):153–163, 2017.

[82] Kai-Min Chung, Yael Kalai, and Salil P. Vadhan. Improved delegation of computation using fully homomorphic encryption. In Tal Rabin, editor, *Advances in Cryptology – CRYPTO 2010*, volume 6223 of *Lecture Notes in Computer Science*, pages 483–501. Springer, Heidelberg, August 2010.

[83] T Anne Cleary. Test bias: Prediction of grades of negro and white students in integrated colleges. *Journal of Educational Measurement*, 5(2):115–124, 1968.

[84] Aloni Cohen and Kobbi Nissim. Towards formalizing the gdpr's notion of singling out. *CoRR*, abs/1904.06009, 2019.

[85] Aloni Cohen and Sunoo Park. Compelled decryption and the Fifth Amendment: Exploring the technical boundaries. *Harvard Journal of Law & Technology*, 32:169–234, 2018.

[86] Aloni Cohen, Sarah Scheffler, and Mayank Varia. Telling the truth about compelled encryption and the contents of the mind. In *Privacy Law Scholars Conference*, 2021. Available upon request.

[87] Nancy S Cole and Michael J Zieky. The new faces of fairness. *Journal of Educational Measurement*, 38(4):369–382, 2001.

[88] Commonwealth v. Baust, 89 Va. Cir. 267 (2014).

[89] Commonwealth v. Davis, Pa: Supreme Court, Middle Dist. 2019.

[90] Commonwealth v. Gelfgatt, 468 Mass. 512, 11 N.E.3d 605, 11 N.E. (2014).

[91] Commonwealth v. Jones, 481 Mass. 540 - Mass: Supreme Judicial Court 2019.

[92] Sam Corbett-Davies, Emma Pierson, Avi Feller, Sharad Goel, and Aziz Huq. Algorithmic decision making and the cost of fairness. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '17, page 797–806, New York, NY, USA, 2017. Association for Computing Machinery.

[93] Henry Corrigan-Gibbs and Dan Boneh. Prio: Private, robust, and scalable computation of aggregate statistics. In Aditya Akella and Jon Howell, editors, *14th USENIX Symposium on Networked Systems Design and Implementation, NSDI 2017, Boston, MA, USA, March 27-29, 2017*, pages 259–282. USENIX Association, 2017. https://www.usenix.org/conference/nsdi17/technical-sessions/presentation/corrigan-gibbs.

[94] Craig Costello, Cédric Fournet, Jon Howell, Markulf Kohlweiss, Benjamin Kreuter, Michael Naehrig, Bryan Parno, and Samee Zahur. Geppetto: Versatile verifiable computation. In *2015 IEEE Symposium on Security and Privacy*, pages 253–270. IEEE Computer Society Press, May 2015.

[95] National Research Council et al. *Fairness in employment testing: Validity generalization, minority issues, and the General Aptitude Test Battery.* National Academies Press, 1989.

[96] Mark A Cowen. The act-of-production privilege post-Hubbell: United States v. Ponds and the relevance of the reasonable particularity and foregone conclusion doctrines. *George Mason Law Review*, 17:863, 2009.

[97] Scott Craver. Zero knowledge watermark detection. In *International Workshop on Information Hiding*, pages 101–116. Springer, 1999.

[98] Curcio v. United States, 354 U.S. 118 - Supreme Court 1957.

[99] Cybersecurity & Infrastructure Security Agency. Joint statement from elections infrastructure government coordinating council & the election infrastructure sector coordinating executive committees, November 12, 2020. https://www.cisa.gov/news/2020/11/12/joint-statement-elections-infrastructure-government-coordinating-council-election.

[100] Ivan Damgård, Valerio Pastro, Nigel P. Smart, and Sarah Zakarias. Multiparty computation from somewhat homomorphic encryption. In Reihaneh Safavi-Naini and Ran Canetti, editors, *Advances in Cryptology – CRYPTO 2012*, volume 7417 of *Lecture Notes in Computer Science*, pages 643–662. Springer, Heidelberg, August 2012.

[101] George Danezis, Cédric Fournet, Jens Groth, and Markulf Kohlweiss. Square span programs with applications to succinct NIZK arguments. In Palash Sarkar and Tetsu Iwata, editors, *Advances in Cryptology – ASIACRYPT 2014, Part I*, volume 8873 of *Lecture Notes in Computer Science*, pages 532–550. Springer, Heidelberg, December 2014.

[102] Richard B Darlington. Another look at “cultural fairness”. *Journal of Educational Measurement*, 8(2):71–82, 1971.

[103] Alex Davidson, Ian Goldberg, Nick Sullivan, George Tankersley, and Filippo Valsorda. Privacy pass: Bypassing internet challenges anonymously. *Proceedings on Privacy Enhancing Technologies*, 2018(3):164–180, July 2018.

[104] Cyprien de Saint Guilhem, Lauren De Meyer, Emmanuela Orsini, and Nigel P. Smart. BBQ: Using AES in picnic signatures. In Kenneth G. Paterson and Douglas Stebila, editors, *SAC 2019: 26th Annual International Workshop on Selected Areas in Cryptography*, volume 11959 of *Lecture Notes in Computer Science*, pages 669–692. Springer, Heidelberg, August 2019.

[105] Alfredo De Santis and Giuseppe Persiano. Zero-knowledge proofs of knowledge without interaction (extended abstract). In *33rd Annual Symposium on Foundations of Computer Science*, pages 427–436. IEEE Computer Society Press, October 1992.

[106] Akshay Degwekar, Vinod Vaikuntanathan, and Prashant Nalini Vasudevan. Fine-grained cryptography. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology – CRYPTO 2016, Part III*, volume 9816 of *Lecture Notes in Computer Science*, pages 533–562. Springer, Heidelberg, August 2016.

[107] Dorothy E. Denning and Dennis K. Branstad. A taxonomy for key escrow encryption systems. *Communications of the ACM*, 39(3):34–40, 1996.

[108] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.

[109] Whitfield Diffie and Susan Landau. *Privacy on the line: The politics of wiretapping and encryption.* The MIT Press, 2010.

[110] Roger Dingledine, Nick Mathewson, and Paul F. Syverson. Tor: The second-generation onion router. In Matt Blaze, editor, *USENIX Security 2004: 13th USENIX Security Symposium*, pages 303–320. USENIX Association, August 2004.

[111] Doe v. United States, 487 US 201 - Supreme Court 1988.

[112] Cynthia Dwork and Christina Ilvento. Fairness Under Composition. In Avrim Blum, editor, *10th Innovations in Theoretical Computer Science Conference (ITCS 2019)*, volume 124 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 33:1–33:20, Dagstuhl, Germany, 2018. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[113] Ronald Dworkin. *Taking rights seriously.* A&C Black, 2013.

[114] Frank H. Easterbrook. Cyberspace and the law of the horse. In *University of Chicago Legal Forum*, pages 207–216, 1996.

[115] Shohei Egashira, Yuyu Wang, and Keisuke Tanaka. Fine-grained cryptography revisited. In Steven D. Galbraith and Shiho Moriai, editors, *Advances in Cryptology – ASIACRYPT 2019, Part III*, volume 11923 of *Lecture Notes in Computer Science*, pages 637–666. Springer, Heidelberg, December 2019.

[116] Uriel Feige, Amos Fiat, and Adi Shamir. Zero-knowledge proofs of identity. *Journal of Cryptology*, 1(2):77–94, June 1988.

[117] Joan Feigenbaum and Daniel J. Weitzner. On the incommensurability of laws and technical mechanisms: Or, what cryptography can't do. In *26th International Security Protocols Workshop*, pages 266–279. Springer, 2018.

[118] Adrienne Porter Felt, Richard Barnes, April King, Chris Palmer, Chris Bentzel, and Parisa Tabriz. Measuring HTTPS adoption on the web. In Engin Kirda and Thomas Ristenpart, editors, *USENIX Security 2017: 26th USENIX Security Symposium*, pages 1323–1338. USENIX Association, August 2017.

[119] Felten v. RIAA, US DC NJ Case #CV-01-2669.

[120] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew M. Odlyzko, editor, *Advances in Cryptology – CRYPTO'86*, volume 263 of *Lecture Notes in Computer Science*, pages 186–194. Springer, Heidelberg, August 1987.

[121] Dario Fiore, Rosario Gennaro, and Valerio Pastro. Efficiently verifiable computation on encrypted data. In Gail-Joon Ahn, Moti Yung, and Ninghui Li, editors, *ACM CCS 2014: 21st Conference on Computer and Communications Security*, pages 844–855. ACM Press, November 2014.

[122] Ben Fisch, Daniel Freund, and Moni Naor. Physical zero-knowledge proofs of physical properties. In Juan A. Garay and Rosario Gennaro, editors, *Advances in Cryptology – CRYPTO 2014*, pages 313–336, Berlin, Heidelberg, 2014. Springer Berlin Heidelberg.

[123] Fisher v. United States, 425 US 391 - Supreme Court 1976.

[124] Anthony W Flores, Kristin Bechtel, and Christopher T Lowenkamp. False positives, false negatives, and false analyses: A rejoinder to machine bias: There's software used across the country to predict future criminals. and it's biased against blacks. *Federal Probation*, 80:38, 2016.

[125] 47 U.S. Code §230 Protection for private blocking and screening of offensive material.

[126] Jonathan Frankle, Sunoo Park, Daniel Shaar, Shafi Goldwasser, and Daniel J. Weitzner. Practical accountability of secret processes. In William Enck and Adrienne Porter Felt, editors, *USENIX Security 2018: 27th USENIX Security Symposium*, pages 657–674. USENIX Association, August 2018.

[127] Tore Kasper Frederiksen, Jesper Buus Nielsen, and Claudio Orlandi. Privacy-free garbled circuits with applications to efficient zero-knowledge. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology – EUROCRYPT 2015, Part II*, volume 9057 of *Lecture Notes in Computer Science*, pages 191–219. Springer, Heidelberg, April 2015.

[128] David Froelicher, Juan R Troncoso-Pastoriza, Jean Louis Raisaro, Michel Cuendet, Joao Sa Sousa, Jacques Fellay, and Jean-Pierre Hubaux. Truly privacy-preserving federated analytics for precision medicine with multiparty homomorphic encryption. *bioRxiv*, 2021.

[129] Brian Fung and Rachel Metz. This may be America's first known wrongful arrest involving facial recognition, 2020. https://www.cnn.com/2020/06/24/tech/aclu-mistaken-facial-recognition/index.html.

[130] Ariel Gabizon. AuroraLight: Improved prover efficiency and SRS size in a sonic-like system. Cryptology ePrint Archive, Report 2019/601, 2019. https://eprint.iacr.org/2019/601.

[131] Ariel Gabizon, Zachary J. Williamson, and Oana Ciobotaru. PLONK: Permutations over lagrange-bases for oecumenical noninteractive arguments of knowledge. Cryptology ePrint Archive, Report 2019/953, 2019. https://eprint.iacr.org/2019/953.

[132] Galois. *The Future of Voting: End-to-end verifiable internet voting, specification and feasibility assessment study.* U.S. Vote Foundation, July 2015. https://usvotefoundation-drupal.s3.amazonaws.com/prod/E2EVIV_full_report.pdf.

[133] Chaya Ganesh, Yashvanth Kondi, Arpita Patra, and Pratik Sarkar. Efficient adaptively secure zero-knowledge from garbled circuits. In Michel Abdalla and Ricardo Dahab, editors, *PKC 2018: 21st International Conference on Theory and Practice of Public Key Cryptography, Part II*, volume 10770 of *Lecture Notes in Computer Science*, pages 499–529. Springer, Heidelberg, March 2018.

[134] Sanjam Garg, Shafi Goldwasser, and Prashant Nalini Vasudevan. Formalizing data deletion in the context of the right to be forgotten. In Anne Canteaut and Yuval Ishai, editors, *Advances in Cryptology – EUROCRYPT 2020, Part II*, volume 12106 of *Lecture Notes in Computer Science*, pages 373–402. Springer, Heidelberg, May 2020.

[135] Urs Gasser. Cloud innovation and the law: Issues, approaches, and interplay. *Berkman Center Research Publication*, 7 2014.

[136] Rosario Gennaro, Craig Gentry, Bryan Parno, and Mariana Raykova. Quadratic span programs and succinct NIZKs without PCPs. In Thomas Johansson and Phong Q. Nguyen, editors, *Advances in Cryptology – EUROCRYPT 2013*, volume 7881 of *Lecture Notes in Computer Science*, pages 626–645. Springer, Heidelberg, May 2013.

[137] Craig Gentry and Daniel Wichs. Separating succinct non-interactive arguments from all falsifiable assumptions. In Lance Fortnow and Salil P. Vadhan, editors, *43rd Annual ACM Symposium on Theory of Computing*, pages 99–108. ACM Press, June 2011.

[138] Irene Giacomelli, Jesper Madsen, and Claudio Orlandi. ZKBoo: Faster zero-knowledge for Boolean circuits. In Thorsten Holz and Stefan Savage, editors, *USENIX Security 2016: 25th USENIX Security Symposium*, pages 1069–1083. USENIX Association, August 2016.

[139] Gilbert v. California, 388 US 263 - Supreme Court 1967.

[140] Tarleton Gillespie. *Custodians of the Internet: Platforms, content moderation, and the hidden decisions that shape social media.* Yale University Press, 2018.

[141] Alexander Glaser, Boaz Barak, and Robert J Goldston. A zero-knowledge protocol for nuclear warhead verification. *Nature*, 510(7506):497–502, 2014.

[142] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or A completeness theorem for protocols with honest majority. In Alfred Aho, editor, *19th Annual ACM Symposium on Theory of Computing*, pages 218–229. ACM Press, May 1987.

[143] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to prove all NP-statements in zero-knowledge, and a methodology of cryptographic protocol design. In Andrew M. Odlyzko, editor, *Advances in Cryptology – CRYPTO'86*, volume 263 of *Lecture Notes in Computer Science*, pages 171–185. Springer, Heidelberg, August 1987.

[144] Oded Goldreich, Silvio Micali, and Avi Wigderson. Proofs that yield nothing but their validity or all languages in NP have zero-knowledge proof systems. *Journal of the ACM*, 38(3):691–729, 1991.

[145] Shafi Goldwasser and Silvio Micali. Probabilistic encryption and how to play mental poker keeping secret all partial information. In *14th Annual ACM Symposium on Theory of Computing*, pages 365–377. ACM Press, May 1982.

[146] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof systems. *SIAM Journal on Computing*, 18(1):186–208, 1989.

[147] Shafi Goldwasser and Sunoo Park. Public accountability vs. secret laws: Can they coexist? a cryptographic proposal. In *Proceedings of the 2017 on Workshop on Privacy in the Electronic Society*, pages 99–110, 2017.

[148] Google AI. Responsible ai practices. https://ai.google/responsibilities/responsible-ai-practices.

[149] Sen. Lindsey Graham, Sen. Richard Blumenthal, Sen. Kevin Cramer, Sen. Dianne Feinstein, Sen. Josh Hawley, Sen. Doug Jones, Sen. Robert Casey, Sen. Sheldon Whitehouse, Sen. Richard Durbin, Sen. Joni Ernst, Sen. John Kennedy, Sen Ted Cruz, Sen. Chuck Grassley, Sen. Rob Portman, Sen. Lisa Murkowski, Sen. John Cornyn, and Sen. Kelly Loeffler. S.3398 - EARN IT act of 2020, 2020.

[150] Sen. Phil Gramm. Financial Services Modernization Act of 1999, 1999.

[151] Jens Groth. Non-interactive zero-knowledge arguments for voting. In John Ioannidis, Angelos Keromytis, and Moti Yung, editors, *ACNS 05: 3rd International Conference on Applied Cryptography and Network Security*, volume 3531 of *Lecture Notes in Computer Science*, pages 467–482. Springer, Heidelberg, June 2005.

[152] Jens Groth. Short pairing-based non-interactive zero-knowledge arguments. In Masayuki Abe, editor, *Advances in Cryptology – ASIACRYPT 2010*, volume 6477 of *Lecture Notes in Computer Science*, pages 321–340. Springer, Heidelberg, December 2010.

[153] Jens Groth. On the size of pairing-based non-interactive arguments. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology – EUROCRYPT 2016, Part II*, volume 9666 of *Lecture Notes in Computer Science*, pages 305–326. Springer, Heidelberg, May 2016.

[154] Jens Groth, Markulf Kohlweiss, Mary Maller, Sarah Meiklejohn, and Ian Miers. Updatable and universal common reference strings with applications to zk-SNARKs. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology – CRYPTO 2018, Part III*, volume 10993 of *Lecture Notes in Computer Science*, pages 698–728. Springer, Heidelberg, August 2018.

[155] Yaron Gvili, Julie Ha, Sarah Scheffler, Mayank Varia, Ziling Yang, and Xinyuan Zhang. Turboikos: Improved non-interactive zero knowledge and post-quantum signatures. In Kazue Sako and Nils Ole Tippenhauer, editors, *Applied Cryptography and Network Security*, volume 12727 of *Lecture Notes in Computer Science*, Kamakura, Japan, June 2021. Springer, Heidelberg. https://doi.org/10.1007/978-3-030-78375-4_15.

[156] Yaron Gvili, Julie Ha, Sarah Scheffler Mayank Varia, Ziling Yang, and Xinyuan Zhang. TurboIKOS. https://github.com/sarahscheffler/TurboIKOS, 2021.

[157] Yaron Gvili, Sarah Scheffler, and Mayank Varia. Booligero: Improved sublinear zero knowledge proofs for boolean circuits. In *Financial Cryptography and Data Security (forthcoming)*, Lecture Notes in Computer Science. Springer, Heidelberg, March 2021.

[158] Oliver L Haimson and Anna Lauren Hoffmann. Constructing and enforcing "authentic" identity online: Facebook, real names, and non-normative identities. *First Monday*, 2016.

[159] Hale v. Henkel, 201 US 43 - Supreme Court 1906.

[160] Alexander Hamilton, James Madison, and John Jay. *The Federalist No. 78: No judicial supremacy*, pages 471–479. Bantam Dell, 1788.

[161] TX Hammes. The future of warfare: Small, many, smart vs. few & exquisite? *War on the Rocks*, 7, 2015. https://warontherocks.com/2014/07/the-future-of-warfare-small-many-smart-vs-few-exquisite/.

[162] Moritz Hardt, Eric Price, and Nati Srebro. Equality of opportunity in supervised learning. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 3315–3323, 2016.

[163] David Heath and Vladimir Kolesnikov. Stacked garbling for disjunctive zero-knowledge proofs. In Anne Canteaut and Yuval Ishai, editors, *Advances in Cryptology – EUROCRYPT 2020, Part III*, volume 12107 of *Lecture Notes in Computer Science*, pages 569–598. Springer, Heidelberg, May 2020.

[164] Úrsula Hébert-Johnson, Michael P. Kim, Omer Reingold, and Guy N. Rothblum. Multicalibration: Calibration for the (computationally-identifiable) masses. In Jennifer G. Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 1944–1953. PMLR, 2018.

[165] Hiibel v. Sixth Judicial Dist. Court of Nevada, Humboldt County, 542 U.S. 177 - Supreme Court 2004.

[166] J Henry Hinnefeld, Peter Cooman, Nat Mammo, and Rupert Deese. Evaluating fairness metrics in the presence of dataset bias. *arXiv preprint arXiv:1809.09245*, 2018.

[167] Martin Hirt, Ueli Maurer, Daniel Tschudi, and Vassilis Zikas. Network-hiding communication and applications to multi-party protocols. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology – CRYPTO 2016, Part II*, volume 9815 of *Lecture Notes in Computer Science*, pages 335–365. Springer, Heidelberg, August 2016.

[168] Hoffman v. United States, 341 US 479 - Supreme Court 1951.

[169] Max Hoffmann, Michael Klooß, and Andy Rupp. Efficient zero-knowledge arguments in the discrete log setting, revisited. In Lorenzo Cavallaro, Johannes Kinder, XiaoFeng Wang, and Jonathan Katz, editors, *ACM CCS 2019: 26th Conference on Computer and Communications Security*, pages 2093–2110. ACM Press, November 2019.

[170] Thibaut Horel, Sunoo Park, Silas Richelson, and Vinod Vaikuntanathan. How to subvert backdoored encryption: Security against adversaries that decrypt all ciphertexts. In Avrim Blum, editor, *ITCS 2019: 10th Innovations in*

Theoretical Computer Science Conference, volume 124, pages 42:1–42:20. LIPIcs, January 2019.

[171] Ben Hutchinson and Margaret Mitchell. 50 years of test (un) fairness: Lessons for machine learning. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, pages 49–58, 2019. https://www.youtube.com/watch?v=nksfAaEJtm8.

[172] In re Grand Jury Proceedings, 41 F. 3d 377 - Court of Appeals, 8th Circuit 1994.

[173] In re Grand Jury Subpoena, 383 F. 3d 905 - Court of Appeals, 9th Circuit 2004.

[174] In re Grand Jury Subpoena Duces Tecum, 1 F. 3d 87 - Court of Appeals, 2nd Circuit 1993.

[175] In re Grand Jury Subpoena Duces Tecum Dated March 25, 2011 (United States v. Doe), 670 F.3d 1335 - 11th Circuit 2012.

[176] In re Grand Jury Subpoena to Sebasetien Boucher, No. 2:06-mJ-91, 2009 WL 424718.

[177] Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. Zero-knowledge from secure multiparty computation. In David S. Johnson and Uriel Feige, editors, *39th Annual ACM Symposium on Theory of Computing*, pages 21–30. ACM Press, June 2007.

[178] Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. Cryptography with constant computational overhead. In Richard E. Ladner and Cynthia Dwork, editors, *40th Annual ACM Symposium on Theory of Computing*, pages 433–442. ACM Press, May 2008.

[179] Eric Jardine. The dark web dilemma: Tor, anonymity and online policing. *Global Commission on Internet Governance Paper Series, No. 21*, 2015.

[180] Eric Jardine, Andrew M Lindner, and Gareth Owenson. The potential harms of the tor anonymity network cluster disproportionately in free countries. *Proceedings of the National Academy of Sciences*, 117(50):31716–31721, 2020.

[181] Joseph Jarone. An act of decryption doctrine: Clarifying the act of production doctrine's application to compelled decryption. *FIU Law Review*, 10:767, 2014.

[182] Marek Jawurek, Florian Kerschbaum, and Claudio Orlandi. Zero-knowledge using garbled circuits: how to prove non-algebraic statements efficiently. In Ahmad-Reza Sadeghi, Virgil D. Gligor, and Moti Yung, editors, *ACM CCS*

2013: 20th Conference on Computer and Communications Security, pages 955–966. ACM Press, November 2013.

[183] Sen. Samuel Ervin Jr. Privacy Act of 1974, 1974.

[184] David Kahn. *The Codebreakers: The comprehensive history of secret communication from ancient times to the internet.* Simon and Schuster, 1996.

[185] Daniel Kales and Greg Zaverucha. An attack on some signature schemes constructed from five-pass identification schemes. In Stephan Krenn, Haya Shulman, and Serge Vaudenay, editors, *CANS 20: 19th International Conference on Cryptology and Network Security*, volume 12579 of *Lecture Notes in Computer Science*, pages 3–22. Springer, Heidelberg, December 2020.

[186] Seny Kamara. Restructuring the NSA metadata program. In Rainer Böhme, Michael Brenner, Tyler Moore, and Matthew Smith, editors, *FC 2014 Workshops*, volume 8438 of *Lecture Notes in Computer Science*, pages 235–247. Springer, Heidelberg, March 2014.

[187] Jonathan Katz. Universally composable multi-party computation using tamper-proof hardware. In Moni Naor, editor, *Advances in Cryptology – EUROCRYPT 2007*, volume 4515 of *Lecture Notes in Computer Science*, pages 115–128. Springer, Heidelberg, May 2007.

[188] Jonathan Katz, Vladimir Kolesnikov, and Xiao Wang. Improved non-interactive zero knowledge with applications to post-quantum signatures. In David Lie, Mohammad Mannan, Michael Backes, and XiaoFeng Wang, editors, *ACM CCS 2018: 25th Conference on Computer and Communications Security*, pages 525–537. ACM Press, October 2018.

[189] Michael Kearns, Seth Neel, Aaron Roth, and Zhiwei Steven Wu. Preventing fairness gerrymandering: Auditing and learning for subgroup fairness. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 2564–2572, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018. PMLR.

[190] Marcel Keller, Valerio Pastro, and Dragos Rotaru. Overdrive: Making SPDZ great again. In Jesper Buus Nielsen and Vincent Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2018, Part III*, volume 10822 of *Lecture Notes in Computer Science*, pages 158–189. Springer, Heidelberg, April / May 2018.

[191] John A Kemp. The background of the fifth amendment in english law: A study of its historical implications. *William & Mary Law Review*, 1:247, 1957.

[192] Helen Kennedy. Beyond anonymity, or future directions for internet identity research. *New media & society*, 8(6):859–876, 2006.

[193] Orin S Kerr. Compelled decryption and the privilege against self-incrimination. *Texas Law Review*, 97:767, 2018.

[194] Orin S Kerr, October 2020. https://twitter.com/OrinKerr/status/1313224067214835713?s=20.

[195] Orin S Kerr. Decryption originalism: The lessons of Burr. *Available at SSRN*, 2020.

[196] Orin S Kerr, May 2021. https://twitter.com/OrinKerr/status/1394285391973261315?s=20.

[197] Joe Kilian. A note on efficient zero-knowledge proofs and arguments (extended abstract). In *24th Annual ACM Symposium on Theory of Computing*, pages 723–732. ACM Press, May 1992.

[198] Jeffrey Kiok. Missing the metaphor: Compulsory decryption and the fifth amendment. *Boston University Public Interest Law Journal*, 24:53–80, 2015.

[199] James J Kirkpatrick. *Testing and fair employment: Fairness and validity of personnel tests for different ethnic groups.* New York University Press; University of London Press, 1968.

[200] Jon Kleinberg, Sendhil Mullainathan, and Manish Raghavan. Inherent Trade-Offs in the Fair Determination of Risk Scores. In Christos H. Papadimitriou, editor, *8th Innovations in Theoretical Computer Science Conference (ITCS 2017)*, volume 67 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 43:1–43:23, Dagstuhl, Germany, 2017. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[201] Yashvanth Kondi and Arpita Patra. Privacy-free garbled circuits for formulas: Size zero and information-theoretic. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology – CRYPTO 2017, Part I*, volume 10401 of *Lecture Notes in Computer Science*, pages 188–222. Springer, Heidelberg, August 2017.

[202] Vassiliki Koufi, Flora Malamateniou, Aggeliki Tsohou, and George Vassilacopoulos. A framework for privacy-preserving access to next-generation ehrs. In *e-Health–For Continuity of Care*, pages 740–744. IOS Press, 2014.

[203] Stanton D Krauss. The life and times of boyd v united states (1886-1976). *Michigan Law Review*, 76(1), 1977.

[204] Joshua A. Kroll, Edward W. Felten, and Dan Boneh. Secure protocols for accountable warrant execution, 2014. https://www.cs.princeton.edu/~felten/warrant-paper.pdf.

[205] Anunay Kulshrestha and Jonathan Mayer. Identifying harmful media in end-to-end encrypted communication: Efficient private membership computation. In *30th {USENIX} Security Symposium ({USENIX} Security 21)*, 2021.

[206] Kyllo v. United States, 533 US 27 - Supreme Court 2001.

[207] Rio LaVigne, Andrea Lincoln, and Virginia Vassilevska Williams. Public-key cryptography in the fine-grained setting. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology – CRYPTO 2019, Part III*, volume 11694 of *Lecture Notes in Computer Science*, pages 605–635. Springer, Heidelberg, August 2019.

[208] Rio LaVigne, Chen-Da Liu Zhang, Ueli Maurer, Tal Moran, Marta Mularczyk, and Daniel Tschudi. Topology-hiding computation for networks with unknown delays. In Aggelos Kiayias, Markulf Kohlweiss, Petros Wallden, and Vassilis Zikas, editors, *PKC 2020: 23rd International Conference on Theory and Practice of Public Key Cryptography, Part II*, volume 12111 of *Lecture Notes in Computer Science*, pages 215–245. Springer, Heidelberg, May 2020.

[209] Lawrence Lessig. The law of the horse: What cyberlaw might teach. *Harvard law review*, 113(2):501–549, 1999.

[210] Leonard W. Levy. *Origins of the Fifth Amendment: the right against self-incrimination.* New York: Oxford University Press, 1968.

[211] Chao Lin, Debiao He, Neeraj Kumar, Kim-Kwang Raymond Choo, Alexey Vinel, and Xinyi Huang. Security and privacy for the internet of drones: Challenges and solutions. *IEEE Communications Magazine*, 56(1):64–69, 2018.

[212] Yehuda Lindell. How to simulate it - A tutorial on the simulation proof technique. Cryptology ePrint Archive, Report 2016/046, 2016. https://eprint.iacr.org/2016/046.

[213] Yehuda Lindell and Benny Pinkas. An efficient protocol for secure two-party computation in the presence of malicious adversaries. *Journal of Cryptology*, 28(2):312–350, April 2015.

[214] Helger Lipmaa. Succinct non-interactive zero knowledge arguments from span programs and linear error-correcting codes. In Kazue Sako and Palash Sarkar, editors, *Advances in Cryptology – ASIACRYPT 2013, Part I*, volume 8269 of

Lecture Notes in Computer Science, pages 41–60. Springer, Heidelberg, December 2013.

[215] Carsten Lund, Lance Fortnow, Howard J. Karloff, and Noam Nisan. Algebraic methods for interactive proof systems. In *31st Annual Symposium on Foundations of Computer Science*, pages 2–10. IEEE Computer Society Press, October 1990.

[216] Maggie MacAlpine, 3:18 PM EST January 11, 2021. https://twitter.com/MaggieMacAlpine/status/1348726036771590153.

[217] David Madras, Toniann Pitassi, and Richard S. Zemel. Predict responsibly: Increasing fairness by learning to defer. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Workshop Track Proceedings*. OpenReview.net, 2018.

[218] Mary Maller, Sean Bowe, Markulf Kohlweiss, and Sarah Meiklejohn. Sonic: Zero-knowledge SNARKs from linear-size universal and updatable structured reference strings. In Lorenzo Cavallaro, Johannes Kinder, XiaoFeng Wang, and Jonathan Katz, editors, *ACM CCS 2019: 26th Conference on Computer and Communications Security*, pages 2111–2128. ACM Press, November 2019.

[219] Simcha Mandelbaum. The privilege against self-incrimination in Anglo-American and Jewish law. *The American Journal of Comparative Law*, pages 115–119, 1956.

[220] Matter of Harris, 221 US 274 - Supreme Court 1911.

[221] Matter of Residence in Oakland, California, 354 F. Supp. 3d 1010 - Dist. Court, ND California 2019.

[222] Nathan K. McGregor. The weak protection of strong encryption: Passwords, privacy, and Fifth Amendment privilege. *Vanderbilt Journal of Entertainment & Technology Law*, 12:581–609, 2010.

[223] Ralph C Merkle. Secure communications over insecure channels. *Communications of the ACM*, 21(4):294–299, 1978.

[224] Microsoft Corporation. Picnic. https://microsoft.github.io/Picnic/.

[225] Ian Miers, Christina Garman, Matthew Green, and Aviel D. Rubin. Zerocoin: Anonymous distributed E-cash from Bitcoin. In *2013 IEEE Symposium on Security and Privacy*, pages 397–411. IEEE Computer Society Press, May 2013.

[226] David R Millen and John F Patterson. Identity disclosure and the creation of social capital. In *CHI'03 extended abstracts on Human factors in computing systems*, pages 720–721, 2003.

[227] Bhabendu Kumar Mohanta, Soumyashree S Panda, and Debasish Jena. An overview of smart contract and use cases in blockchain technology. In *2018 9th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, pages 1–4. IEEE, 2018.

[228] Mooney v. Holohan, 294 US 103 - Supreme Court 1935.

[229] Andrew Morgan and Rafael Pass. Paradoxes in fair computer-aided decision making. In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, AIES '19, page 85–90, New York, NY, USA, 2019. Association for Computing Machinery.

[230] Zachary Morris. US drones: Smaller, less capable drones for the near future. *Military Review*, 98(3):38, 2018.

[231] Muhammad Munir. Fundamental guarantees of the rights of the accused in islamic criminal justice system. *Hamdard Islamicus*, 40:45–65, 2017.

[232] Moni Naor and Benny Pinkas. Oblivious transfer and polynomial evaluation. In *31st Annual ACM Symposium on Theory of Computing*, pages 245–254. ACM Press, May 1999.

[233] National Academies of Sciences, Engineering, and Medicine. *Securing the Vote: Protecting American Democracy.* The National Academies Press, Washington, DC, 2018.

[234] National Association of Criminal Defense Lawyers. Compelled decryption primer, 2019. https://www.nacdl.org/Content/Compelled-Decryption-Primer.

[235] National Bureau of Standards. Cryptographic algorithms for protection of computer data during transmission and dormant storage. *Federal Register*, 38(93):12723–12789, 1973.

[236] C. Andrew Neff. A verifiable secret shuffle and its application to e-voting. In Michael K. Reiter and Pierangela Samarati, editors, *ACM CCS 2001: 8th Conference on Computer and Communications Security*, pages 116–125. ACM Press, November 2001.

[237] Jesper Buus Nielsen, Peter Sebastian Nordholt, Claudio Orlandi, and Sai Sheshank Burra. A new approach to practical active-secure two-party computation. In Reihaneh Safavi-Naini and Ran Canetti, editors, *Advances in Cryptology – CRYPTO 2012*, volume 7417 of *Lecture Notes in Computer Science*, pages 681–700. Springer, Heidelberg, August 2012.

[238] Jesper Buus Nielsen and Claudio Orlandi. LEGO for two-party secure computation. In Omer Reingold, editor, *TCC 2009: 6th Theory of Cryptography Conference*, volume 5444 of *Lecture Notes in Computer Science*, pages 368–386. Springer, Heidelberg, March 2009.

[239] Kobbi Nissim, Aaron Bembenek, Alexandra Wood, Mark Bun, Marco Gaboardi, Urs Gasser, David R. O'Brien, and Salil Vadhan. Bridging the gap between computer science and legal approaches to privacy. In *Harvard Journal of Law & Technology*, volume 31, pages 687–780, 2016 2018.

[240] 13 U.S. Code §195 Use of sampling. https://www.law.cornell.edu/uscode/text/13/195.

[241] Office of the Director of National Intelligence. Assessing russian activities and intentions in recent us elections, intelligence community assessment, January 2017. https://www.dni.gov/files/documents/ICA_2017_01.pdf.

[242] Kellie Ottoboni and Philip B Stark. Election integrity and electronic voting machines in 2018 Georgia, USA. In *International Joint Conference on Electronic Voting*, pages 166–182. Springer, 2019.

[243] Bryan Parno, Jon Howell, Craig Gentry, and Mariana Raykova. Pinocchio: Nearly practical verifiable computation. In *2013 IEEE Symposium on Security and Privacy*, pages 238–252. IEEE Computer Society Press, May 2013.

[244] Juha Partala, Tri Hong Nguyen, and Susanna Pirttikangas. Non-interactive zero-knowledge for blockchain: A survey. *IEEE Access*, 8:227945–227961, 2020.

[245] Nancy S Petersen and Melvin R Novick. An evaluation of some models for culture-fair selection. *Journal of Educational Measurement*, pages 3–29, 1976.

[246] Riana Pfefferkorn. The earn it act: How to ban end-to-end encryption without actually banning it, January 2020. http://cyberlaw.stanford.edu/blog/2020/01/earn-it-act-how-ban-end-end-encryption-without-actually-banning-it.

[247] Minerva Pinto. The future of the foregone conclusion doctrine and compelled decryption in the age of cloud computing. *Temple Political & Civil Rights Law Review*, 25:223, 2016.

[248] Geoff Pleiss, Manish Raghavan, Felix Wu, Jon Kleinberg, and Kilian Q Weinberger. On fairness and calibration. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5680–5689. Curran Associates, Inc., 2017.

[249] Harsha S Gardiyawasam Pussewalage and Vladimir A Oleshchuk. Privacy preserving mechanisms for enforcing security and privacy requirements in e-health solutions. *International Journal of Information Management*, 36(6):1161–1173, 2016.

[250] Cooper Quintin. Tor is for everyone: why you should use Tor, 2014. https://gizmodo.com/tor-is-for-everyone-why-you-should-use-tor-1591191905.

[251] Siladitya Ray. Facebook gets preliminary approval to settle facial recognition lawsuit, 2020. https://www.forbes.com/sites/siladityaray/2020/08/20/facebook-gets-preliminary-approval-to-settle-facial-recognition-lawsuit.

[252] Omer Reingold, Guy N. Rothblum, and Ron D. Rothblum. Constant-round interactive proofs for delegating computation. In Daniel Wichs and Yishay Mansour, editors, *48th Annual ACM Symposium on Theory of Computing*, pages 49–62. ACM Press, June 2016.

[253] Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the Association for Computing Machinery*, 21(2):120–126, 1978.

[254] Phillip Rogaway. The moral character of cryptographic work. Cryptology ePrint Archive, Report 2015/1162, 2015. https://eprint.iacr.org/2015/1162.

[255] Bertrand Russell. Scientists appeal for abolition of war. *Bulletin of the Atomic Scientists*, 11(7):236–237, 1955.

[256] Laurent Sacharoff. Unlocking the Fifth Amendment: Passwords and encrypted devices. *Fordham Law Review*, 87:203–251, 2018.

[257] Babak Salimi, Bill Howe, and Dan Suciu. Data management for causal algorithmic fairness. *arXiv preprint arXiv:1908.07924*, 2019.

[258] Stefan Savage. Lawful device access without mass surveillance risk: A technical design discussion. In David Lie, Mohammad Mannan, Michael Backes, and XiaoFeng Wang, editors, *ACM CCS 2018: 25th Conference on Computer and Communications Security*, pages 1761–1774. ACM Press, October 2018.

[259] Sarah Scheffler and Jacob Ostling. Dismantling false assumptions about autonomous weapon systems. *Unpublished manuscript (ACM CS-Law Student Paper Competition)*, October 2019.

[260] Sarah Scheffler, Sean Smith, Yossi Gilad, and Sharon Goldberg. The unintended consequences of email spam prevention. In *International Conference on Passive and Active Network Measurement*, pages 158–169. Springer, 2018.

[261] Sarah Scheffler and Mayank Varia. Protecting cryptography against compelled self-incrimination. *USENIX Security 2021: 30th USENIX Security Symposium*, 2021.

[262] James Scheibner, Jean Louis Raisaro, Juan Ramón Troncoso-Pastoriza, Marcello Ienca, Jacques Fellay, Effy Vayena, and Jean-Pierre Hubaux. Revolutionizing medical data sharing using advanced privacy-enhancing technologies: Technical, legal, and ethical synthesis. *Journal of Medical Internet Research*, 23(2):e25120, 2021.

[263] Schmerber v. California, 384 U.S. 757 - Supreme Court 1966.

[264] Sec. & Exch. Comm'n v. Huang, No. CV 15-269, 2015 WL 5611644 (E.D. Pa. Sept. 23, 2015).

[265] Aaron Segal, Bryan Ford, and Joan Feigenbaum. Catching bandits and only bandits: Privacy-preserving intersection warrants for lawful surveillance. In *4th USENIX Workshop on Free and Open Communications on the Internet*. USENIX Association, 2014.

[266] Seo v. State, 109 N.E.3d 418 (Ind. Ct. App. 2018).

[267] Srinath Setty. Spartan: Efficient and general-purpose zkSNARKs without trusted setup. In Daniele Micciancio and Thomas Ristenpart, editors, *Advances in Cryptology – CRYPTO 2020, Part III*, volume 12172 of *Lecture Notes in Computer Science*, pages 704–737. Springer, Heidelberg, August 2020.

[268] Srinath Setty and Jonathan Lee. Quarks: Quadruple-efficient transparent zkSNARKs. Cryptology ePrint Archive, Report 2020/1275, 2020. https://eprint.iacr.org/2020/1275.

[269] Shapiro v. United States, 335 US 1 - Supreme Court 1948.

[270] Slochower v. Board of Higher Ed. of New York City, 350 US 551 - Supreme Court 1956.

[271] Matthew Smith and Matthew Green. A discussion of surveillance backdoors: Effectiveness, collateral damage and ethics. In James Bindenagel, Matthias Herdegen, and Karl Kaiser, editors, *International Security in the 21st Century: Germany's International Responsibility*, pages 131 – 142. Bonn

University Press, Göttingen, Germany, February 2017. https://doi.org/10.14220/9783737007627.131.

[272] Michael A. Specter, James Koppel, and Daniel J. Weitzner. The ballot is busted before the blockchain: A security analysis of voatz, the first internet voting application used in U.S. federal elections. In Srdjan Capkun and Franziska Roesner, editors, *USENIX Security 2020: 29th USENIX Security Symposium*, pages 1535–1553. USENIX Association, August 2020.

[273] State v. Andrews, 197 A. 3d 200 - NJ: Appellate Div. 2018.

[274] State v. Stahl, 206 So. 3d 124 (Fla. Dist. Ct. App. 2016).

[275] Sol. Steinmetz and Robert K. Barnhart. *The Barnhart dictionary of etymology.* H.W. Wilson Co., Bronx, N.Y., 1988.

[276] John Suler. The online disinhibition effect. *Cyberpsychology & behavior*, 7(3):321–326, 2004.

[277] Nick Szabo. Formalizing and securing relationships on public networks. *First Monday*, 1997.

[278] Matt Tait. Going dark, crypto wars, and cryptographic safety valves. In *Encryption and Surveillance workshop, affiliated with Crypto 2018*, August 2018. https://crypto.iacr.org/2018/affevents/legal/medias/Matt_Tait.pptx.

[279] Muhammad Tanveer, Neeraj Kumar, Mohammad Mehedi Hassan, et al. RAMP-IoD: A robust authenticated key management protocol for the internet of drones. *IEEE Internet of Things Journal*, 2021.

[280] Muhammad Tanveer, Amjad Hussain Zahid, Musheer Ahmad, Abdullah Baz, and Hosam Alhakami. Lake-iod: Lightweight authenticated key exchange protocol for the internet of drone environment. *IEEE Access*, 8:155645–155659, 2020.

[281] Terry v. Ohio, 392 US 1 - Supreme Court 1968.

[282] Dan Terzian. The fifth amendment, encryption, and the forgotten state interest. *UCLA Law Review*, 61:298–312, 2014.

[283] Dan Terzian. Forced decryption as a foregone conclusion. *6 California Law Review Circuit 27*, 2015.

[284] The Law Library of Congress. Miranda warning equivalents abroad, 2016. https://www.loc.gov/law/help/miranda-warning-equivalents-abroad/miranda-warning-equivalents-abroad.pdf.

[285] The State of Alabama et al. v. United States Department of Commerce et al., No. 3:2021cv00211.

[286] Robert L Thorndike. Concepts of culture-fairness. *Journal of Educational Measurement*, 8(2):63–70, 1971.

[287] Tor Project. Tor FAQ, 2019. https://2019.www.torproject.org/docs/faq.html.en#WhatProtectionsDoesTorProvide.

[288] Trail of Bits. Reverie. https://github.com/trailofbits/reverie, 2021.

[289] Meltem Sönmez Turan, Elaine B Barker, William E Burr, and Lidong Chen. Sp 800-132. recommendation for password-based key derivation: Part 1: Storage applications, 2010. https://doi.org/10.6028/NIST.SP.800-132.

[290] Sherry Turkle. *Life on the Screen*. Simon and Schuster, 2011.

[291] Nirvan Tyagi, Muhammad Haris Mughees, Thomas Ristenpart, and Ian Miers. BurnBox: Self-revocable encryption in a world of compelled access. In William Enck and Adrienne Porter Felt, editors, *USENIX Security 2018: 27th USENIX Security Symposium*, pages 445–461. USENIX Association, August 2018.

[292] Ullmann v. United States, 350 U.S. 422 - Supreme Court 1956.

[293] United States Census Bureau. Title 13, U.S. Code. https://www.census.gov/history/www/reference/privacy_confidentiality/title_13_us_code.html.

[294] United States v. Doe, 465 US 605 - Supreme Court 1984.

[295] United States v. Hubbell, 530 US 27 - Supreme Court 2000.

[296] United States v. Kirschner, 823 F. Supp. 2d 665 - Eastern District of Michigan 2010.

[297] United States v. Miller, 425 US 435 - Supreme Court 1976.

[298] United States v. Spencer, No. 17-cr-00259-CRB-1 (N.D. Cal. Apr. 26, 2018).

[299] United States v. White, 322 US 694 - Supreme Court 1944.

[300] U.S. Constitution Amendment V.

[301] U.S. Constitution Amendment XIV.

[302] US v. Apple MacPro Computer, 851 F.3d 238 - Court of Appeals, 3rd Circuit 2017.

[303] US v. Burns, Dist. Court, MD North Carolina 2019.

[304] US v. Fricosu, 841 F. Supp. 2d 1232 - Dist. Court, D. Colorado 2012.

[305] US v. Greenfield, 831 F. 3d 106 - Court of Appeals, 2nd Circuit 2016.

[306] US v. Maffei, Dist. Court, ND California 2019.

[307] US v. Ponds, 454 F. 3d 313 - Court of Appeals, Dist. of Columbia Circuit 2006.

[308] Muthuramakrishnan Venkitasubramaniam. personal communication, September 2020.

[309] VeraCrypt. Plausible deniability. https://veracrypt.fr/en/Plausible%20Deniability.html.

[310] Bas Vergouw, Huub Nagel, Geert Bondt, and Bart Custers. Drone technology: Types, payloads, applications, frequency spectrum issues and future developments. In *The future of drone use*, pages 21–45. Springer, 2016.

[311] Victor v. Nebraska, 511 US 1 - Supreme Court 1994.

[312] Riad S. Wahby, Ioanna Tzialla, abhi shelat, Justin Thaler, and Michael Walfish. Doubly-efficient zkSNARKs without trusted setup. In *2018 IEEE Symposium on Security and Privacy*, pages 926–943. IEEE Computer Society Press, May 2018.

[313] Xiao Wang, Samuel Ranellucci, and Jonathan Katz. Authenticated garbling and efficient maliciously secure two-party computation. In Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu, editors, *ACM CCS 2017: 24th Conference on Computer and Communications Security*, pages 21–37. ACM Press, October / November 2017.

[314] Xiao Wang, Samuel Ranellucci, and Jonathan Katz. Authenticated garbling and efficient maliciously secure two party computation, 2017. https://github.com/emp-toolkit/emp-ag2pc, last updated.

[315] Julia Weber and Edwin W Kruisbergen. Criminal markets: the dark web, money laundering and counterstrategies-an overview of the 10th research conference on organized crime. *Trends in Organized Crime*, 22(3):346–356, 2019.

[316] Chenkai Weng, Kang Yang, Jonathan Katz, and Xiao Wang. Wolverine: Fast, scalable, and communication-efficient zero-knowledge proofs for boolean and arithmetic circuits. Cryptology ePrint Archive, Report 2020/925, 2020. https://eprint.iacr.org/2020/925.

[317] Andrew T. Winkler. Password protection and self-incrimination: Applying the Fifth Amendment privilege in the technological era. *Rutgers Computer & Technology Law Journal*, 39:194–215, 2013.

[318] Timothy A Wiseman. Encryption, forced decryption, and the constitution. *ISJLP*, 11:525, 2015.

[319] Charles V. Wright and Mayank Varia. Crypto crumple zones: Enabling limited access without mass surveillance. In *IEEE European Symposium on Security and Privacy, EuroS&P*, pages 288–306. IEEE, 2018.

[320] Charles V. Wright and Mayank Varia. A cryptographic airbag for metadata: Protecting business records against unlimited search and seizure. In *8th USENIX Workshop on Free and Open Communications on the Internet*. USENIX Association, 2018.

[321] Howard Wu, Wenting Zheng, Alessandro Chiesa, Raluca Ada Popa, and Ion Stoica. DIZK: A distributed zero knowledge proof system. In William Enck and Adrienne Porter Felt, editors, *USENIX Security 2018: 27th USENIX Security Symposium*, pages 675–692. USENIX Association, August 2018.

[322] Sen. Ron Wyden and Sen. Cory Booker. S.1108 - Algorithmic Accountability Act of 2019, 2019.

[323] Sen. Ron Wyden, Sen. Marco Rubio, and Sen. Mark R. Warner. S.681 - Student Right to Know Before You Go Act of 2019, 2019.

[324] Tiancheng Xie, Jiaheng Zhang, Yupeng Zhang, Charalampos Papamanthou, and Dawn Song. Libra: Succinct zero-knowledge proofs with optimal prover computation. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology – CRYPTO 2019, Part III*, volume 11694 of *Lecture Notes in Computer Science*, pages 733–764. Springer, Heidelberg, August 2019.

[325] Andrew Chi-Chih Yao. How to generate and exchange secrets (extended abstract). In *27th Annual Symposium on Foundations of Computer Science*, pages 162–167. IEEE Computer Society Press, October 1986.

[326] Samee Zahur, Mike Rosulek, and David Evans. Two halves make a whole - reducing data transfer in garbled circuits using half gates. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology – EUROCRYPT 2015, Part II*, volume 9057 of *Lecture Notes in Computer Science*, pages 220–250. Springer, Heidelberg, April 2015.

[327] Meike Zehlike, Philipp Hacker, and Emil Wiedemann. Matching code and law: achieving algorithmic fairness with optimal transport. *Data Mining and Knowledge Discovery*, 34(1):163–200, 2020.

[328] Jiaheng Zhang, Tiancheng Xie, Yupeng Zhang, and Dawn Song. Transparent polynomial delegation and its applications to zero knowledge proof. In *2020 IEEE Symposium on Security and Privacy*, pages 859–876. IEEE Computer Society Press, May 2020.

Curriculum Vitae

[redacted]

[redacted]

[redacted]

[redacted]

[redacted]

[redacted]