Improving Security Policy Decisions with Models

Tristan Caulfield
t.caulfield@ucl.ac.uk

David Pym
d.pym@ucl.ac.uk

Abstract

Security managers face the challenge of designing security policies that deliver the objectives required by their organizations. We explain how a rigorous methodology, grounded in mathematical systems modelling and the economics of decision-making, can be used to explore the operational consequences of their design choices and help security managers to make better decisions. The methodology is based on constructing executable system models that illustrate the effects of different policy choices. Models are designed to be composed, allowing complex systems to be expressed as combinations of smaller, complete models. They capture the logical and physical structure of systems, the choices and behavior of agents within the system, and the security managers' preferences about outcomes. Models are parameterized from observations of the real world and the effectiveness of different security policies is explored through simulation. Utility theory is used to describe the extent to which security managers' policies deliver their security objectives.

1 Introduction

Security managers face the challenging task of setting policies that will meet their organizations' security objectives without being fully aware of what the exact consequences of these policies will be once they are implemented. A policy is a collection of principles and rules that describe how an organization intends to protect the confidentiality, integrity, and availability of its systems and the information that they process. Policies may refer both to the physical and to the virtual environments in which the organization operates and, typically, will be concerned with the behaviour of people, processes, and technologies. For example, a company may require that only certain members of staff may access a particular database and that those staff must do so only from a specific room within the company's building, access to which is controlled by encrypted badges. Many factors influence the eventual effectiveness of a policy: the security culture of an organization, the level of threats against it, the difficulty of compliance, and so on.

Particularly important is the realization that employees only have a limited capacity to follow security policies. In most cases, security is secondary to productivity and employees that have exhausted their 'compliance budget' [2] will seek to create work-arounds for policies that are perceived as overly demanding. It is hard to predict how employees will respond to a policy, and many are implemented without giving this any consideration at all.

With all of these different factors, and the difficulty accurately of predicting the final outcome, selecting a security policy is filled with uncertainty and often done based on experience, best practice, or simply what everyone else is doing. These approaches don't reflect the particular circumstances or characteristics of an organization and the performance of the policy is likely to suffer as a result, meaning that the security requirements of an organization will not be met.

In this article, we explain our solution to the problem of predicting the effectiveness of an organization's security policies and their impact on its operations. We build upon a range of ideas from the economics of information security [1, 3, 7] and mathematical systems modelling [5] to create a modelling methodology and framework for building models that can be used to predict the consequences of different policy choices.

A model is a representation of the system of interest, and this can then be used to help think about and explore details of the system's operation. A model allows you to ask questions such as 'How often does this type of event occur under these conditions?' or 'What will the consequences be if this parameter is adjusted?' For security policy, models should let you explore the interactions between the security policies, the technologies and tools used to implement them, the people who interact with the policies, and the environment in which all of those things reside. A security manager should be able to ask questions about different policy choices and then understand the outputs of the model in terms of how well it fulfils an organization's requirements for security. This way, predictions about different policies can be made, and the results compared to find the best option.

In practice, organizations can be large and complex, as can be the systems and policies used to secure them. To handle this complexity, our methodology is designed to be compositional. This means that smaller models, representing parts of the overall system, can be combined together to create models representing larger parts of the system. Using the smaller models, the effects of policies that apply to specific parts of the system can explored independently. When composed together, their combined effect on the whole system can be determined.

2 Modelling

The models in our methodology are executable systems models, based on classical models of distributed systems [6], which allows us to capture in a model all of the features necessary for making a useful tool for thinking about security policy.

Distributed systems modelling uses the concepts of locations, resources, and processes. Locations represent physical and logical locations; for example, a room, someone's pocket, a location in computer memory, or a person's mind. These locations are connected together by links, and together the locations and links form the model's location graph. Locations contain resources, which represent things needed for the model, such as physical objects, people, or information. Finally, processes model dynamic parts of the system. Models are executable; they have to be run in order to produce output. As a model runs, its processes manipulate the resources of the model.

Underlying these concepts is a strong mathematical foundation (e.g., [5, 10] and references therein). Such a foundation ensures that the models we implement soundly represent the systems that we wish to understand. This is achieved by using a mathematical description of the components of a system, together with a formal semantics of the language used to implement system models. Moreover, this semantically justified set-up supports our compositional theory of modelling, ensuring that combinations of sound models are themselves sound. We explain this mathematical foundation in [4].

Locations are most simply modelled mathematically using directed graphs — that is, collections of vertices connected by edges that have a specified direction.

Resources are modelled using a mathematical structure that captures some basic resource-like properties: given two stocks of a given type of resource, such as computer memory or money, we expect to be able to combine them, to form a greater stock, and also compare them, to determine which stock is the larger. The appropriate abstract mathematical structure for this purpose is called an ordered monoid [5].

The appropriate mathematical set-up for modelling processes uses a theory called process algebra [13],

in which complex processes are constructed from basic actions using combinators, similar to logical connectives, for sequencing, choice, concurrent or parallel execution, recursion, and so on. In our set-up, locations, resources, and processes co-evolve and we write

$$L, R, E \xrightarrow{a} L', R', E'$$

to indicate that a process E with access to resources R at location L can evolve by performing action a to become the process E' with access to resources R' at location L'. This relationship between the states L, R, E of model is defined logically by a formal operational semantics [5].

The systems that we choose to model do not exist in isolation. Rather, they operate in an environment with which, typically, they interact. Events within the environment, which is not modelled in detail, may have effects within the model and vice versa. Mathematically, we handle this situation stochastically. That is, we use probability distributions to describe the flow of events between a model and its environment. For example, in the tailgating model described above, we model the arrival of employees at the building (from the outside world, which is the environment) using a 'negative exponential' distribution [10].

This mathematical set-up can be captured in programmable tools. The Gnosis tool [5] captures it very closely, but is essentially a proof-of-concept implementation. We have used the Julia language [11] to implement the mathematical framework in a way that supports a natural programming style.

A model is a simplification of the real-world system with the level of abstraction and level of detail chosen to be appropriate for the questions that the modeller wishes to answer. A model is almost always not a complete a description of the real-world system: 'the map is not the territory'.

The process of building a model and collecting the data needed to parametrize it is a cycle. An initial model is constructed, based on observations of the real-world system. The consequences of the model are then interpreted as real-world consequences, and these are then compared to the real-world observations. If they are close enough, then the model can be considered complete; if not, then the model is adjusted or rebuilt, an the cycle continues until the model is good enough for its purpose.

What the model is capable of accurately representing depends significantly on the data that is available and the ease of collecting it. If the model requires data that is not available, the modeller must adjust the model to remove the need for that data.

3 Models of Organizational Security Policies

The three models used as examples are models of different aspects of an organization's security. The first model looks at tailgating behaviour of employees and attackers at the entrance to an office building. The second is about how confidential documents are shared between employees inside the office, when the normal, secure method is unavailable. The final model looks at how often employees' devices, possibly containing confidential information, are lost. The models are represented in Figure 1, which shows the physical and virtual locations used in each model, as well as a depiction of the agents' processes.

These three models are designed so that they can later be composed, allowing the modeller to examine the interactions between parts of the organization's security policy. How the models fit together is determined by interfaces, which are defined as part of the model. The interfaces specify locations that are shared between the models, such as the Outside and Atrium locations in Figure 1.

When composed, these models provide an analysis of how confidential information that is processed within the organization can be lost. Shared confidential documents that end up on a mobile device within the document-sharing model may be exposed if the mobile device is lost within the device-loss model. Similarly, shared confidential documents that end up on portable media with the document-sharing model

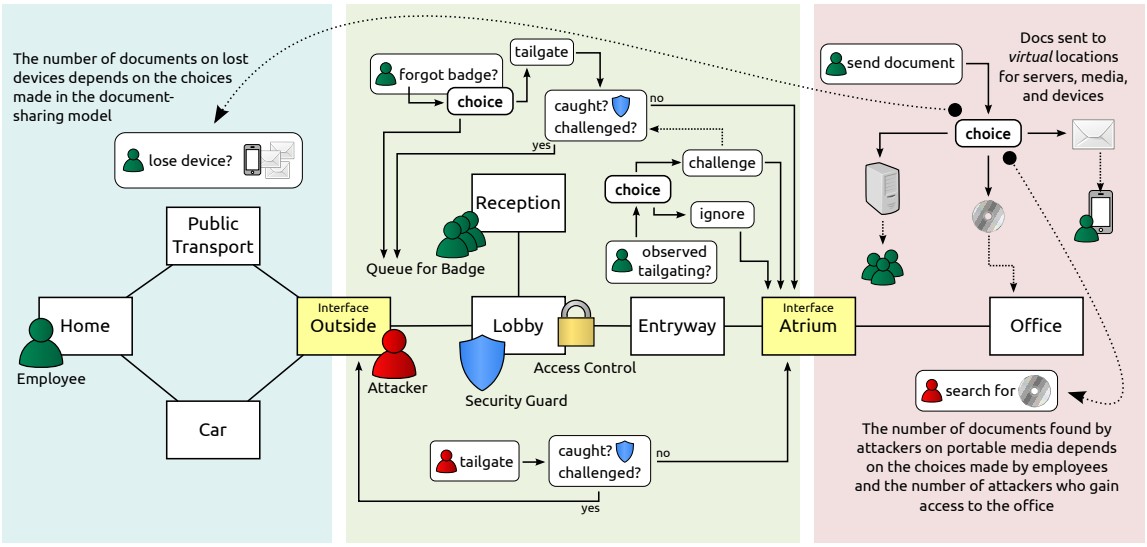

Figure 1: The three models. On the left is the device loss model, in the centre is the tailgating model, and on the right is the document-sharing model. The models compose together at the Outside and Atrium locations. In composed models, events and decisions in one component model can influence the behaviour of the other component models.

may be exposed when attackers gain access to them having penetrated the access control represented in the tailgating model. While each individual model captures particular aspects of organizational security, the composed model gives an explanation of how breaches of physical security and cybersecurity can lead to the exposure of an organization's confidential information.

When a model with interfaces is uncomposed, some processes that start at interface locations are initiated by the environment. For example, in Figure 1, when the tailgating model is not composed with the device-loss model, the employee processes are started by the environment in the Outside location. When the model is composed, the employees are started by the environment in the Home location of the device-loss model. The environment no longer starts the employee processes at the Outside location. Instead, the process is executed in this location as a consequence of the device-loss model.

We explain the mathematical theory of interfaces and their use in composing models in [4].

3.1 Tailgating

The tailgating model looks at the behaviour of employees and attackers as they enter the building. In the model, employees must use a swipe card to access the office. If they forget their card, the employee must make a decision: to attempt to tailgate behind another employee, or to queue at reception to get a temporary badge.

The tailgating model is in the centre of Figure 1. It is comprised of five locations: Outside, Lobby, Reception, Entryway, and Atrium. Agents start in the Outside location and progress to the Lobby. If they don't have a swipe card, this is where they make the decision about whether or not to tailgate. Only employees make this decision: attackers always attempt to tailgate.

Agents who choose to queue at reception move to the Reception location. Those that tailgate move to

Entryway, as do those that remembered their cards. If employees in the entryway observe a tailgater, they have another choice: to intervene and stop the tailgating agent, or to ignore it and continue. If the tailgating agent is not challenged, it proceeds to the Atrium.

3.2 Document Sharing

The second model examines choices around how to share information within the office. Employees need to send documents to each other. In normal operation, these documents are shared between on-site employees using a server that restricts access to only those with the correct permission. This model examines what happens outside of normal operation when employees must share the documents using a different method.

This model is on the right-hand side of Figure 1. It has only two locations: Atrium and Office. Agents arrive in the Office from the Atrium. Employees begin to work, and when a problem occurs sharing a document the normal way, a decision about how to share it is made. There are three choices in the model. First, the document can be put onto a global share, accessible by every employee in the company. Second, the document can be emailed to the recipient. Finally, the document can be given to the other employee on some form of portable media, like a CD or USB key. In each of these cases, a resource representing a document is moved to a location representing the place where it is digitally stored; that is, a virtual location.

Attackers in this model enter the office and then begin to look around to try and find any media that has been left lying around. How successful an attacker is at finding any depends on the number of disks, and how many desks there are in the office.

3.3 Device Loss

The final model is quite simple and deals with device loss outside of the office. Employees may have confidential information on their devices which, when the devices are lost, may be exposed. This model is on the left side of Figure 1. It has four physical locations: Home, Car, Public Transport, and Outside. Each employee starts at home, and travels to work using either public or private transportation. After the work day is over, the employees return home using the same method. Those that take public transportation risk losing or leaving behind their phones during the commute.

Here, mobile devices are modelled as bundle of a resource, which represents the device itself, and a location, which represents the device's data store. In the composed model, confidential documents sent by email are moved to the devices' data stores (i.e, downloaded by an email client).

4 Agents and Decision-making

Models contain the notion of agents: autonomous decision-makers that interact with other agents and resources within a model. In the distributed-systems modelling approach [5], there is no native concept of an agent. Instead, we create the notion of an agent using a bundle of a process, a resource, and locations. An agent has a resource that represents its location within the model. It also has a process that describes how the agent behaves — how its resource moves around the location graph, and how it interacts with other resources. Finally, an agent has several locations which contain resources the agent is carrying or wearing, or information that the agent knows.

An important part of the models is the way agents, typically employees, interact with the security policies and technologies in place. Employees must make decisions about, among other things, whether or not to comply with security. The framework handles this through a choice function, which is used to model a point in a process where a decision is made. Each choice point is a decision between a number of alternatives.

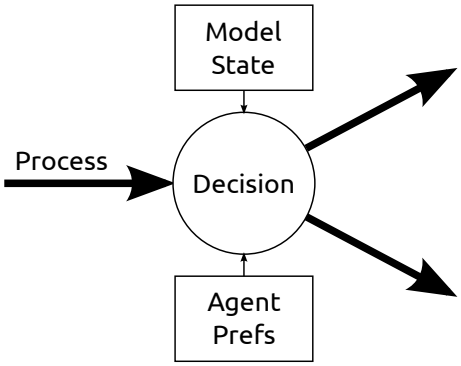

Figure 2: A decision point in an agent's process. The decision is made based on the state of the system and the agent's preferences.

The mechanism used to choose between these alternatives is not set explicitly in the framework: it is left to the model designer to choose a method suitable for the particular models being created.

In our models, each agent has preferences for security and productivity, and make decisions based on these preferences and the current state of the model, as shown in Figure 2. The agent's preferences are drawn at the beginning of model execution from distributions which describe the preferences of the population of agents as a whole. This means that each execution will involve agents with different preference values, possibly resulting in different outcomes.

To decide between different choices, an agent first calculates a value for each choice C:

$$val(C) = p_{sec} f_{sec}(C) + p_{prod} f_{prod}(C)$$

where p_{sec} is the agent's preference towards security, p_{prod} is the agent's preference towards productivity, and $f_{sec}(C)$ and $f_{prod}(C)$ are functions that calculate, given the current state of the model, how secure and how productive, respectively, the choice C is. Whichever choice has the highest value is then selected by the agent.

5 System Manager's Utility

So far, we have discussed how models are built and how agents within them make decisions. Now, we consider how system managers can understand the values of policies described by models. To be a useful tool for thinking about different policy choices, a model must be capable of indicating whether one particular choice is better than another. This is done by expressing the output of a model as a utility [12] for the system manager, which gives a value for the system's performance relative to the manager's objectives. If the utility of one policy choice is greater than that of another, then it better achieves the manager's objectives.

A security manager cares about different attributes of the system [12]. These attributes could be broad concepts, like confidentiality, integrity, and availability, or they could be more specific, such as the number of successful tailgating attempts, or the time employees spend interacting with security. For each of these attributes, the security manager has a target value; the target value for availability would presumably be high, while the target number of successful tailgating attempts might be zero. The manager cares about each of these attributes to a different degree — some are more important than others — and also cares differently

about how the values for each attribute deviate from their target. For example, a small increase in the time an employee spends interacting with security might not matter very much until the deviation crosses a threshold, whereas any decrease in availability matters a great deal.

The utility can then be described by the equation

$$Utility = \sum_{a \in A} w_a f_a(v_a - \bar{v}_a)$$

where for an attribute a from the set A of attributes the mananger cares about, w_a is the relative weighting, or importance, of that attribute, v_a is the actual value of the attribute, $\bar{v_a}$ is the target value, and f_a is the function that represents how the system manager cares about deviations from the target value for this particular attribute. If the values of the attributes are stochastic, then this represents the expected utility.

The target values, weights, and the deviation functions must be obtained from the system manager. The actual values for each attribute can be collected from the real world, to evaluate the current system policy, or from models, to evaluate possible changes to policy.

5.1 Attribute Values from Models

The values of the different attributes depend on the decisions agents make. Each decision has a number of alternatives, and each alternative has a value to the system manager. A particular decision, D, with choices $C_1, C_2, \ldots, C_n$, made by an agent during the execution of a model, can be written as

$$D = C_1^{\lambda_1} C_2^{\lambda_2} \ldots C_n^{\lambda_n}$$

where the choice that was selected has $\lambda = 1$ and the rest have $\lambda = 0$. This is in the form of a Cobb-Douglas production function [8].

As an agent explores a model, making decisions D, the values v_a of one or more of the system manager's chosen attributes a are affected, so affecting the overall utility for the manager. For example, if one of the manager's attributes is the number of incidents of tailgating, then each time an agent makes a decision to tailgate or not — so complying or not with the policy — the utility is affected.

This form is only useful for talking about decisions that have been made after the execution of a model. To think about decisions before the model has been executed, we need to consider the expected value of the decisions. The choice an agent selects at a decision point can be stochastic; either because the choice function itself is stochastic, or because the decision is based on the current state of the model, which is based on draws from distributions. A standard transformation of the previous function allows it to be used for expected values [14]:

$$E[D] = \lambda_1 ln(C_1) + \lambda_2 ln(C_2) + \ldots + \lambda_n ln(C_n)$$

where $\lambda_1, \ldots, \lambda_n$ are now probabilities and sum to 1.

The expected values of decisions are useful because they tell us about the values for all possible different paths the models execution could take, rather than just the path taken during a single execution. However, calculating the probabilities for different choices, while in theory possible, is extremely difficult. Instead, a different approach can be used: by executing the model a very large number of times, an approximation of the expected values can be determined and these can be used to calculate the expected utility of the policy as a whole. This expected utility is what is used by the system manager to judge the value of a policy.

6 Example: Policy Choices

To show how models can be useful tools for thinking about security policy we will use an example: the normal, secure method for sharing documents with other employees often breaks or does not allow a document to be shared with the necessary people. The security manager for the organization is concerned about how many documents are ending up on a globally-accessible shared folder, allowing all employees to view it and wants to consider recommending sharing the documents via email or via portable media as an alternative.

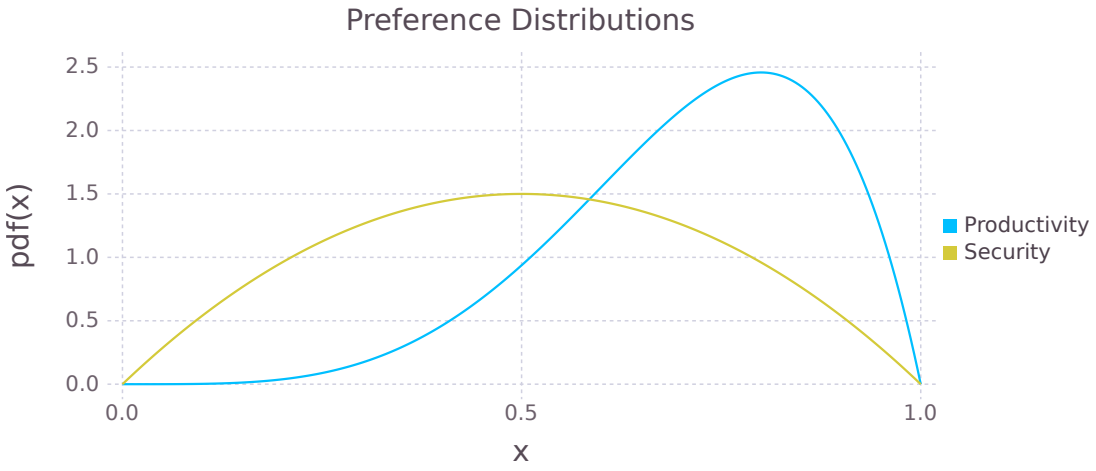

Figure 3: The distributions of preferences used for agents in the models. The horizontal axis is preference for the given attributes (here, productivity and security); the vertical axis is its probability density function. Most agents will have a higher preference for productivity than for security.

The decisions the employees make in the model depend on their preferences towards productivity and security. For each execution of the model, the preferences for each employee are drawn from probability distributions; these are shown in Figure 3 and would be selected to match the actual attitudes of employees in the company. In this example, the distributions show a fairly typical situation where the employees are much more likely to favor productivity over security.

	Documents			
Policy	To e-Mail	To Global	To Media	Found by Attacker
Neutral	0	142.9163	0	0
Media	0	91.4387	51.5196	0.2278
Email	99.2685	44.1443	0	0

Table 1: Results from the Document Sharing model.

The security manager then runs the model for the current policy choice, giving no suggestion to em-

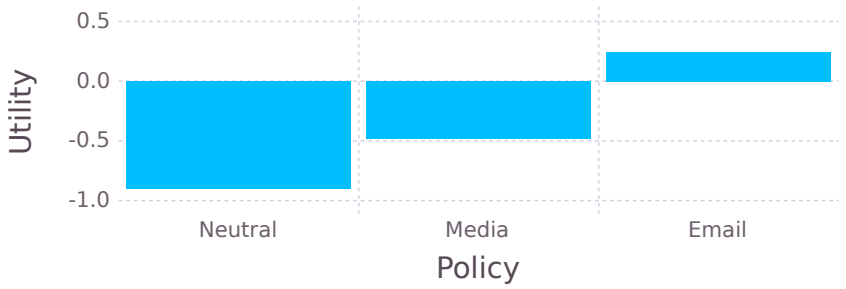

Figure 4: Security manager's utility for different policy choices in the Document Sharing model.

ployees, and the two alternatives, suggesting to employees that sending documents via email or media is the preferred, more secure method. The results of these simulation runs are presented in Table 1. This shows, for each policy alternative, the average values (over 10,000 simulation runs, with 400 agents) of how many documents were distributed by email, how many by media, how many by the globally accessible share, and how many documents were found by attackers on media inside the office.

When no guidance — the Neutral policy — is used, all of the employees choose to use the globally accessible share; when employees are instructed that Media is preferable, a number of them do choose to use that method, but a larger number still use the global share; when employees are instructed to use email, the majority of them do so, but some still use the global share. In the model, the global share is the fastest and therefore most productive choice. Email is not as productive as the global share, but is still faster than using portable media. This explains why some employees still use the global share even when they are instructed that an alternative method is more secure: those employees have a much higher preference for productivity than security.

Which of the three policy choices is best can be determined by looking at the security manager's expected utility [14, 12] for each choice:

$$E[Utility] = E\left[\sum_{a \in A} w_a f_a(v_a(\Phi_a) - \bar{v}_a)\right]$$

The expected value for each attribute depends on the stochastic process Φ_a, which is derived from the sequences of decisions D. In theory, this can be calculated using queueing theory and the knowledge about the stochastic elements of the model and environment; in practice, this is extremely difficult. Instead, we can approximate this value by executing the model a large number of times. As the number of executions increases, the accuracy of this approximation increases.

For this example, let's assume that the security manager cares about only two attributes in this model: the number of documents put into the global share, and the number of documents found by attackers inside the office. The manager would like to reduce the number of documents in the global share to about half of its current value, so sets a target of $\bar{v}_{global} = 70$; the target value for the number of documents found by an attacker is $\bar{v}_{found} = 0$. The manager is concerned about missing the target for the global share by a lot, but does not care to the same degree when beating the target; to represent this, we use a Linex function $f_{global}(x) = -(e^{0.01x} + 0.01x - 1)$ [15]. The manager cares linearly when beating the target

		Documents				
Policy	Guard	To Email	To Global	To Media	Found by Attacker	Lost
Neutral	0	0	175.5418	0	0	0
	1	0	175.5736	0	0	0
Media	0	0	108.4025	62.0687	0.3281	0
	1	0	108.5656	62.0678	0.149	0
Email	0	113.2898	49.3636	0	0	0.1
	1	112.7899	49.708	0	0	0.1112

Table 2: Results from the Composed model.

value, but exponentially when not meeting the target. Since the target for attackers finding documents is 0, we can use $f_{found}(x) = -e^x + 1$ for the other function. The manager is more concerned about the attackers finding documents than employees putting documents on the global share, so we use the weights $w_{global} = 0.5, w_{found} = 1.0$ for the relative importance of the two attributes.

Now that these values and functions have been specified, we can calculate the security manager's expected utility for each of the policy choices. The results are shown in Figure 4. The highest expected utility comes when employees are encouraged to use email to share the documents. Transferring the documents by portable media has higher expected utility than the neutral policy, but lags behind email because of the larger number of employees who still use the global share, and because of the documents that attackers find.

Next, let's consider this model when it is composed with the other two. The attackers are always present in the model as it is, and the security manager thinks that the access control at the entrance of the building could have an effect on how many documents they discover. Additionally, the security manager is concerned about employees losing devices containing documents they have downloaded from emails, and wants to know if email is really the best of the choices.

In this example, we'll only add one additional attribute to the utility calculations: the number of documents lost on devices outside of the office. There is also one more policy choice: whether or not a guard is present to watch for tailgaters at the entrance to the building. The results from the simulations are shown in Table 2. The presence of the guard changes the number of documents attackers find inside the office under the media policy, but has little effect on the other two policies.

For the utility calculations for the new attribute, the manager uses a target $\bar{v}_{lost} = 0$, the same function $f_{lost}(x) = -e^x + 1$ as the one for documents found inside the office, but with a greater weighting $w_{lost} = 1.5$ because the manager feels this is more severe. The values and functions for the other attributes from the document sharing model stay exactly the same. This new attribute is simply added to the utility calculations for each policy choice.

The new expected utilities can be seen in Figure 5. Once again, the highest expected utility is achieved by the email policy, followed by the media policy. The presence of the guard increases the expected utility value of the media policy, but not enough catch up to email. In this example, the security manager would feel confident suggesting that employees use email to send each other documents when the normal system is not working.

However, in reality, the security manager would be concerned with a greater number of attributes, and would likely have very different weightings for them. In the tailgating model, for instance, the security manager might also be concerned with attributes like the actual number of tailgating attempts by employees and attackers, how many of them are successful, how much time employees spend getting through security, or how much time they spend getting replacement badges from reception. The manager would also likely

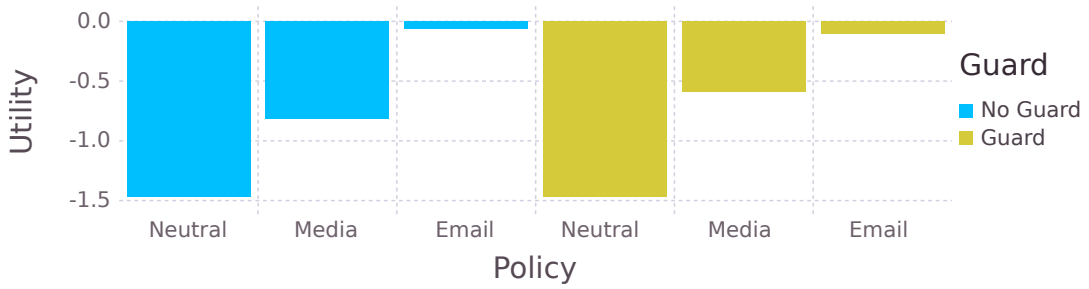

Figure 5: Security manager's utility for different policy choices in the composed model.

have greater concerns about documents sent by email: how safe are accounts from outside attack? Could documents be exposed that way? This could be addressed at a simple level by adding an attribute with a large weighting for the number of documents sent over email, representing the risk of this method, or in greater depth by composing these models with another that models the email system and its vulnerability to cyberattack.

7 Conclusions

For system managers, making decisions about security policy can be a challenging task because of the uncertainty that arises from the difficulty in predicting the consequences of specific policy choices. In this article, we have presented a methodology and framework for modelling systems security that provides a systematic approach for exploring the consequences of policy design decisions.

The models used in this methodology capture the aspects of the real-world system needed to evaluate security policy, including the physical and virtual locations in the system, its logical and technological components, and the behaviour of the agents interacting with the system. In this methodology, models of complex systems are constructed as compositions of smaller models of specific parts of the systems. The underlying mathematical foundation of the methodology ensures that the models soundly represent the systems of interest and that composition of models is also sound.

We have demonstrated our methodology with several examples that explain the construction of models, including their composition, the behaviour of agents, and the valuation of policies. These models illustrate how our methodology applies equally well to both cybersecurity and physical security, and that these can be intimately interconnected. Many extensions are possible. For example, the document-sharing model can be composed with a model of the organization's connection to the internet, so allowing us to explore the exposure of its confidential information deriving from attacks on its mail and web servers.

The process of building a model of a system helps in understanding the relationships between its physical, logical, behavioural, and policy components. Collecting the data that is required to parametrize the model can give insight into the current performance of the system and the behaviour of agents within it. Once these steps have been completed, the model itself allows reasoning about and experiment with alternatives, giving predictions about how the system will perform under different circumstances and under different policies. This ability to think about systems and their security policies in a structured way, based on gathered data,

moves the decision-making process away from guesswork and towards a more scientific approach.

References

[1] A. Beautement et al.. Modelling the Human and Technological Costs and Benefits of USB Memory Stick Security. In *Managing Information Risk and the Economics of Security*, M. Eric Johnson (editor), Springer, 2008. 141–163.

[2] A. Beautement, M. Angela Sasse, and M. Wonham. The Compliance Budget. Proceedings of the 2008 workshop on New Security Paradigms Workshop (NSPW '08), 47–58. ACM New York, NY, USA, 2008. ISBN: 978-1-60558-341-9. doi: 10.1145/1595676.1595684

[3] Y. Beres, D. Pym, and S. Shiu. Decision Support for Systems Security Investment. In *Proc. Business-driven IT Management (BDIM)*, IEEE Xplore, 2010.

[4] T. Caulfield and D. Pym. Modelling and Simulating Systems Security Policy. Forthcoming, *Proc. SIMUTools*, Athens, 2015. ACM Digital Library.

[5] M. Collinson, B. Monahan, and D. Pym. *A Discipline of Mathematical Systems Modelling*. College Publications, 2012.

[6] George Coulouris, Jean Dollimore, and Tim Kindberg. *Distributed Systems: Concepts and Design*. Addison Wesley; 3rd edition, 2000.

[7] L.A. Gordon and M.P. Loeb. *Managing Cybersecurity Resources: A Cost-Benefit Analysis*. McGraw Hill, 2006.

[8] D.F. Heathfield. *Production Functions*. Macmillan Press, 1971.

[9] C. Ioannidis, D. Pym, and J. Williams. Information security trade-offs and optimal patching policies. *European Journal of Operational Research*, 216(2):434–444, 2011.

[10] R. Jain. *The Art of Computer Systems Performance Analysis*. Wiley and Sons, 1991.

[11] julia. `http://julialang.org`.

[12] R.L. Keeney and H. Raiffa. *Decisions with multiple objectives*. Wiley, 1976.

[13] R. Milner. Calculi for synchrony and asynchrony. *Theoret. Comp. Sci.*, 25(3):267–310, 1983.

[14] H. Varian. Intermediate Microeconomics. W.R. Norton & Company, 2014

[15] A. Zellner. Bayesian prediction and estimation using asymmetric loss functions. *Journal of the American Statistical Association*, 81:446–451, 1986.