ASSESSING BEAR: TOOL USABILITY FOR WIRELESS CTF

A Thesis

presented to

the Faculty of California Polytechnic State University,

San Luis Obispo

In Partial Fulfillment

of the Requirements for the Degree

Master of Computer Science in Computer Science

by

Donald Sanchez

June 2021

COMMITTEE MEMBERSHIP

TITLE:	Assessing BEAR: Tool Usability for Wireless CTF
AUTHOR:	Donald Sanchez
DATE SUBMITTED:	June 2021
COMMITTEE CHAIR:	Bruce DeBruhl, Ph.D. Associate Professor of Computer Science
COMMITTEE MEMBER:	Dongfeng Fang, Ph.D. Assistant Professor of Computer Science
COMMITTEE MEMBER:	Lynne Slivovsky, Ph.D. Professor of Computer Engineering

ABSTRACT

Assessing BEAR: Tool Usability for Wireless CTF

Donald Sanchez

Capture the Flag (CTF) is a common and popular type of event in the cyber security space with audiences ranging from large security conference participants to even those in middle or high school. Event participants bring their favorite set of tools and any level of knowledge they have to compete against other teams in solving cyber security related challenges. These types of challenges can range anywhere from reverse engineering programs and hacking WiFi to utilizing interesting command line commands and messing with browser developer consoles. There are plenty of general CTF events that happen throughout every month, as well as plenty of resources for those. However, CTFs focused on wireless technologies are not as prevalent. Just this last year a Wireless themed CTF, named Wireless CTF (WCTF), became publicly available to participate in. With this CTF as the target, a tool set will be put together in this thesis to help introduce some of WCTF's topics: WiFi penetration testing, POCSAG radio signal decoding, and Morse Code Signal Decoding. Tools will be chosen based on the BEAR scoring rubric, created in this thesis, to assess a given tools usability, and chosen tools will be used against challenge topics found in WCTF to test the validity of the scoring rubric and evaluate changes in a participants knowledge of each topic.

ACKNOWLEDGMENTS

Thanks to:

- Andrew Guenther, for uploading this template
- Bruce DeBruhl and Dongfeng Fang, for their help and support through my thesis and experiment work

TABLE OF CONTENTS

LIST OF FIGURES

Chapter 1

INTRODUCTION

Capture the Flag (CTF) are a popular event in the cyber security community, with a few events happening every month in venues ranging from large security conferences to small online hosted events [3]. Each event will generally have some theme—jeopardy, attack and defense, etc—to indicate the types of challenges to expect as well as what types of tools one should bring [3]. This is generally the case for most of the publicly available CTFs; due to the large number of them that can be found online to use as references, newcomers can have an easier time finding tools they could try and use either through trial and error in attempting challenges or researching the challenges involved in some CTFs. However, this is not the case with CTFs focused on wireless technology. RF Sanctuary, formerly Wireless Village, is a group that hosts wireless themed CTF events for participants at security conferences such as Defcon, Bsides, and Schmoo con [32]. Due to these wireless CTFs being limited to security conference participants, newcomers wanting to come prepared will have a much harder time knowing what challenges and tools to expect to use without first being able to attend. However, just this past year, RF Sanctuary released their Wireless CTF (WCTF) as a public event. Using this new event as the basis, this thesis will try to alleviate these previous issues by creating a tool scoring methodology, and lab materials to enable newcomers to be able to find tools they would want to use and help introduce them to the topics that they may encounter when participating in this type of event.

1.1 Contributions

My contributions in this thesis take the form of the following:

- BEAR: a tool scoring rubric for evaluating a tools level of usability
- Labs: WEP WiFi penetration testing, POCSAG pager signal decoding, Morse Code signal decoding
  - Utilize mixture of cyber security education principles to help introduce topics from RF Sanctuaries WCTF event
  - Test the ability to assess tool usability with BEAR

1.2 Organization

Following this intro section will begin literature review and background relevant to this thesis that will include: previous and my tool scoring methods, principles of education for enhancing learning materials, and explanations for CTF and lab materials. The section to follow will cover usability scoring of tools and tool selection. Lastly, the remaining sections will elaborate on the experiments performed for this thesis and conclude with the results and future work to be done.

Chapter 2

LITERATURE REVIEW AND BACKGROUND

The following sections will begin by reviewing previous methods for scoring various types of cyber security tools. Based on these previous methods, we will begin to define and form the method for scoring tools based on their usability that will be used for this thesis. After these beginning sections, the remaining sections will cover education principles that are utilized to improve learning of the topics involved in this thesis' lab experiments, and is concluded with general background and information explaining the topics to be involved in the lab experiments.

2.1 Previous Methods of Tool Related Scoring

Since this research will revolve around reviewing and scoring different cyber security tools in order to gauge their usability, this section will review previous works where these types of tools are evaluated and scored in some manner.

In Frank van der Loo's research penetration testing tools for web applications were compared to one another [40]. The motivation for the authors research was due to the plethora of automated penetration testing tools available for web applications, each that produce results relatively quickly and easily; however, with so many tools to use, commercial and free, there was little testing done in regard to the quality of the tools themselves, which Frank van der Loo tries to address in their research [40]. Frank van der Loo chooses a handful of tools, based on availability, and takes note of the vulnerabilities each tool is reported to be able to detect; the author then puts each of the tools against vulnerable web applications to verify if the tools are able to perform effectively and return desired and expected results [40].

The next work will involve the work done by Gabriela et al. for comparing risk analysis tools related to cyber security [31]. The motivation for research done by Gabriela et al. revolves around evaluating the most relevant and popular tools that are available for decision making and risk assessment in the cyber security space [31]. Each tool that was reviewed used various strategies or techniques in order to prioritize risks; most of these involve the use of Common Vulnerability Scoring System (CVSS)—a metric for scoring the risk a vulnerability may pose to a business as well as its characteristics and severity [6]—or Common Vulnerabilities and Exposures (CVE)—a unique identifier to easily find vulnerabilities and patch information [4]. This works comparisons come down to looking at the overall effectiveness of each tool and their used metrics for assessing potential vulnerabilities [31].

Next, work done by Adam Hahn et al. shows another type of evaluation of cyber security assessment tools [24]. The North American Electric Council (NERC) created their own Critical Infrastructure Protection (CIP) requirements that involves all cyber assets that are supporting bulk power systems; the research of Adam Hahn et al. explores the possibility of whether or not the methodologies and tools that are commonly used for traditional information technology (IT) systems are able to sufficiently fulfill cyber security assessment needs for power systems [24]. Each tools capabilities are tested against several different NERC CIP compliance categories including system configuration review, network traffic review, network rule set review, network discovery, port and protocol identification, and vulnerability scanning [24]. For each assessment, a tool is given a mark for whether or not it is able to provide full, partial or incomplete coverage for each set of compliance points made by NERC

CIP [24]. Like the previous work, Adam Hahn et al. shows more evaluations that shed light on the overall effectiveness of each tool.

For the next work, Chiem's research work in [15] focuses on rating effectiveness of prevalent penetration testing tools that are used in the profession. Chiem states that, similar for other professions, the penetration testing space receive efficient aid from many automated tools, but, due too the large number of tools that are available for use, penetration testers can experience difficulty finding tools that are the most suitable for a given task [15]. Chiem breaks the tools into separate groups, service fingerprinting and vulnerability scanning, for performance evaluations [15]. The performance metrics used for Chiem's testing involve response times, the number of services identified, as well as the number of vulnerabilities able to be detected and are all recorded and organized to compare in a quantitative graph [15]. Once again like the other past works, the tools being tested all scored on a strictly performance, or effectiveness, based assessment.

For the final work that will be looked at for tool assessments will be Mandar's research in [33]. The motivation behind Mandar's work stems from the growing need for security audits to be performed due to the ever growing threat of cyber attacks [33]. Important aspects involved in protecting and securing important systems involves penetration testing of the systems web applications and the network; while the need for penetration testing is growing the need for benchmarks or standardization of the processes that tools use continues to grow as well [33]. Mandar takes four vulnerability scanners and runs them in a few different types of environments to evaluate them: black box, no system information is known; grey box, only system infrastructure and web applications are known; white box, full system information is known [33]. The tools are evaluated based on many categories including some of the following: The ability to handle each of black, grey, and white box environments, having active and or passive crawling, coverage the crawler is able to provide, scanning speed, vulnerability detection rate, false positive reported, and several more [33]. Mandar's work as a whole remains in the same frame as the previous works talked about reporting results for the performance and effectiveness of the researched tools.

To the best of my knowledge most if not all of the previous research work that I am able to find follows the same performance evaluation of rating overall effectiveness a tool is able to achieve by meeting expected goals, number of vulnerabilities found, accuracy of risk assessment, etc. This type of evaluation, however, is not what is being looked for in this research. Due to this, I instead attempt to create my own evaluation framework for tool usability which will be expanded upon in the next section.

2.2 Tool Scoring Categories

Usability, or what determines how easily someone can learn and understand how to use a particular system, software, tool, etc, can vary from person to person and come in different forms. As a general idea of what good usability entails, Lorrie Cranor's, a security and privacy teacher for Carnegie Mellon University, definition of usability will be used as part of this papers use of usability. Lorrie Cranor defines what is usable as being the following: Intuitive / obvious, efficient, learnable, memorable, few errors, not annoying / enjoyable, status transparent, and meets users needs (utility) [19]. Lorrie Cranor further focuses these ideas for security and privacy by defining what is usable security and privacy as — security and privacy software is usable if people who are expected to use it: [19]

1. are reliably made aware of the security / privacy tasks they need to perform;

2. are able to figure out how to successfully perform those tasks;
3. don't make dangerous errors; and
4. are sufficiently comfortable with the interface to continue using it.

Using Lorrie Cranor's usability definitions several common ideas, or categories, can be interpreted and extracted to describe tools for the purposes of this paper. Lorrie Cranor's points about needing to be intuitive or obvious, learnable and memorable, and being reliably made aware of tasks or functions the user can use or need to perform, as well as how to perform them successfully, can all fall under a broader category that describes a tools "ease of use". Another category that can be found in these definitions is a tools "reliability" which involves the number of errors a tool might run into, and having a transparent status for the current state the tool is in (i.e. tool updated to ensure stable newest version). One final category from Lorrie Cranor's work that can be interpreted is a tools "breadth"; this category involves a tools ability to meet the users needs where the more needs a tools is able to fulfill for the user the higher the breadth that tool will have.

For the last category that will be used for scoring tools the article by Jason R. C. Nurse et al.[30] will be used to reference "accessability". In [30] general recommendations that have been proposed in order to improve the usability of cyber security interfaces and systems were compiled together, allowing Jason R. C. Nurse et al. to put together a list of guidelines. The guideline that will pertain to this paper is "make security functionality visible and accessible"[30]. As the article describes, this guideline is fulfilled when there is visibility and ease of access, and without these a users task becomes more difficult[30]. This ultimately results in a reduced level of system usability [30]. In this same way, the idea of "accessability" leading to improved usability will be applied as a category for scoring tools in this paper.

To sum these categories up that will be used to assess the usability of a tool, they are as follows:

- Breadth: ability to meet users needs where the more needs that are able to be fulfilled the higher the breadth
- Ease of Use: intuitive or obvious, learnable and memorable, and being reliably made aware of tasks or functions that a user can use or need to perform, and how to perform them successfully
- Accessibility: ensure presence of visibility and ease of access
- Reliability: number of errors a tool could encounter, and having a transparent status for the current state that the tool is in

2.3 Education Principles

Now that we have a basis established for scoring tools, some structure will needed for conveying educational information about the tools and experiment exercises. In order to do this, a set of instructional design principles will be established. These instructional design principles will be drawn from works by Zhang et al. [43] [44] [42] And Kumaraguru et al. [26] which will include the following principles:

- Segmenting: Segmenting a lesson involves breaking a larger complex lesson into smaller pieces and shown one at a time rather than as one large continuous unit, which helps the learning process for people [43][42]
- Signaling: Learning becomes more efficient with the use of cues that highlight material organization; these cues direct the users attention to key messages

within a lesson to aid the process of information discovery and understanding [43]

- Personalizing: Through the use of a conversational style, as opposed to a formal style, learning capabilities can be enhanced; due to being presented in a way to make the learner feel as though they are in a conversation, the learner will make more of an effort to understand the instructional materials being presented to them. [42][26] This conversational style can be introduced through the use of words in the instructional text like the following: "I", "we", "me", "my", "you", and "your" [26]

- Multimedia: the principle of multimedia states that through the use of both images and text together learning is much more conducive as opposed to having each element isolated from one another. [42][43] According to the dual coding theory, it is suggested that graphics, text and audio are coded into memory separately [18]. Studies are able to show that a combination of text and images allows for better comprehension and an increase in long-term memory retention.[28]

- Lean-by-doing: Just as Cal Poly uses the idea of learn by doing to promote increased learning, [26], and [43] also utilize learn by doing. As suggested in [10] those given the option to put what they have just learned into practice had improvements in their knowledge transfer and had better performance compared to those who were not given this option.

2.4 Cyber Security CTF

CTF is a popular type of competition in the area of computer security, not the physical sport variant, where competitors compete against one another individually or in teams to score points; points can be scored in a variety of ways depending on the style being used [25]. These different styles as discussed in [25] include the following:

- Quiz: Typically question and answer style questions related to security topics
- Jeopardy: Broken into several categories, each category can contain several challenges related to a particular security area of focus including areas such as web exploitation, cryptography, reverse engineering, binary exploitation, forensics, etc [5].
- Attack-Defense: involves teams trying to both attack a vulnerable system as well as teams attempting to defend that vulnerable system from being hacked.
- Mixture: Can have a combination of attack and defense and jeopardy style challenges.
- King of the Hill: Teams compete in order to hack into some vulnerable system and keep it under their control, by defending from other teams hacking attempts, for as long as they can.

Out of the available styles of CTF, Jeopardy tends to be the most used style as can be seen according to CTF time: a website that keeps track of current and upcoming CTF events and information [3]. As such, jeopardy style CTF challenges will be the focused style for this papers created CTF learning materials.

These CTF type challenges will serve as a great educational tool for this experiment as they follow closely as a learn by doing type of activity. As talked about in Erik Trickel et al. [39], these live types of cyber security exercises are a benefit to the security community in a number of ways.

- CTF exercises allow for participants to take any theories, concepts, or other knowledge about the challenge topic and practice those techniques.
- With events being finite in terms of taking place during a limited window of time as well as a competitive environment between participants, participants learning experience will be improved.
- Cyber security events in a live setting allow for deeper engagement and increases academic learning time, resulting in learning at a faster rate, and more easily achieve mastery with concepts involved.

2.5 WiFi: WEP Penetration Testing

2.5.1 What is WEP

WEP, or Wired Equivalent Privacy, is an early security architecture of the IEEE 802.11 wireless LAN standard; WEP's purpose was to try and make wireless LANs at least as secure as a wired LAN connection [14]. However, WPA fell short of this goal and was found to be inadequate at providing protection for these connections; due to this fact new standards were created to replace WEP, but WEP still remains today as a usable protocol in order to support backwards compatibility [14].

2.5.2 How WEP Works

For a mobile station (STA), or user device, to connect to a network, the STA must authenticate with the networks access point (AP), or router in most use cases [14]. This authentication takes place in order for encrypted communications to occur between a STA and AP and prevent bad actors from using the network without authenticating;

the process for authenticating a STA involves a challenge and response protocol that is made through the exchange of four messages [14]. As described in [7], the four messages involved in the authentication exchange are shown in figure 2.1 where the STA first will send a request to begin authentication to the AP; the AP will then respond with a challenge, or a random set of 128 bytes [7]. The STA will then encrypt the challenge text with the secret key provided by the user trying to authenticate and send the encrypted challenge back to the AP; the AP will then try to decrypt the received encrypted challenge text [7]. If the AP is able to successfully decrypt the challenge text then the AP knows that the STA has the correct secret key and will send a success response completing the authentication of the STA to the AP; if the decryption fails, the AP will send a failure response and the STA will not be authenticated [7].

After successfully authenticating with an AP, the STA and AP begin communicating using encrypted messages; the encryption used for WEP is the RC4 stream cipher [14]. A stream cipher uses some seed value or key in order to produce a long sequence of pseudo-random bytes; these bytes are then XORed to the message meant to be encrypted byte by byte resulting in a RC4 encrypted message [14]. In WEP the secret key along with an initialization vector (IV) are used as the seed to RC4, as shown in figure 2.2; the presence of the IV as a part of the seed is used to prevent RC4 from always generating the same stream of pseudo random sequences since the secret will remain the same throughout the STA and AP communications [14]. The IVs that are appended to the secret key are 24 bits long while the secret is usually 104 bits [14].

2.5.3 WEP Weaknesses

WEP has been known to be weak WiFi protocol from some time now and newer and stronger standards have been made. Out of the several weaknesses that WEP

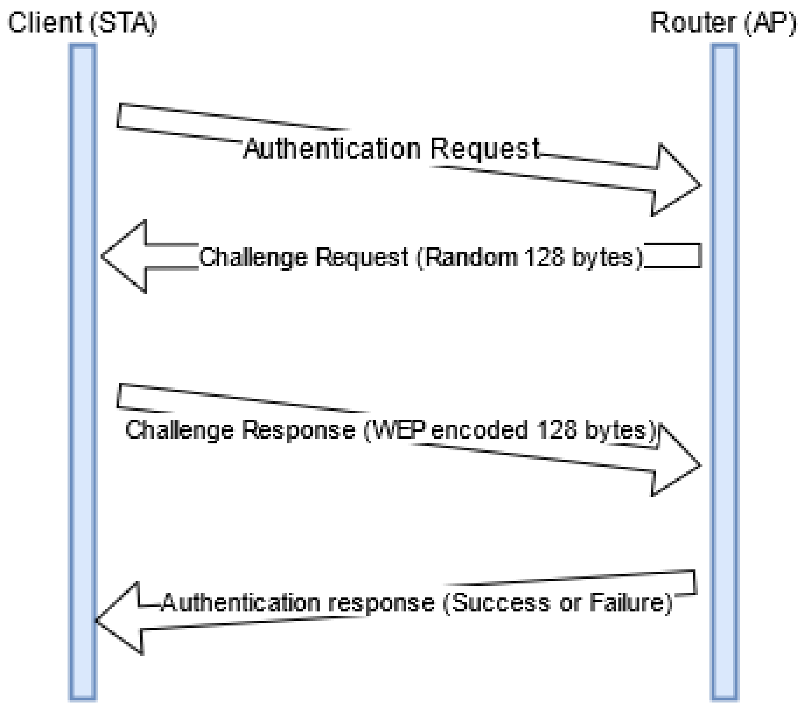
Client (STA)
Router (AP)
Authentication Request
Challenge Request (Random 128 bytes)
Challenge Response (WEP encoded 128 bytes)
Authentication response (Success or Failure)

Figure 2.1: Challenge and response shared key authentication for WEP [7]

has there are a few that are of interest and will pertain to the experiments later in this paper. One of these weaknesses involves WEP's authentication. In WEP's authentication scheme the STA is only authenticated when it attempts to connect to the network, and once that authentication has completed and the STA becomes associated with the AP, anyone has the ability to send messages using the name of that STA by spoofing or mimicking its MAC address [14]. Another weakness revolves around the IV's used for the RC4 encryption. The IVs that are used are only 24 bits long; this means that there are only less than 17 million unique possible IVs [14]. Typical WiFi devices are capable of transmitting about 500 full length frames a second; this means that every IV will have been used after only 7 hours, or 7/n hours per n devices connected to the network [14]. When repeated IVs are used, messages will end up with repeated pseudo-random sequences used during message encryption. The last weakness to bring up is that RC4 is used incorrectly in WEP

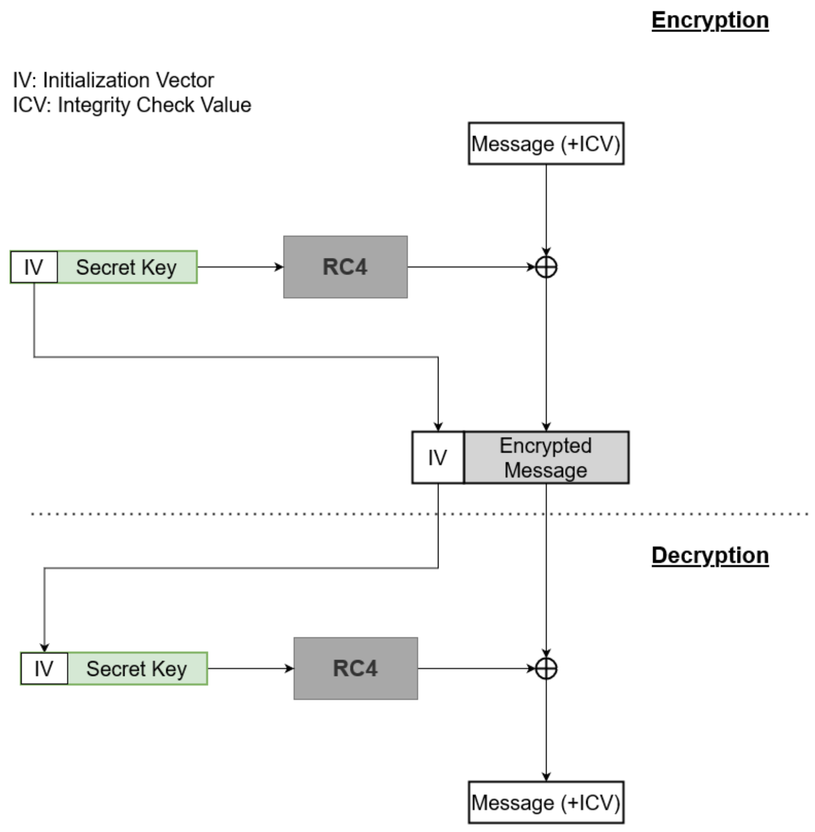

Figure 2.2: WEP Encryption and Decryption diagram [14]

implementations; weak keys exist that produce non-random looking sequences in the first few bytes of the RC4 output [14]. For RC4 it recommended to discard the first 256 bytes of its output to prevent this weak key issue, but WEP does not implement this fix [14]. With these weaknesses in mind attackers can use automated tools for launching attacks such as those talked about in Martin Beck et al. [38] in order to break the 104 bit secret key by eavesdropping messages between the STA and AP [14].

2.6 Radios and Signal Transmission

2.6.1 How Radios Transmit Signals

Radio receivers and transmitters are a widely used technology with many purposes including listening to music or broadcasters in the car, disaster and emergency communications, restaurant pager systems, etc. Radios are able to transmit these types of data through the use of modulation and demodulation for the desired data or information to be transmitted and understood by the receiver that gets the signal [13]. Radios utilize a carrier wave, a static frequency sinusoidal wave, that is modulated, or modified, by the desired data to change one or more aspects of the carrier wave: amplitude, frequency, or phase [13]. The resulting wave will contain the information for both the carrier wave and the data intended to be sent and is transmitted [13]. Some of the common and basic examples of modulation can be seen with AM and FM radio, amplitude modulation, and frequency modulation respectively [13].

For AM and FM radio signals, once the carrier waves frequency or amplitude becomes modulated as a function of the audio content from a radio station, the receiver of the radio signal must demodulate the signal in order to listen to the audio that was intended to be sent [13]. In the case of AM demodulation, one option that be used involves multiplying the modulated wave with the carrier wave and the use of a low pass filter in order to correct the shifting that occurs due to the multiplication resulting in original data [16]. For FM demodulation, one method to retrieve the original data involves using a high pass filter in order to change the FM signal into an AM signal at which point AM demodulation techniques can be used [17].

For lab experiments performed later in this paper, one of the signal types that will be looked at is POCSAG, a radio pager communication protocol; POCSAG uses a type of digital frequency modulation called Frequency Shift Keying (FSK) where the 0's and 1's of a piece of data are represented as two set frequencies that the signal will change between instantaneously, called coherent FSK, or with discontinuities or gaps, noncoherent FSK [13][41]. The other type of signal is Morse Code, sometimes referred to as Continuous Wave (CW) [29]. Morse Code transmission do not use a form of modulation in order to transmit data—instead the carrier wave is turned on and off to make long and short interruptions [29]. Different combinations of these interruptions form letters which are used to communicate using Morse Code [29].

2.6.2 What is a Fox Hunt

Contrary to name of this event, a fox hunt in the amateur radio space hunts or attempts to locate a hidden radio transmitter, called a fox for this particular event [36]. Fox hunting is a popular event among amateur radio hobbyists and numerous competitions are organized for all ages around the world [36]. The foxes will usually operate in either the 2m or 80m radio bands, 144MHz-148MHz and 3.5MHz-4MHz respectively; these frequencies are chosen as they are more universally available to all licensed amateur radio users [36]. Foxes are placed in locations within an area, specified for the fox hunt, that is unknown to the participants, and participants are given the frequencies that the foxes will be operating, or transmitting, on [36].

Different styles of fox hunts can also be seen where, instead of just a stationary or fixed location fox, mobile or pedestrian foxes can be utilized giving extra factors to consider when trying to locate a given fox [36]. During the event the hidden foxes will be transmitting a signal pattern, usually a series of tones, followed by an identification id, or call sign of the licensed radio operator, in Morse code and finally followed by a period of silence; this transmission pattern will happen on a repeated cycle throughout the event [36]. Event participants will make use of antennas in order to receive the signals being transmitted by the foxes; the usual type of antenna used for this are directional antennas which allow for receiving signals strongly in one particular direction which reduces potential interference from other incoming signals not in the direction of the antenna [36].

Chapter 3

TOOL ASSESSMENT

3.1 Scoring Methodology

As was brought up earlier and in [33],[40], [31], [24], and [15], to the best of my knowledge there have only been assessments done to evaluate overall effectiveness or performance of cyber security tools. Due to the lack of usability tool assessments I have created my own scoring rubric for usability. I wanted this rubric to follow similar patterns to other already used cyber security risk assessment models; so, I formed my rubric by taking inspiration from STRIDE (Spoofing, tampering, repudiation, information disclosure, denial of service, and elevation of privilege) and LINDDUN (linkability, identifiability, nonrepudiation, detectability, disclosure of information, unawareness, and noncompliance), two models for categorizing threats, as well as DREAD (Damage potential, reproducibility, exploitability, affected users, and discoverability) for scoring a threats risk based on the five categories it is made of [34][35]. With these other model ideas in mind, I take my tool scoring categories "breadth", "ease of use", "accessibility", and "reliability", made earlier based on [19] and [30], to form the mnemonic name for my scoring rubric: BEAR.

Much like how each letter or category from DREAD receives a score, each category of BEAR will also receive a score to represent its contribution to usability. Each category in BEAR will be scored out of ten points split between a set of one or more subcategories. "Breadth" will cover one subcategory called "use coverage" which will account for all ten points, and it will receive two points for every use the tool covers including things such as the following: WEP, WPA, WPA2, password cracking, radio

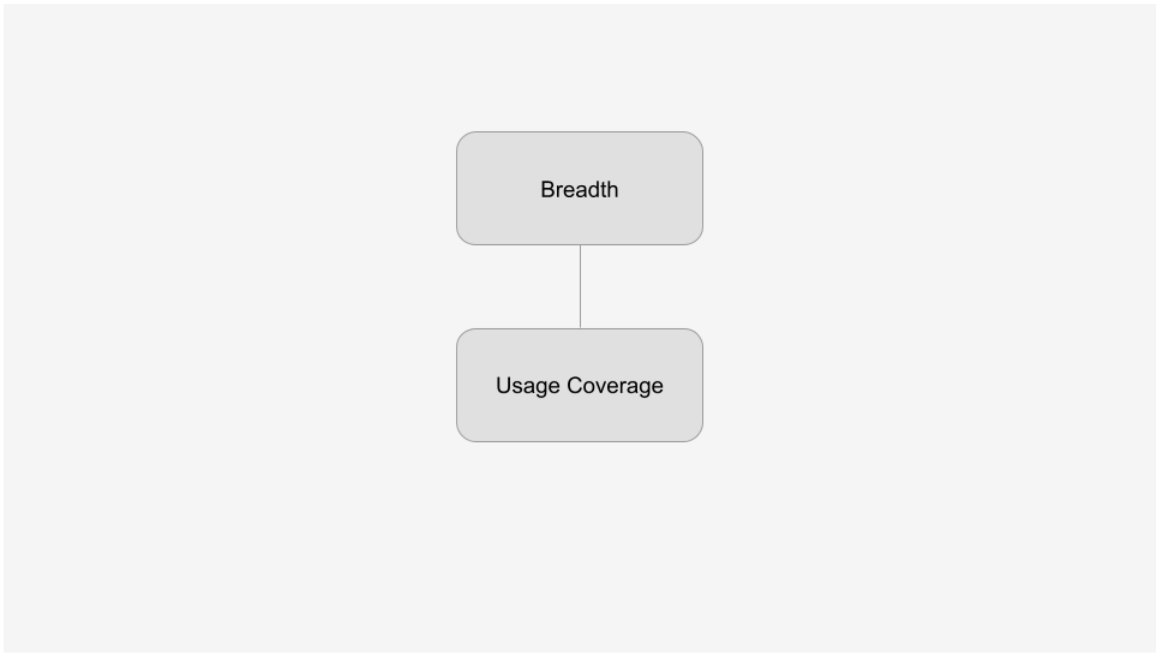

Figure 3.1: Main category Breadth and its subcategory breakdown

decoding, bluetooth, etc. The next category “Ease of use” will cover three subcategories: “intuitive”, “good documentation”, and “available tutorials”. How intuitive a tool is will gauge how easy it is for a user to pick up and work with will cover three points. Whether or not there is available and good documentation for working with the tool and providing information of available functionality will cover four points. The remaining three points will be covered by whether or not there are a number of tutorials available to show proper usage of tool functionality. The third category, “Accessibility”, will be made up of four subcategories: “payment requirement”, “available operating systems”, “easy to install”, and “high spec computer requirement” . First, “payment requirement” will begin with one point and will lose that point if the tool requires payment to use or half a point if there is a limited or trial version available. For “available operating systems”, with a total of three points, one point will be given for each operating system the tool is available to be used on. For the next subcategory, “easy to install”, up to a total of three points will be given depending on if the installation process is easy and straightforward for a user. The

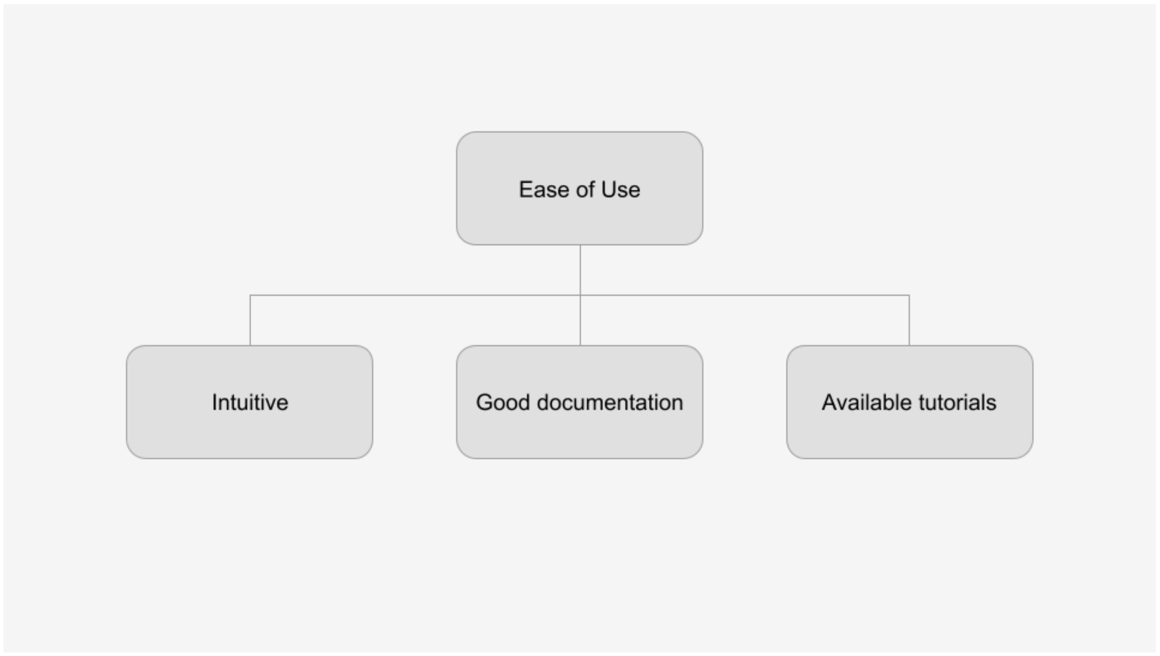

Figure 3.2: Main category Ease of Use and its subcategory breakdown

final subcategory, “high spec computer requirement”, will cover the remaining three points for “Accessibility”; these last three points will be given based on whether or not the tool requires a computer with higher than average components in order to function properly. The final category, “reliability”, will be split into two subcategories: “kept up to date by author”, and “reliant on many dependencies”. The subcategory of “kept up to date by author” will cover and start with a total of five points; one point will be taken away for every two years the tool hasn’t been given an update. The other subcategory “reliant on many dependencies” makes up the remaining five points and will be given based on the number of dependencies a program is reliant upon—the more dependencies the tool has the more likely the chance for an error to occur due to their updates, or the main tools updates.

In summary the categories, subcategories and their distribution of points between them are as follows:

- Breadth

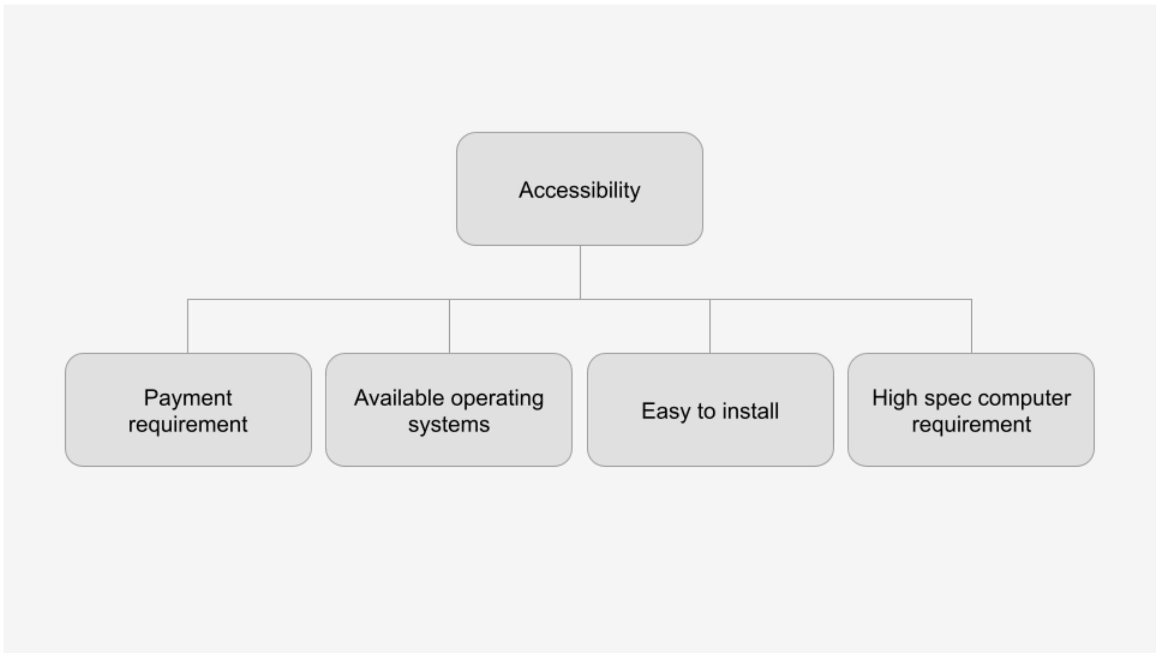

Figure 3.3: Main category Accessibility and its subcategory breakdown

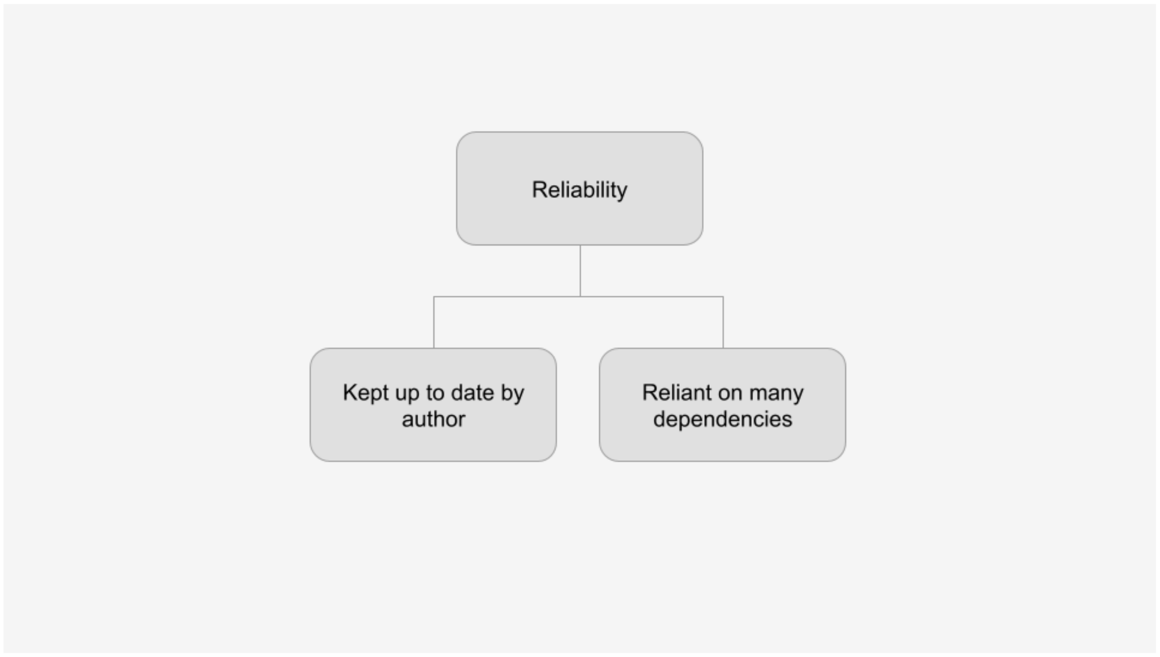

Figure 3.4: Main category Reliability and its subcategory breakdown

  - Usage coverage: 10 points

- Ease of Use
  - Intuitive: 3 points
  - Good documentation: 4 points
  - Available tutorials: 3 points
- Accessibility
  - Payment requirement: 1 points
  - Available operating systems: 3 points
  - Easy to install: 3 points
  - High spec computer requirement: 3 points
- Reliability
  - Kept up to date by author: 5 points
  - Reliant on many dependencies: 5 points

3.2 Tool Scoring and Selection

With the formation of this scoring rubric I have taken a set of a little more than 70 tools—covering many different areas of use such as WiFi, password cracking, network monitoring, radio decoding, Bluetooth, web, etc—and have given all of them scores using this rubric as can bee seen in the graphs in Appendix A. Each tool in this list has been chosen based on being either directly applicable to the challenges found in RF Sanctuaries WCTF, or able to provide assistance to the user in some form against the challenges. After scoring this set of tools, tools can be searched and sorted based on individual categories or a mixture of all of them. Depending on what is needed or desired, tools that have multiple use cases, are easy to use and work with, are easily accessible, are reliable and do what they are supposed to, or a combination of these, can easily be searched for, and will be used to select a subset of tools to tackle WCTF topics used in this thesis.

Based on a combination of these scores and the specific categories that will be looked at from RF Sanctuaries WCTF I have chosen a set of 4 higher scoring software tools that will be introduced, along with their respective CTF category, to experiment participants. These tools include the following: Aircrack-ng, SDR#, PDW, Fldigi. Aircrack-ng is a tool suite with WiFi network security in mind, and it will be the main tool for exploiting the WEP WiFi protocol in this thesis' experiment [8]. Aircrack-ng offers not only tools for attacking but for monitoring, and even cracking WEP, WPA, and WPA2 protocols [8]. SDR# is a program for utilizing software defined radio; with it—along with an appropriate USB interface, an rtl-sdr dongle in the case of this experiment—the user will be able to pickup and listen to a wide range of different radio signals [9]. This piece of software will be used by participants to listen for a specific type of radio signal and, with the help of the next couple of program, decode them. One feature SDR# has that is very useful, for the later experiments, is that a sample signal can be recorded and replayed on an automatic loop making signal analysis much easier. PDW is a small program used for monitoring and decoding POCSAG and FLEX radio signals, and it is popular among radio enthusiast and professionals [21]. Using PDW, participants will be able to see exactly what messages are being sent using POCSAG. For the last tool, Fldigi is a program that supports decoding of most digital modes used on amateur radio bands [20]. Fldigi will mainly be used in this thesis' experiment for taking the Morse Code signals found by participants and decoding them into a readable format.

Chapter 4

EXPERIMENT METHODOLOGY AND LABS

4.1 Approach

The purpose of the labs that were made for this thesis were to provide environments for the experiment participants to be introduced to topics that a person would most likely see in events like RF Hackers Sanctuary's Wireless CTF (WCTF) [23], as well as have participants use my tool scoring rubric to give their own scores on each tools overall usability. Each participant will first complete an entry survey that will gather information about any initial knowledge, familiarity, and skill level each participant has with any of the topics of software defined radio (SDR), WiFi penetration testing, radio signal types, WiFi security protocols, or any tools related to these topics. After completion of the labs, an exit survey will then gauge any changes to the experience and knowledge the participants feel they have had and will ask for them to gives scores for "breadth", "ease of use", "accessibility", and "reliability" for each tool used throughout the lab experiment.

To try and improve the learning experience during the lab, I utilize a mixture of the education principles, mentioned earlier in the paper, segmenting, signaling, personalizing, multimedia, and learn by doing. Segmenting is introduced into the labs by having each lab split into multiple sections. The WiFi and SDR labs begins with setup and information gathering tasks and lead into password cracking for the WEP lab, and signal finding and signal decoding for the SDR lab; while the fox hunt lab is broken down into 3 overall tasks to complete [42]. For the WEP and SDR labs, signaling is used draw and direct the attention of the participants by making the relevant and important pieces of information in bold to stand out [43]. Personalizing is utilized through several sections of each lab by varied use of “I”, “we”, “me”, “my”, “you”, and “your” for a more conversational style of speech [26][42]. For multimedia, video and images are given along with textual instructions to aid the fox hunt lab participants through the antenna construction process, and the SDR labs will have participants looking at images and audio samples to enhance learning retention through dual coding theory [18][42]. Lastly learn by doing is incorporated into each lab as every participant will be learning about a topic or tool and immediately putting that knowledge to use by working through each task [26].

4.2 Labs

These next sections will talk about some of the details of labs used in this thesis and the tasks performed by each of the participants. However, the full documents for each of these labs can be found in Appendix B.

4.2.1 Radio Direction Finding

Radio direction finding can be seen as a similar idea to human audio direction finding where, based on some source of sound, we are able to locate the direction the source of sound is originating from [12]. Radio direction finding can be seen in the same manner, but for the purpose of locating the origin of radio signals. This has several applications including animal tracking, search and rescue, locating transmitters and signal jamming devices in military environments, and monitoring and locating of illicit or interfering transmitters [12]. For introducing the topic of Radio Direction Finding, as well as the basic equipment used in fox hunts, I created and hosted a fox hunt lab in collaboration with a Cal Poly professor and their Wireless Security class. Radio fox hunts, or just fox hunts—not to be mistaken with real fox hunting—are an event where participants utilize radio direction finding techniques in order to locate hidden radio transmitters called foxes [36]. This lab took place on campus grounds over the course of two days. Utilizing the education principles of segmenting, personalizing, and learn by doing, students are tasked with making their own directional antenna from scratch, as well as finding two radio foxes, hidden on campus, by using the previously constructed antennas. For building the antenna on day one, students follow a guide written by Michael Martens in [27]; an easy to follow guide that provides step by step instructions along with images and even a video guide for helping the students through the build process. Once the antennas have been built, students can test whether or not they are working by either connecting their antenna to a handheld radio or after setting up radio software with an rtl-sdr dongle, a USB device used to connect an antenna to a computer for radio software, and listening to the test fox in the lab. After task one was finished task two involved set up instructions for SDR software for linux, windows, and mac operating systems; this software will be used together with the constructed antenna in order to listen to and find the foxes that will be hidden in the final task. For the second day students will work on the third and final task, finding the hidden foxes. Similar to general fox hunts described in [36], students are given the operating frequency ranges that each of the two foxes will be operating in and are shown a map of the area on campus where the foxes can be hidden.

For the final task, students will only be using basic techniques in order to find the foxes; students will only be using their directional antenna and monitoring the signal strength in the direction of their antenna to get a general idea of where each fox could be located. When the students start to get closer to a particular fox, the signal will become stronger and be harder to pinpoint its origin. In order to overcome this difficulty, students will have to attenuate, or reduce the power of, the incoming signal.

The required attenuation for this lab will be achieved through the use of the harmonics given off by the foxes. The harmonics are integer multiples of the fundamental or reference frequency, the foxes operating frequency in this case [37]. Nearly all signals contain energy at harmonic frequencies but are usually less than the fundamental frequency [37]. By using the harmonics, students will be able to attenuate the signal and be able to more accurately locate the foxes when in close proximity. An example of using harmonics can be seen in figure 4.1

4.2.2 Student Experiment

Separate from the previous lab, the following three labs were performed as a direct experiment with individual students. Participating students were supplied the required hardware equipment and instructions for setting up hardware and necessary software environments to perform the required lab tasks. Each of these experiments will be conducted by the participants in a location of their choosing once they have received their equipment. Participants will begin with a short entry survey to gauge any initial experience they might have with any of the tools of interest, and after completion of the experiment an exit survey is used to gauge any change in learning over the course of the experiment as well as allow the participants to evaluate and score the tools for their overall usability using the BEAR score creating in this thesis.

4.2.2.1 WiFi Penetration Testing

The WiFi lab will serve to introduce the topic of penetration testing and some tools that can be used for the task. Using the education principles of segmenting, signaling, personalizing, and learn by doing to aid in learning, students will be tasked with gathering information about a wireless router, and, using the discovered information,

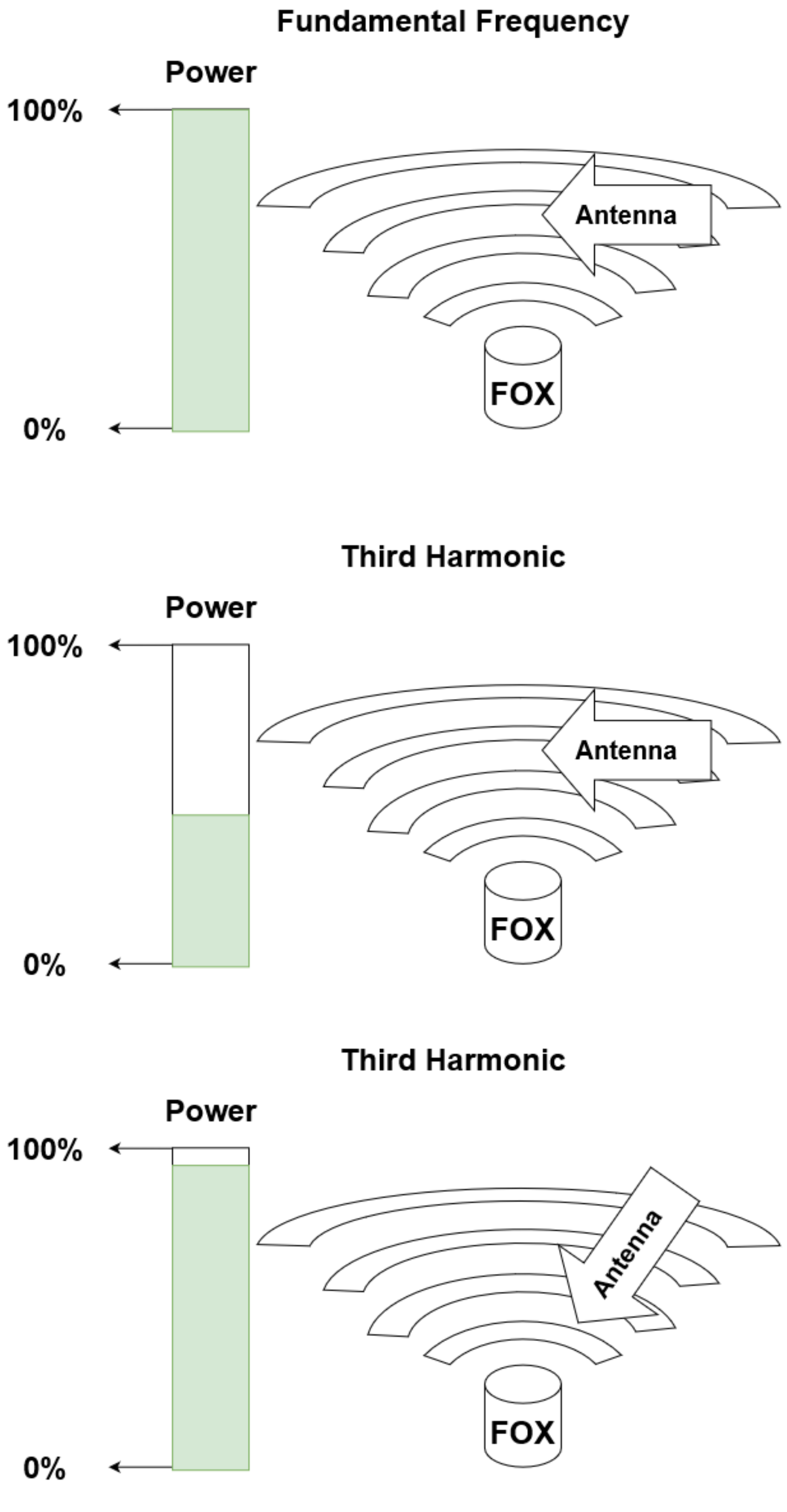

Figure 4.1: Example of using harmonics to more accurately locate origin of signal when in close proximity to a radio fox

learn how to use the provided tools to exploit the vulnerable router to gain its password. Tools provided and used for this lab include Aircrack-ng a tool for penetration testing wireless networks, a WEP protected WiFi router, and a USB WiFi Adapter that allows for packet injection.

In the information gathering stage, participants will start learning about the provided tools by gathering necessary pieces of information to exploit the router using a weak security protocol. Using Aircrack-ng, participants will begin by setting the provided USB WiFi adapter into monitor mode, allowing them to pick up information about nearby routers. Information that will be noted will include the BSSID, or broadcast id, and the channel the router is operating on. With these pieces of information the participants can use Aircrack-ng to specifically eavesdrop on and collect traffic going to the target router. The objective of collecting network traffic to this router is to obtain a large number of IV's for Aircrack-ng to later automate the process of revealing the password of the target router. However, passively waiting for traffic to be generated can take a long amount of time if the router is not very active. Instead this is where the participants will require the WiFi adapter that is capable of packet injection. Using packet injection the participants will be able to forge packets, by spoofing the MAC address of a device that is communicating with the router, and send them to the router. As mentioned earlier in this thesis, this is possible due to WEP only authenticating devices when they connect. So by spoofing the MAC address of an already connected device packets can be forged and sent as the connected device. This will allow for traffic to be generated more quickly and speed up the process of collecting IVs.

This process can take a few hours or more but once enough IVs have been collected, roughly 100,000-150,000 or more, participants can attempt to retrieve the routers key. By giving the captured router traffic to Aircrack-ng an automated attack can be started to attempt retrieve the key or password. If Aircrack-ng fails to return a key then participants can wait for more IVs to be collected and attempt the process again. Once enough IVs have been collected Aircrack-ng will be able to return the current key being used by the router to encrypt traffic and allow for connections.

4.2.2.2 POCSAG Pager Decoding

The POCSAG Decoding lab will serve to introduce the topic of POCSAG pager digital radio signals, how to identify them, and tools that can be used to decode them. Using the education principles of segmenting, signaling, personalizing, multimedia, and learn by doing to help improve learning, participants will begin by gathering information about POCSAG signals as well as learning how to use the provided tools for discovering these types of signals and decoding what messages they are being used to send. The tools for this lab will include SDR# and an rtl-sdr, an sdr program and USB dongle for listening to radio signals [11], PDW for decoding POCSAG, and VB-Audio Cable just for routing audio from one program to the other.

For the information gathering task, participants will lookup what POCSAG signals look like on waterfall plots or frequency-to-amplitude graphs, what POCSAG signals sound like when listening to them, and possible frequency ranges they can be found in. As SDR# will show waterfall plots, an frequency-to-amplitude graph, and play the audio of signals the user is listening to participants will have many different way to confirm and reinforce what type of signal they are currently looking at. Now armed with these pieces of information, participants should have an easier time in the next task where they will begin trying to find a POCSAG signal.

For the signal finding task, participants will have two separate environments to try and find the signal they are looking for. The first environment will be controlled sample that already contains a POCSAG signal transmission, allowing the participants to more easily begin working with this type of signal. Later on after working with the controlled sample, participants can begin trying to see if they can find any POCSAG signals in their local area. This task will allow the participants to work with and become more familiar with what they can do inside of SDR# by learning how to move and adjust the tuning bar to focus on the desired signal, how to use the squelch setting to remove unwanted noise, and which demodulation mode to use.

In the final task, participants will now get more familiar with PDW which will be used to decode the POCSAG signal being listened to by SDR#. PDW settings will be updated by the participants in order to enable POCSAG decoding, and, once properly configured, VB-Audio Cable is used to send the SDR# audio to PDW. Once here, if everything is set up and configured properly, PDW will begin returning the decoded POCSAG messages to the participants. If not, minor adjusts will need to be made in SDR# until a proper decode is retrieved.

4.2.2.3 CW Morse Code

The Morse Code lab will introduce the topic of Morse Code radio signals. Similar to the POCSAG lab, participants will be learning how to identify and find these signals, and about tools used to decode them. Using the education principles of segmenting, signaling, personalizing, multimedia, and learn by doing to aid in learning, participants will be gathering information on Morse Code signals, and learning to use the provided tools for signal discovery and decoding. This lab will also include SDR# and rtl-sdr for finding and listening to the desired signal. VB-Audio Cable for audio routing, and Fldigi for decoding the found Morse Code signals.

In the information gathering task, similar to the POCSAG lab, participants will be looking up visual, audio, and frequency range information about Morse Code signals in order to better understand how to find and identify these signals.

In the signal finding task—now that the participants will more or less know what to look for—participants will again have two separate environments one controlled sample that already contains an example of Morse Code Signals, and a real life environment where participants can try to locate Morse Code in their local area. Participants will once again be working with SDR# in order to locate and listen to Morse Code signals. After locating a Morse Code signal they would like to decode participants will begin working with a new tool for decoding in the next task.

For the final task, participants will begin to use Fldigi. Participants will now begin sending the audio found in the previous task to Fldigi using VB-Audio Cable. Now participants will begin getting familiar with and learning how to configure and use Fldigi to focus on the correct pieces of audio that show up in its own waterfall plot. If successfully set up, and configured Fldigi will begin interpreting the Morse Code audio and printing the resulting characters to the screen. Depending on how clear the signal is and if there is a sufficient amount of volume full words and sentences should be retrieved. Otherwise, settings can be adjusted or a new signal can be looked at to find better results.

Chapter 5

RESULTS

5.1 Experiment Results

Due to the less than favorable circumstances caused by the pandemic, Covid-19, for the past almost year and a half, collecting points of data became very difficult, and, as a result, I was unable to get as much data as I would have liked. In the case of the fox hunt experiment, a test run of the experiment was completed with a Cal Poly class early in the year of 2020 [22], and, due to the pandemic, the fox hunt was unable to be run again as an official experiment to gather data from each participant. As a consequence, the data for the fox hunt results will include only my first hand account of the experiment test run and the performance of each group that I witnessed. As for the set of three labs—WiFi penetration testing, POCSAG decoding, and Morse Code decoding—an official experiment was able to be performed to collect the results of the individual participants, but with the pandemic going on I was only able to gather two participants to compare against my own data. So, these two sets of data, along with my own, will be the only results to represent the three lab experiment.

The fox hunt lab overall had a very high success rate from beginning to end out of the five groups of three to four students involved. For the Antenna assembly task, 100% of the groups were successful in assembling and testing working directional antennas on their first attempts. Once students moved onto setting up software to utilize their antennas, only one group began to run into issues—these issues were able to be resolved in a short amount of time though—resulting in an 80% success rate for the second task. On the last task—finding the hidden radio foxes—most groups were able to find both of the hidden foxes without issue. About 60% of the groups were able to find both foxes within the first half the lab period. For the remaining 40%, one group eventually found the last fox on their own in the last half of the lab period, and, with a little help, the remaining group was also able to locate the last fox. Overall the lab turned out well from my observation and I feel that it would give great results if performed again as an official experiment to gather data from each participant.

For the three individual labs, participants were tasked with using a set of tools—Aircrack-ng, PDW, Fldigi, and SDR#—chosen based on their BEAR scores in order to solve wireless CTF challenges like those that can be found in RF Sanctuaries WCTF [23].

After completion of the three individual labs—WiFi penetration testing, POCSAG signal decoding, and Morse Code signal decoding—participants would score each of the tools that were used—with the BEAR scoring rubric—to assess their usability. The results of these tool scores can be seen in figures 5.1 and 5.2. These figures show how the scores are split up between each of the four categories that make up BEAR—breadth, ease of use, accessibility, and reliability—as well as how those scores are divided among the sub categories that make up the four main categories. With my scores as a baseline when compared to the participants, the participants scores differed about 12% on average.

Lastly, figure 5.3 shows the change in experience level that each participant feels that they had between starting and finishing the experiment. With a maximum rating of 10, participants change in experience averages to an increase of about 4.375 points. While there is not a significant sample size for this experiment, with these small differences in tool scores and large increases in topic experience, I believe these show

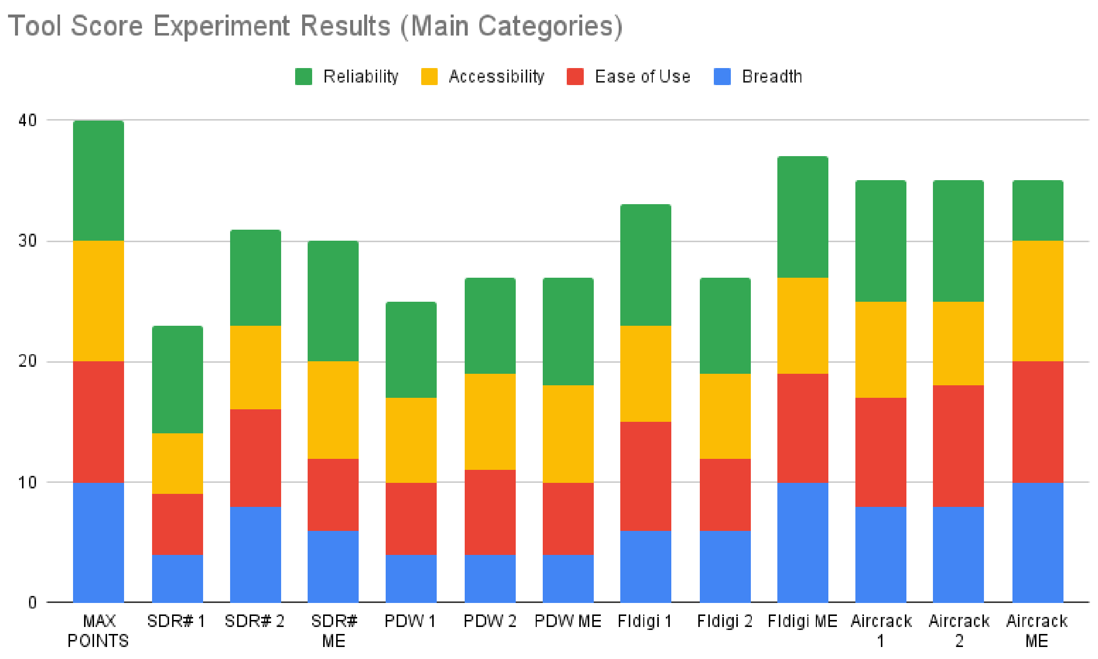

Figure 5.1: Tool score experiment results and comparison for all tools using main categories of BEAR usability rubric

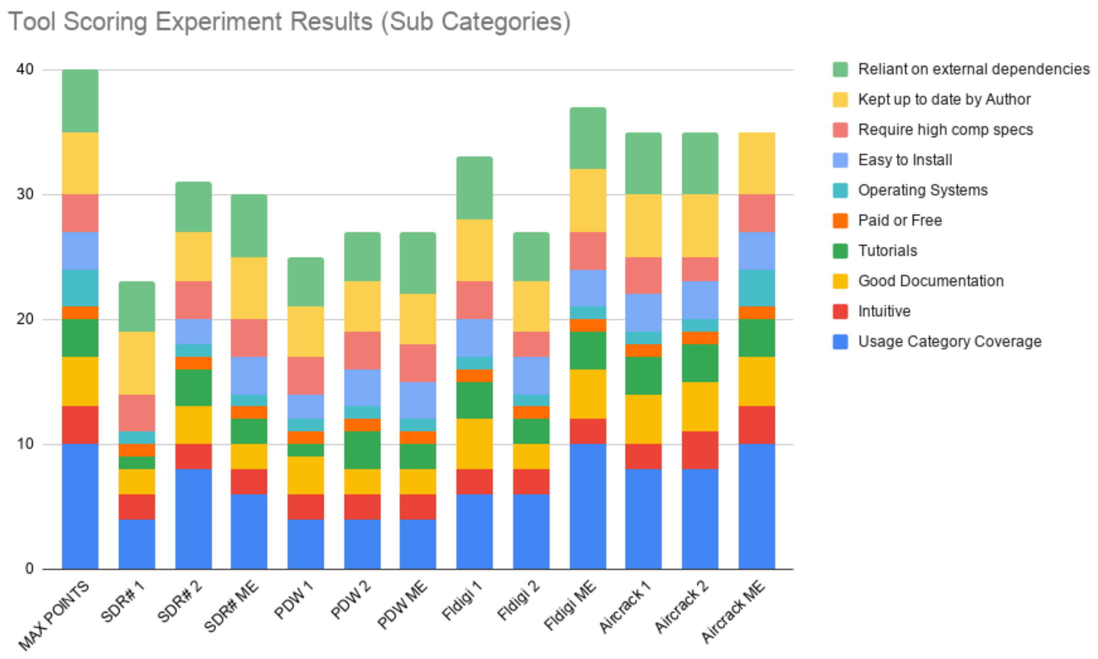

Figure 5.2: Tool score experiment results and comparison for all tools using sub categories of BEAR usability rubric

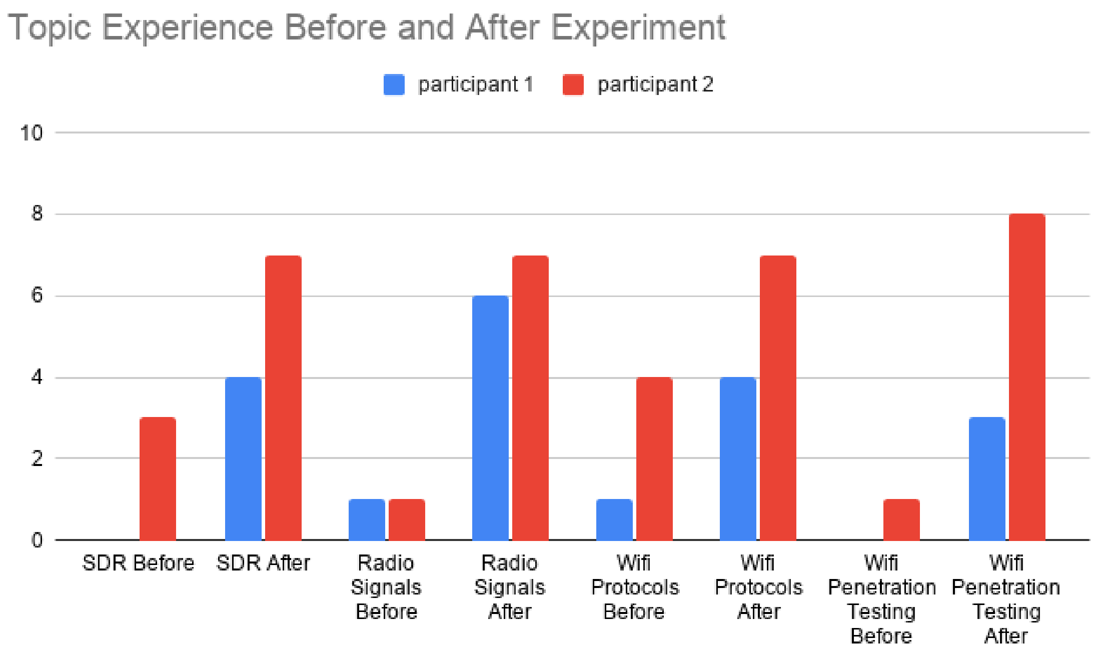

Figure 5.3: Participant experience levels with experiment topics before and after results

potential promise if applied to a larger sample size of participants with a more diverse level of experience

Chapter 6

CONCLUSION

The two goals of this thesis were to create a method of assessing cyber security tools, for wireless themed CTFs, based on usability, and to create a set of lab materials to introduce wireless themed CTF topics, utilizing a mixture of cyber security education principles to improve learning, and test the viability of the created tool scoring method. The creation of the usability tool scoring method was achieved and took the form of BEAR. However, the last goal wasn't fully achieved. Due to limitations and complications caused by the Covid-19 pandemic, the amount of data achieved was much less than I would have liked. As a consequence of this, the results from the labs don't offer as high of an accuracy as they could have. Despite this, I still believe that from the available resulting data these experiments still have promise to show better results if shown to a larger and more diversely experienced audience.

6.1 Future Work

In the future this experiment could receive a large benefit from being shown to a larger and more diverse set of participants. Incorporating many different experience and knowledge levels would provide new insight into where the experiment could be further improved as well as incorporating helpful feedback and comments from the participants. The fox hunt lab could further be expanded in the future to include not only the basic radio direction finding techniques used in this experiment but more advanced techniques like utilizing multiple antennas and correlation to calculate the angle of arrival of a signal in a 2d horizontal space [12].

BIBLIOGRAPHY

[1] Band plan: 80 and 2 meter bands. `http://www.arrl.org/band-plan`.

[2] Cal Poly Github. `http://www.github.com/CalPoly`.

[3] Ctf time. `https://ctftime.org`.

[4] Cve and cvss: Explained. `https://www.syxsense.com/cve-and-cvss-explained/`.

[5] Pico ctf. `https://picoctf.org`.

[6] Vulnerability metrics. `https://nvd.nist.gov/vuln-metrics/cvss`.

[7] Chapter 1 - a brief overview of the wireless world. In L. Barken, E. Bermel, J. Eder, M. Fanady, M. Mee, M. Palumbo, and A. Koebrick, editors, *Wireless Hacking*, pages 3–21. Syngress, Rockland, 2004.

[8] Aircrack-ng. Description, 2020. `https://www.aircrack-ng.org/index.html`.

[9] Airspy. Airspy, a high quality approach to software-define radio, 2021. `https://airspy.com/`.

[10] V. A. Aleven and K. R. Koedinger. An effective metacognitive strategy: Learning by doing and explaining with a computer-based cognitive tutor. *Cognitive science*, 26(2):147–179, 2002.

[11] R.-S. Blog. About rtl-sdr. `https://www.rtl-sdr.com/about-rtl-sdr/`.

[12] E. Botha and K. Faul. An introduction to radio direction finding, 2020. `https://www.alarisantennas.com/blog/an-introduction-to-radio-direction-finding/`.

[13] J. Browne. Basics of modulation and demodulation, 2017. `https://www.mwrf.com/technologies/systems/article/21847080/basics-of-modulation-and-demodulation`.

[14] L. Buttyán and L. Dóra. Wifi security–wep and 802.11 i. *EURASIP Journal on Wireless Communications and Networking*, 1:1–13, 2006.

[15] T. P. Chiem. *A study of penetration testing tools and approaches*. PhD thesis, Auckland University of Technology, 2014.

[16] A. A. Circuits. How to demodulate an am waveform, 2011. `https://www.allaboutcircuits.com/textbook/radio-frequency-analysis-design/radio-frequency-demodulation/how-to-demodulate-an-am-waveform/`.

[17] A. A. Circuits. How to demodulate an fm waveform, 2011. `https://www.allaboutcircuits.com/textbook/radio-frequency-analysis-design/radio-frequency-demodulation/how-to-demodulate-an-fm-waveform/`.

[18] J. M. Clark and A. Paivio. Dual coding theory and education. *Educational Psychology Review*, 3(3):149–210, Sep 1991.

[19] L. Cranor. Usability and design, 2021. `https://canvas.cmu.edu/courses/21700/files/folder/slides?preview=6003877`.

[20] Dave-W1HKJ. Welcome to the fldigi, 2021. `https://sourceforge.net/p/fldigi/wiki/Home/`.

[21] Disciminator and P. Hunt. Pdw paging decoding software, 2021. `https://www.discriminator.nl/pdw/index-en.html`.

[22] D. Fang and D. Sanchez. Fox hunting. University Class Lab, 2020.

[23] R. Farina. Rf ctf information, 2021.
`https://github.com/rfhs/rfhs-wiki/wiki`.

[24] A. Hahn and M. Govindarasu. An evaluation of cybersecurity assessment tools on a scada environment. In *2011 IEEE Power and Energy Society General Meeting*, pages 1–6, 2011.

[25] S. Kucek and M. Leitner. An empirical survey of functions and configurations of open-source capture the flag (ctf) environments. *Journal of Network and Computer Applications*, 151:102470, 2020.

[26] P. Kumaraguru, Y. Rhee, S. Sheng, S. Hasan, A. Acquisti, L. F. Cranor, and J. Hong. Getting users to pay attention to anti-phishing education: Evaluation of retention and transfer. In *Proceedings of the Anti-Phishing Working Groups 2nd Annual ECrime Researchers Summit*, eCrime '07, page 70–81, New York, NY, USA, 2007. Association for Computing Machinery.

[27] M. Martens. Build it: 2 meter tape measure yagi beam antenna, 2017.
`https://www.jpole-antenna.com/2017/02/07/build-it-2-meter-tape-measure-yagi-beam-antenna/`.

[28] R. E. Mayer and R. B. Anderson. The instructive animation: Helping students build connections between words and pictures in multimedia learning. *Journal of educational Psychology*, 84(4):444, 1992.

[29] E. Notes. What is morse code? what is cw?
`https://www.allaboutcircuits.com/textbook/radio-frequency-analysis-design/radio-frequency-demodulation/how-to-demodulate-an-fm-waveform/`.

[30] J. R. C. Nurse, S. Creese, M. Goldsmith, and K. Lamberts. Guidelines for usable cybersecurity: Past and present. In *2011 Third International Workshop on Cyberspace Safety and Security (CSS)*, pages 21–26, 2011.

[31] G. Roldán-Molina, M. Almache-Cueva, C. Silva-Rabadão, I. Yevseyeva, and V. Basto Fernandes. A comparison of cybersecurity risk analysis tools. *Procedia Computer Science*, 121:568–575, 01 2017.

[32] R. H. Sanctuary. Welcome to the rf hackers sanctuary, 2021. `https://wctf.us/`.

[33] M. P. Shah. *Comparative Analysis of the Automated Penetration Testing Tools*. PhD thesis, Dublin, National College of Ireland, 2020.

[34] N. Shevchenko. Threat modeling: 12 available methods, 2018. `https://insights.sei.cmu.edu/blog/threat-modeling-12-available-methods/`.

[35] Simplilearn. What is threat modeling: Process and methodologies, 2021. `https://www.simplilearn.com/what-is-threat-modeling-article`.

[36] W. Suparman and M. J. Homam. Development of transmitter and receiver for fox hunting activity. In *AIP Conference Proceedings*, volume 2173, page 020020. AIP Publishing LLC, 2019.

[37] TechTarget. Harmonic. `https://whatis.techtarget.com/definition/harmonic`.

[38] E. Tews and M. Beck. Practical attacks against wep and wpa. In *Proceedings of the Second ACM Conference on Wireless Network Security*, WiSec '09, page 79–86, New York, NY, USA, 2009. Association for Computing Machinery.

[39] E. Trickel, F. Disperati, E. Gustafson, F. Kalantari, M. Mabey, N. Tiwari, Y. Safaei, A. Doupé, and G. Vigna. Shell we play a game? ctf-as-a-service for security education. In *2017 {USENIX} Workshop on Advances in Security Education ({ASE} 17)*, 2017.

[40] F. van der Loo. *Comparison of penetration testing tools for web applications.* PhD thesis, Master thesis, Radboud University Nijmegen, 2011. http://www. ru. nl/publish . . . , 2011.

[41] Wavecom. Transmission modes: Pocsag. `http://www.wavecom.ch/content/ext/DecoderOnlineHelp/default.htm#!worddocuments/pocsag.htm`.

[42] L. Zhang-Kennedy, Y. Abdelaziz, and S. Chiasson. Cyberheroes: The design and evaluation of an interactive ebook to educate children about online privacy. *International Journal of Child-Computer Interaction*, 13:10–18, 2017.

[43] L. Zhang-Kennedy and S. Chiasson. A systematic review of multimedia tools for cybersecurity awareness and education. *ACM Comput. Surv.*, 54(1), Jan. 2021.

[44] L. Zhang-Kennedy, S. Chiasson, and R. Biddle. The role of instructional design in persuasion: A comics approach for improving cybersecurity. *International Journal of Human–Computer Interaction*, 32(3):215–257, 2016.

APPENDICES

Appendix A

TOOL SCORING RUBRIC

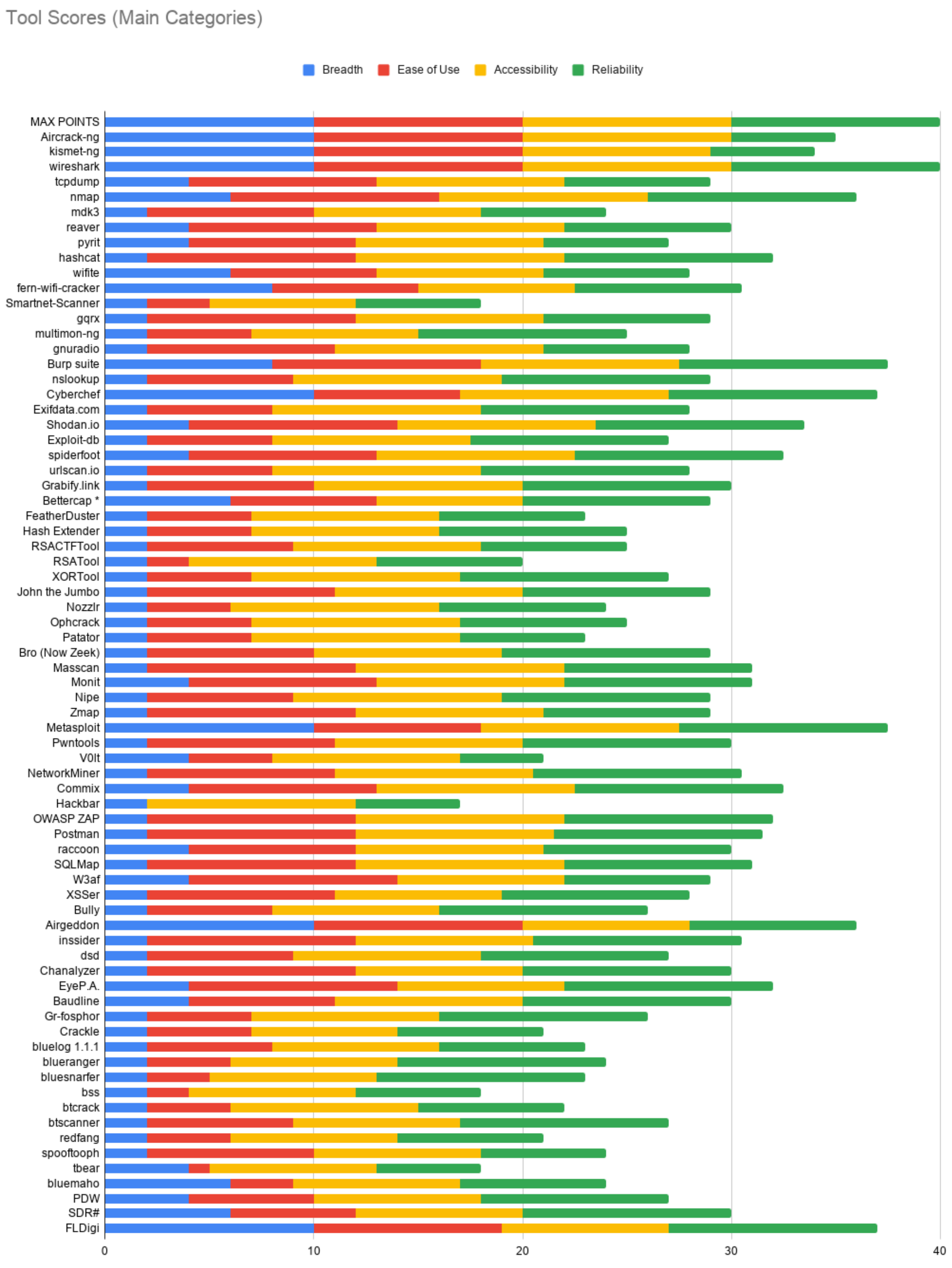

Figure A.1: Tool scores for all tools using main categories of BEAR usability rubric

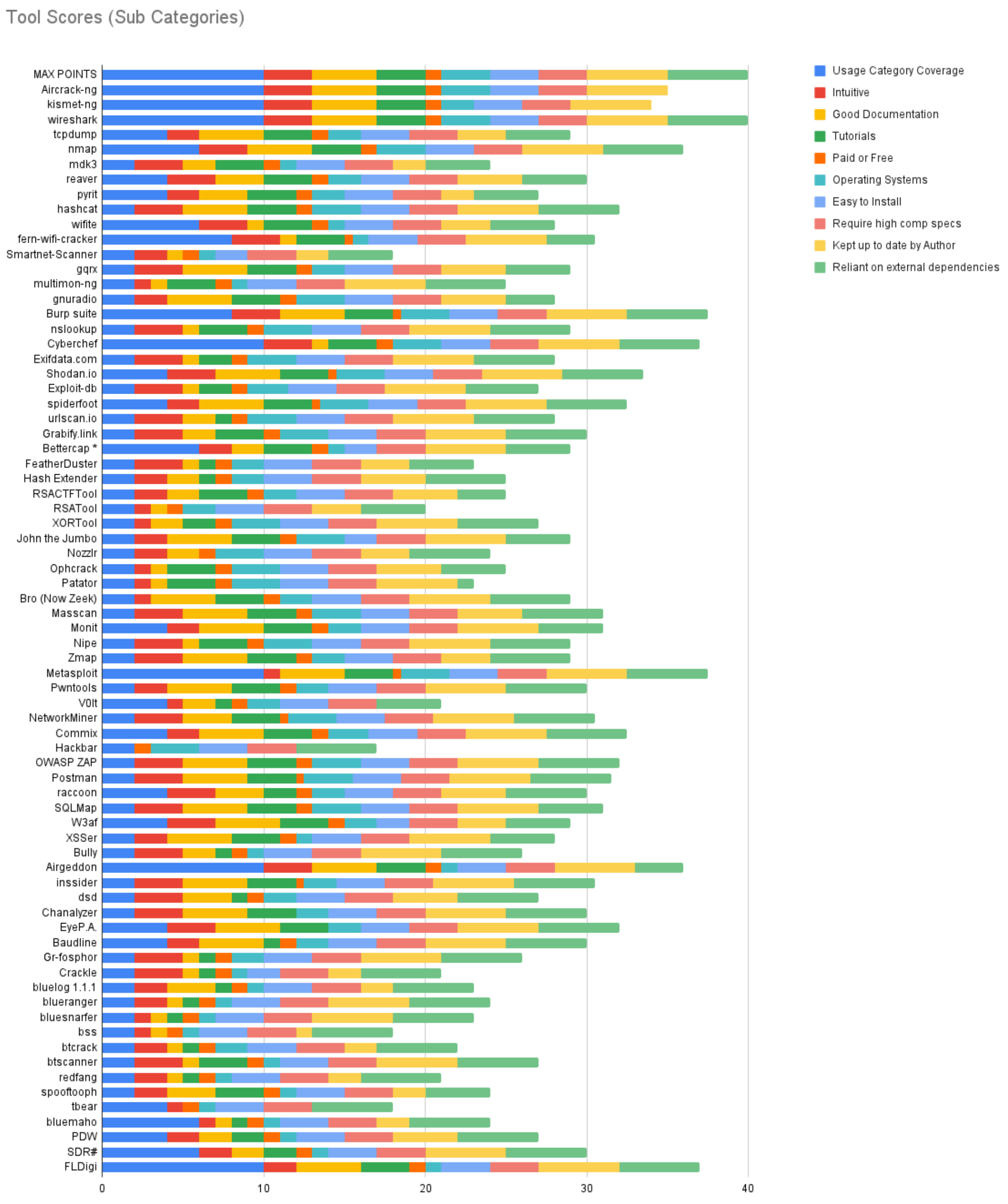

Figure A.2: Tool scores for all tools using sub categories of BEAR usability rubric

Appendix B

LABS

CSC-429
Wireless Security (Winter 2020)

Lab 6: Fox Hunting
Report Due: Mar. 6, 2020

Objectives

The objectives for this lab assignment are as follows:

- Making your own directional antenna.
- Fox hunt for a hidden transmitter.

Equipment:

- A directional antenna
- RTL-SDR device
- GQRX for Mac and Linux Machines
- SDR# for Windows Machines

Requirements

Based on limited materials, 3-4 students will be a group.

Task 1: Build a directional antenna

Please follow the instruction from this link:

https://www.jpole-antenna.com/2017/02/07/build-it-2-meter-tape-measure-yagi-beam-antenna/.

Make sure you are working on the steps carefully. Don't get hurt.

Task 2:

Installing GQRX if you have Mac or Linux system.

Linux: run "sudo apt-get install gqrx-sdr" from the command line to get the software and dependencies. Then run the program by typing "gqrx" in the terminal window.
Mac: Download GQRX binary package for Mac OS X from the GQRX download page https://gqrx.dk/download. Once the disk image is downloaded open the disk image and run the gqrx.app to start the application.

Installing SDR#
Windows: Download "Windows SDR Software Package" from the airspy download page https://airspy.com/download/.

Once you have the antenna and the software, you will use a RTL-SDR to connect the antenna with your computer. Information of RTL-SDR can be find from the following links:
https://www.rtl-sdr.com/about-rtl-sdr/
https://www.rtl-sdr.com/buy-rtl-sdr-dvb-t-dongles/.

Task 3: Hunting the Foxes.
The Foxes will be on campus.
The Foxes will not be close to buildings with lots of metal used in their construction: Engineering 4, architecture, etc. (Signals may bounce off of these structures a lot).
The frequency range of the Foxes will be 144-148 MHz. The Fox transmissions will be a series of tones followed by a Morse code call sign. (The Morse code call signs are the same, however, one fox will play higher pitched tones than the other fox).

For this Fox hunt, we won't use attenuators. Once you start getting close to the foxes it might become harder to tell where the signal is coming from since the signal will be pretty strong. In order find the signal more easily when it close proximity, you can change your listening frequency to one of the transmitter's harmonics (<transmitter freq> * <harmonic number>). You can usually use the third harmonic (<transmitter freq> * 3) or even the fifth harmonic to help reduce the signal strength in order to locate the direction of the transmission more easily. If you are having a hard time trying to find the signal, you can try starting from places that are high off the ground, which will have less issues from signals bouncing off of structures.

In Your Report

Please summarize the following in your lab report:

1. Describe the steps you did for this lab;
2. Explain the hard time you had.
3. Can you apply this to wireless threat hunting and detection? Please explain it.
4. Provide some information of wireless threat hunting and detection in this report based on your research.

Submission

You need to submit a detailed lab report to describe what you have done and what you have observed. Please also provide explanation to the observations that are interesting or surprising. One report is needed for a group.

Decoding Pager Transmissions with SDR (Windows)

Equipment needed:

- RTL-SDR dongle
- SMA Antenna

Software needed:

- Windows
    - SDR#
    - PDW
    - VB-Audio Cable
    - VirtualBox or VMWare for those without a dedicated windows machine

Tools being tested:

- Windows
    - SDR#
    - PDW

Setup:

- Install SDR#
    - Setup SDR#'s rtl-sdr driver replacement (setup guide can be found here)
- Install PDW
- Install VB-Audio Cable
- Download POCSAG sample for this lab
- Setup a Windows virtual machine if you don't have a dedicated Windows machine
    - You can use the video here or any other resources you can find
    - You will also need to get a windows iso from microsoft here

Note: Looking up tutorials and other documentation materials is highly encouraged

Lab:

This lab will be focused on finding **POCSAG** type pager radio messages and **decoding** them to find what data (usually ascii string messages) they are transmitting. To find these pager messages we will be using **SDR# to demodulate** radio signals for us to be able to listen to them, **VB-Audio Cable to send audio** between **SDR#** and **PDW**, and have **PDW decode** the transmissions..

Tasks:

Information Gathering

- Lookup what **POCSAG** signals typically **look** like (Either on a waterfall plot or a frequency-to-amplitude graph)
- Lookup what **POCSAG** signals **sound** like (This can help you identify a potential signal if a signal looks similar)
- Lookup **frequency ranges** that **POCSAG** transmissions will usually be found in (This will help narrow down where to look when trying to find POCSAG transmissions)
    - You could also lookup if there are any known frequencies in your local area that utilize POCSAG (helpful when searching for signals later)

Signal Finding and Decoding

- Now that you have some general information about **POCSAG** signals, go ahead and start **SDR#** and open the provided **POCSAG** sample.
- Once you have located a **POCSAG** transmission, center SDR#'s **tuning bar** on the transmission using Wide FM (**WFM**) demodulation
    - Adjust **bandwidth** to fit desired signal
    - Note: you can try using **NFM** however, when I tried, SDR# didn't seem to demodulate properly as PDW couldn't decode properly
- Once you are able to listen to the demodulated signal, configure **SDR#** to **output audio** to **VB-Audio Cable**.
    - Hint: adjust the **Squelch** setting in order to increase the signal strength required to produce audible sound (will help take away some of the extra static noise and allow for better decoding)
- Open and configure **PDW** to **receive audio** from **VB-Audio Cable**
- Configure **PWD** to allow for **POCSAG** decoding.
- Once you have everything set up you should start to see **decoded messages** in **PDW.**
    - For this if you are finding test, or unit response messages congratulations you decoded a POCSAG signal!

- Extra - Not Required -
Finding Signals in the Wild (This part is extra and can be done after you finish all labs)

- Connect and set up your **RTL-SDR** and **Antenna**
    - Use the **long telescopic antenna** to be able to adjust to a variable set of frequencies
- Open **SDR#** and set your source to the **RTL-SDR USB**
- Using information you have already gather about **POCSAG** signals, see if you can find any in your area
    - Adjust the antenna length accordingly to get as strong of a signal as you can
    - If you are trying to target specific frequencies use the following formulas (use these within the length limits of the antenna you have)
        - Antenna length in **ft** = 234 / frequency in **MHz**
        - Antenna length in **m** = 71.5 / frequency in **MHz**

Decoding Digital CW (Morse Code) Ham Radio with SDR (Windows)

Equipment needed:
- RTL-SDR dongle
- SMA Antenna

Software needed:
- Windows
  - SDR#
  - Fldigi
  - VB-Audio Cable
  - VirtualBox or VMWare for those without a dedicated windows machine

Tools being tested:
- Windows
  - SDR#
  - Fldigi

Setup:
- Install SDR#
  - Setup SDR#'s rtl-sdr driver replacement (setup guide can be found here)
- Install Fldigi
- Install VB-Audio Cable
- Download CW Sample for this lab
- Setup a Windows virtual machine if you don't have a dedicated Windows machine
  - You can use the video here or any other resources you can find
  - You will also need to get a windows iso from microsoft here

Note: Looking up tutorials and other documentation materials is highly encouraged
Lab:

This lab will be focused on finding **CW digital ham radio** messages and **decoding** them to find what data they are transmitting. To find these ham radio messages we'll be using **SDR#** to **demodulate** radio signals for us to be able to listen to them, and **VB-Audio Cable to send audio** between **SDR#** and **Fldigi,** and have **Fldigi decode** the transmissions.

Tasks:

Information Gathering

- Lookup what **CW** signals typically **look** like (Either on a waterfall plot or a frequency-to-amplitude graph)
- Lookup what **CW** signals **sound** like (This can help you identify a potential signal if a signal looks similar)
- Lookup **frequency ranges** that **CW** transmissions will usually be found in (This will help narrow down where to look when trying to find CW transmissions)
    - You could also lookup if there are any known frequencies in your local area that utilize CW (helpful when searching for signals later)

Signal Finding and Decoding

- Now that you have some general information about **CW** signals, go ahead and start **SDR#** and open the provided **CW** sample.
- Once you have located a CW transmission, center SDR#'s **tuning bar** on the transmission using CW demodulation
    - Adjust **bandwidth** to fit desired signal
- Once you are able to listen to the demodulated signal, configure **SDR#** to **output audio** to **VB-Audio Cable**.
    - Hint: adjust the **Squelch** setting in order to increase the signal strength required to produce audible sound (will help take away some of the extra static noise and can allow for better decoding)
- Open and configure **Fldigi** to **receive audio** from **VB-Audio Cable**
- Set **Op Mode** in **Fldigi** to **CW** and center your marker on its **waterfall plot**
- Once you have everything set up you should start to see **decoded characters** start to form words and statements in **Fldigi.**
    - For this lab try to find a message talking about some descriptions of a battery
    - If you were able to find the transmission or another congratulations you decoded a CW morse code signal!

- Extra - Not Required -
Finding Signals in the Wild (This part is extra and can be done after you finish all labs)

- Connect and set up your **RTL-SDR** and **Antenna**
    - Use the **long telescopic antenna** to be able to adjust to a variable set of frequencies
- Open **SDR#** and set your source to the **RTL-SDR USB**
- Using information you have already gather about **CW** signals, see if you can find any in your area
    - Adjust the antenna length accordingly to get as strong of a signal as you can
    - If you are trying to target specific frequencies use the following formulas (use these within the length limits of the antenna you have)
        - Antenna length in **ft** = 234 / frequency in **MHz**
        - Antenna length in **m** = 71.5 / frequency in **MHz**

Breaking WEP (Linux)

Equipment needed:

- Wifi enable router - set up using WEP encryption
- Laptop
- Spare device to connect to router network
- Wireless dongle with packet injection capability

Software needed:

- Linux
    - Aircrack-ng
- VirtualBox or VMWare for those without a dedicated linux machine

Tools being tested:

- Aircrack-ng

Setup:

- Install Aircrack-ng
- Setup Linux virtual machine: if you do not have a dedicated Linux machine
    - You can grab kali linux from their website here which will have Aircrack-ng already installed for you
- Router:
    - To begin configuring your router, start by powering on the device and connect to it with either an ethernet cable, or through wifi (if you already know the network name and password). Next open your web browser of choice and navigate to the router's configuration page (default gateway address), usually **192.168.0.1** or **192.168.1.1**. If either of these addresses do not work you can find the device's default gateway address by using the **\`ipconfig\`** on windows command prompt or **\`ifconfig\`** on Mac or Linux systems. On the routers configuration page find the wireless security page; here we will be changing the security mode to **WEP**, and setting the **WEP Algorithm** to use **TKIP**. Once the security settings have been set, change the set password to **\`password1234\`** for simplicity sake; as well as setting the routers Wireless network name (SSID) to a name that you will be able to easily recognize.

Note: Looking up tutorials and other documentation materials is highly encouraged
Lab:
This lab will focus on attacking **Wifi** routers that are using the weak **WEP** encryption scheme. For this attack we will need to find several pieces of information: Wifi router **BSSID**, Wifi router **channel**, and a **capture of communications** associated with the target router. This information can be gathered using tools within the **aircrack-ng suite**. During the capture portion of this lab you will need to **capture a large number of packets** in order to find enough **IVs** for this approach to work, results can vary but around 150,000+ IVs should be the target goal. By

getting this large amount of IV’s we are hoping to **find repeated IV’s**, allowing us to **break WEP encryption**. Once we have enough IVs in the captured communications we will use the **aircrack-ng suite** to **crack** the password based on the captured IVs. **If no key** is returned from aircrack-ng try the attack again after **gathering more IVs**.

Information gathering:

To gather the required information we will be using the tool suite aircrack-ng. First, we will be gathering general information about the router. Begin by setting your wireless interface into monitor mode which can be done using **airmon-ng.** You will need to find your wireless interface name to use with this command and can be foundby using **`iwconfig`** on the command line. Make note of the name that monitor mode is enabled on to use our next command.

Once your device is set in monitor mode begin listening to nearby network communications using **airodump-ng** with the name of our wireless interface. You will be presented with a table of **information about nearby network devices**. **Look** for the name that you gave your router and take note of its **BSSID**, and **CH** (channel). With these pieces of information we will now, instead of listening to all nearby network communications, focus on the capture of network communications from our specific device. Using the last pieces of information we took, use **airodump-ng** with the found **BSSID, channel,** and your **wireless interface** and **save** the **output** to a file**.** This will allow us to **only listen**, from our wireless interface, on the **specified channel** to the **given bssid** and **store** its **findings** in a file.

Now that you’re **listening** and **capturing traffic**, a **large number** of **IVs** will be needed for a successful attack. However, capturing traffic on this **controlled lab environment** will be **extremely slow**. So, in order to **speed up this process** a bit, **connect** a **spare device** (laptop or a phone) to the router. This device should show up in your current capture session as a station. Make **note** of its **MAC address** under **station** as we will be **spoofing** this to **inject ARP** traffic, speeding up our IV capture. To start this APR injection method you can utilize **aireplay-ng** set into **arpreplay mode** along with the **router BSSID**, **MAC address** of the **added device**, and your **wireless interface.**

Crack the Password

After a sufficient number of IVs have been captured, all that is left to do is **feed** the saved **capture output** to **aircrack-ng** to find the password. Once aircrack has worked through the file, if there were enough IVs present in the capture the password will eventually be displayed. If the password was unable to be found then try capturing more IVs and attempt the process again.

If you were able to get the password, congratulations you’ve broken WEP!

Appendix C

QUESTIONNAIRES

Entrance Survey

In this entry survey you will be asked a series of questions about any previous experience, or skills you may have in regards to any of the tools or wireless technology topics that will be brought up during this experiment. Please answer as honestly as you can so any change in your experience or skill level by the end of the experiment will be as accurate as possible.

1. How would you rate your experience level with Software Defined Radio (SDR) applications

Mark only one oval.

	0	1	2	3	4	5	6	7	8	9	10	
No experience	⬭	⬭	⬭	⬭	⬭	⬭	⬭	⬭	⬭	⬭	⬭	Very experienced and familiar

2. If you have previous experience with SDR applications please list those that you have used or are familiar with, if any.

3. How would you rate your knowledge level of different types of radio signals

Mark only one oval.

	0	1	2	3	4	5	6	7	8	9	10	
No knowledge	⬭	⬭	⬭	⬭	⬭	⬭	⬭	⬭	⬭	⬭	⬭	Very knowledgeable and familiar

4. If you have previous knowledge of types of radio signals please list those that you know of or are familiar with, if any

5. How would you rate your knowledge level of different types of WiFi protocols

Mark only one oval.

	0	1	2	3	4	5	6	7	8	9	10	
No knowledge	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	Very knowledgeable and familiar

6. If you have previous knowledge of different WiFi protocols please list those that you know of or are familiar with, if any

7. How would you rate your experience level with WiFi penetration testing software

Mark only one oval.

	0	1	2	3	4	5	6	7	8	9	10	
No experience	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	Very experienced and familiar

8. If you have previous experience with WiFi penetration testing applications please list those that you have used or are familiar with, if any.

Thank you for participating

Exit Survey

Now that you have finished working with the provided lab material, this exit survey will ask you a series of questions about any change in familiarity, knowledge level, or experience level with the related materials; and if you feel that, after working with these materials, you would be able to apply any of the used or learned techniques or software to other types of challenges in the same category (WiFi penetration testing, radio signal finding and decoding, etc). You will also be asked to provide scores about certain aspects of the tools used throughout the experiment to gauge their overall usefulness and usability.

1. After taking part in this experiment, how would you now rate your experience level with Software Defined Radio (SDR) applications

Mark only one oval.

	0	1	2	3	4	5	6	7	8	9	10	
No experience	○	○	○	○	○	○	○	○	○	○	○	Very experienced and familiar

2. After taking part in this experiment, how would you now rate your ability to find and decode radio signals

Mark only one oval.

	0	1	2	3	4	5	6	7	8	9	10	
Unable	○	○	○	○	○	○	○	○	○	○	○	Very capable

3. How likely do you feel that you would be able to apply the same skills and techniques towards similar radio challenges but for different radio signals

Mark only one oval.

	0	1	2	3	4	5	6	7	8	9	10	
Not likely	○	○	○	○	○	○	○	○	○	○	○	Very likely

4. After taking part in this experiment, how would you now rate your experience level with WiFi penetration testing software

Mark only one oval.

	0	1	2	3	4	5	6	7	8	9	10	
No experience	○	○	○	○	○	○	○	○	○	○	○	Very experienced and familiar

5. After taking part in this experiment, how would you now rate your ability to use penetration testing tools against WiFi protocols

Mark only one oval.

	0	1	2	3	4	5	6	7	8	9	10	
Unable	○	○	○	○	○	○	○	○	○	○	○	Very capable

6. How likely do you feel that you would be able to apply the same skills and techniques towards similar WiFi challenges but for different WiFi protocols

Mark only one oval.

	0	1	2	3	4	5	6	7	8	9	10	
Not likely	○	○	○	○	○	○	○	○	○	○	○	Very likely

7. Any comments, criticisms, or things about the experiment you would like me to know about?

The sections following this one will be for scoring tools used in the experiment

Tool Scoring for SDR#

In these sections of the survey you will be providing scores to different aspects of each of the tools used. Some of these aspects include, how intuitive the tools were to use, is there good documentation, is the tool kept up to date, is the tool reliant on a lot of dependencies, etc

8. How intuitive was SDR# to use

Mark only one oval.

	0	1	2	3	
Impossible to navigate without full knowledge and experience	○	○	○	○	Easy to understand and work with

9. Does SDR# have good documentation

Mark only one oval.

	0	1	2	3	4	
Bad or no documentation	○	○	○	○	○	Documentation is understandable and easy to read

10. How easy was it to find tutorials and helpful information for SDR#

Mark only one oval.

	0	1	2	3	
No tutorials exist	○	○	○	○	Easy to find tutorials and materials

11. How many categories does SDR# cover, use your best judgment (wep, wpa, wpa2, radio direction finding, etc)

Mark only one oval.

	1	2	3	4	5	
Specific use tool	◯	◯	◯	◯	◯	Diverse use cases

12. Does SDR# require payment to use, have a limited free version, or completely free

Mark only one oval.

	0	1	2	
Requires payment	◯	◯	◯	Completely free

13. Which operating systems does SDR# support

Check all that apply.

- ☐ Windows
- ☐ MacOS
- ☐ Linux

14. How easy was it to install SDR#

Mark only one oval.

	0	1	2	3	
Multiple commands and self management to install	◯	◯	◯	◯	One command install or all in one installer

15. Does SDR# require a high spec computer

Mark only one oval.

	0	1	2	3	
Requires very high spec components	◯	◯	◯	◯	Able to run on low spec components

16. Is SDR# kept up to date

Mark only one oval.

	0	1	2	3	4	5	
Project is dead (hasn't updated for last 10 years or more)	◯	◯	◯	◯	◯	◯	Updated frequently less than 2 years since last update

17. Is SDR# reliant on many external dependencies

Mark only one oval.

	0	1	2	3	4	5	
Many dependencies (10 or more)	◯	◯	◯	◯	◯	◯	Self contained

Tool Scoring for PDW

In these sections of the survey you will be providing scores to different aspects of each of the tools used. Some of these aspects include, how intuitive the tools were to use, is there good documentation, is the tool kept up to date, is the tool reliant on a lot of dependencies, etc

18. How intuitive was PDW to use

Mark only one oval.

	0	1	2	3	
Impossible to navigate without full knowledge and experience	◯	◯	◯	◯	Easy to understand and work with

19. Does PDW have good documentation

Mark only one oval.

	0	1	2	3	4	
Bad or no documentation	◯	◯	◯	◯	◯	Documentation is understandable and easy to read

20. How easy was it to find tutorials and helpful information for PDW

Mark only one oval.

	0	1	2	3	
No tutorials exist	◯	◯	◯	◯	Easy to find tutorials and materials

21. How many categories does PDW cover, use your best judgment (wep, wpa, wpa2, radio direction finding, etc)

Mark only one oval.

	1	2	3	4	5	
Specific use tool	◯	◯	◯	◯	◯	Diverse use cases

22. Does PDW require payment to use, have a limited free version, or completely free

Mark only one oval.

	0	1	2	
Requires payment	◯	◯	◯	Completely free

23. Which operating systems does PDW support

Check all that apply.

- [ ] Windows
- [ ] MacOS
- [ ] Linux

24. How easy was it to install PDW

Mark only one oval.

	0	1	2	3	
Multiple commands and self management to install	⬭	⬭	⬭	⬭	One command install or all in one installer

25. Does PDW require a high spec computer

Mark only one oval.

	0	1	2	3	
Requires very high spec components	⬭	⬭	⬭	⬭	Able to run on low spec components

26. Is PDW kept up to date

Mark only one oval.

	0	1	2	3	4	5	
Project is dead (hasn't updated for last 10 years or more)	⬭	⬭	⬭	⬭	⬭	⬭	Updated frequently less than 2 years since last update

27. Is PDW reliant on many external dependencies

Mark only one oval.

	0	1	2	3	4	5	
Many dependencies (10 or more)	⬭	⬭	⬭	⬭	⬭	⬭	Self contained

Tool Scoring for Fldigi

In these sections of the survey you will be providing scores to different aspects of each of the tools used. Some of these aspects include, how intuitive the tools were to use, is there good documentation, is the tool kept up to date, is the tool reliant on a lot of dependencies, etc

28. How intuitive was Fldigi to use

Mark only one oval.

	0	1	2	3	
Impossible to navigate without full knowledge and experience	⬭	⬭	⬭	⬭	Easy to understand and work with

29. Does Fldigi have good documentation

Mark only one oval.

	0	1	2	3	4	
Bad or no documentation	◯	◯	◯	◯	◯	Documentation is understandable and easy to read

30. How easy was it to find tutorials and helpful information for Fldigi

Mark only one oval.

	0	1	2	3	
No tutorials exist	◯	◯	◯	◯	Easy to find tutorials and materials

31. How many categories does Fldigi cover, use your best judgment (wep, wpa, wpa2, radio direction finding, etc)

Mark only one oval.

	1	2	3	4	5	
Specific use tool	◯	◯	◯	◯	◯	Diverse use cases

32. Does Fldigi require payment to use, have a limited free version, or completely free

Mark only one oval.

	0	1	2	
Requires payment	◯	◯	◯	Completely free

33. Which operating systems does Fldigi support

Check all that apply.

- ☐ Windows
- ☐ MacOS
- ☐ Linux

34. How easy was it to install Fldigi

Mark only one oval.

	0	1	2	3	
Multiple commands and self management to install	◯	◯	◯	◯	One command install or all in one installer

35. Does Fldigi require a high spec computer

Mark only one oval.

	0	1	2	3	
Requires very high spec components	⬭	⬭	⬭	⬭	Able to run on low spec components

36. Is Fldigi kept up to date

Mark only one oval.

	0	1	2	3	4	5	
Project is dead (hasn't updated for last 10 years or more)	⬭	⬭	⬭	⬭	⬭	⬭	Updated frequently less than 2 years since last update

37. Is Fldigi reliant on many external dependencies

Mark only one oval.

	0	1	2	3	4	5	
Many dependencies (10 or more)	⬭	⬭	⬭	⬭	⬭	⬭	Self contained

Tool Scoring for Aircrack-ng

In these sections of the survey you will be providing scores to different aspects of each of the tools used. Some of these aspects include, how intuitive the tools were to use, is there good documentation, is the tool kept up to date, is the tool reliant on a lot of dependencies, etc

38. How intuitive was Aircrack-ng to use

Mark only one oval.

	0	1	2	3	
Impossible to navigate without full knowledge and experience	⬭	⬭	⬭	⬭	Easy to understand and work with

39. Does Aircrack-ng have good documentation

Mark only one oval.

	0	1	2	3	4	
Bad or no documentation	⬭	⬭	⬭	⬭	⬭	Documentation is understandable and easy to read

40. How easy was it to find tutorials and helpful information for Aircrack-ng

Mark only one oval.

	0	1	2	3	
No tutorials exist	⬭	⬭	⬭	⬭	Easy to find tutorials and materials

41. How many categories does Aircrack-ng cover, use your best judgment (wep, wpa, wpa2, radio direction finding, etc)

Mark only one oval.

	1	2	3	4	5	
Specific use tool	◯	◯	◯	◯	◯	Diverse use cases

42. Does Aircrack-ng require payment to use, have a limited free version, or completely free

Mark only one oval.

	0	1	2	
Requires payment	◯	◯	◯	Completely free

43. Which operating systems does Aircrack-ng support

Check all that apply.

- ☐ Windows
- ☐ MacOS
- ☐ Linux

44. How easy was it to install Aircrack-ng

Mark only one oval.

	0	1	2	3	
Multiple commands and self management to install	◯	◯	◯	◯	One command install or all in one installer

45. Does Aircrack-ng require a high spec computer

Mark only one oval.

	0	1	2	3	
Requires very high spec components	◯	◯	◯	◯	Able to run on low spec components

46. Is Aircrack-ng kept up to date

Mark only one oval.

	0	1	2	3	4	5	
Project is dead (hasn't updated for last 10 years or more)	◯	◯	◯	◯	◯	◯	Updated frequently less than 2 years since last update

47. Is Aircrack-ng reliant on many external dependencies

Mark only one oval.

	0	1	2	3	4	5	
Many dependencies (10 or more)	◯	◯	◯	◯	◯	◯	Self contained

Thank you for participating