Modeling the "Tragedy of the Commons" Archetype in Enterprise Computer Security

Shalom N. Rosenfeld[1], Ioana Rus[2] and Michel Cukier[3]

[1]Department of Electrical and Computer Engineering, University of Maryland at College Park,
College Park, Maryland 20742, USA
shalom@umd.edu

[2]Fraunhofer USA Center for Experimental Software Engineering Maryland,
4321 Hartwick Road Suite 500, College Park, Maryland 20742, USA
irus@computer.org

[3]Center for Risk and Reliability, Department of Mechanical Engineering,
University of Maryland at College Park, College Park, Maryland 20742, USA
mcukier@umd.edu

Abstract: The purpose of this study is to understand observed behavior and to diagnose and find solutions to issues encountered in organizational computer security using a systemic approach, namely system archetypes. In this paper we show the feasibility of archetypes application and the benefits of simulation. We developed a model and simulation of some aspects of security based on system dynamics principles. The system dynamics simulation model can be used in support of decision-making, training, and teaching regarding the mitigation of computer security risks. In this paper, we describe the archetype "Tragedy of the Commons", in which an organization's efforts at improvements fail to consider the consumption of a shared resource, and we show the relevance of this archetype in the context of security. We describe a scenario where this archetype can help in diagnosis and understanding, and present simulation of "what-if" scenarios suggesting how an organization might remedy problems observed and maximize its gains from security efforts.

Keywords: software and computer system security, attacks, countermeasures, human factors, system dynamics, system archetypes, system and process modeling and simulation.

1. Introduction

All companies who use computer systems intensively must protect the security properties of their assets against malicious actions. They must employ various countermeasures to mitigate the risk of attacks, including various actions for reduction of vulnerabilities, as well as for detection of attacks and tolerance of intrusions. Identifying the security risks and knowing what is the most effective and efficient combination of countermeasures are very difficult tasks, because an organizational computer system is complex, with many actors and interactions and an inherent uncertainty and unpredictability. In addition, there are resource constraints, as well as trade-offs between security on one hand and other operational properties or business goals on the other hand.

To support the challenging decision-making process of designing an appropriate security strategy, we developed a quantitative executable model of an organization's operational computer security. Like all models, this is an abstraction of the real system, focused on representing the security-significant aspects of the system and associated processes. The model targets and represents the perspective of the person who must make decisions regarding actions to be taken for security assurance and security-related risk mitigation. The user of the model will first identify the underlying security problem that causes an unwanted observed behavior. Then, he/she can set different values for the model parameters, corresponding to different system usage, vulnerabilities, attacks, and defense profiles. The simulator can be run and different "what-if" scenarios can be executed. Simulation will help a security manager, security engineer, or system administrator answer questions such as: if my environment is characterized by these values, then what methods and tools should be selected and applied to manage security risks and satisfy the users' needs of my system? How will the selected actions work together? What is their effectiveness and cost efficiency? To what changes is my environment most sensitive? If I make specific changes in my security strategy, what will be their impact? What changes if my system gets attacked more/less or if the time to exploit changes? Should I hire more system administrators? Should I spend more on training them?

We propose using modeling and simulation of aspects of organizational computer security from a system's perspective, using the systems dynamics approach described by [1]. We take into account that control actions and reactions on any side of this system might have not only a local effect, but could also affect the rest of the system, often resulting in feedback loops. These effects manifest over time with different delays. The properties of the system (security being one of them) will emerge from its structure and all the interactions between its components. We show how archetypes (or patterns of behavior) can bring a systems perspective towards studying an organization's security aspects. The model aims first at understanding security risk reduction in computer systems, then at diagnosing such systems and identifying their weaknesses, as well as prospectively examining the effectiveness of different solutions.

Received July 10, 2007

This paper is structured as follows. Section 2 discusses the notion of archetypes in computer security, exemplifying with one possible instance of the "Tragedy of the Commons" archetype. Section 3 describes how a simulation model can be used to illustrate the archetypal behavior. Section 4 describes the use of simulation for diagnosis, analysis, and comparison of alternative solutions. Section 5 presents related work, and Section 6 concludes this study.

2. Archetypes and the "Tragedy of the Commons"

2.1 Archetypes

Archetypes are a concept related to systems thinking, developed in the mid 1980s in an attempt to describe complex behavior and to convey ideas in an easier, more efficient manner. Archetypes are *frequently-observed patterns of systems behavior* and are a "natural vehicle for clarifying and testing mental models" about systems or situations [2]. The systems literature describes ten distinct archetypes, as listed in [3]. In fact, [4] argues that all of these can be categorized into one of four "core generic" archetype classes (*Underachievement*, *Relative Achievement*, *Out-of-Control*, and *Relative Control*), but then acknowledges that the more common description of archetypes, i.e. that of [2], [3], and [5], is more intuitive and easier to grasp and apply to simulation, so it is used here. (We thus speak of the archetypes "Tragedy of the Commons" and "Limits to Growth", rather than the class *Underachievement*, for example; or "Shifting the Burden" and "Eroding Goals" vs. the *Out-of-Control* class.) For the most part, archetypes have been applied in business or industrial processes. There has recently been some work performed at MIT regarding systems thinking and archetypes in systems safety [6], but in computer security this is a new idea.

Here, we use archetypes for understanding and modeling security aspects (needs, problems, actions) in the context of an enterprise that uses computers/information technology systems for running its business and needs to ensure the security of its systems. Our focus in this paper is on the application of the common archetypes to computer security; however, an archetype is only useful as it lends insight to a particular situation. In approaching particular scenarios in computer security, we keep in mind that other, specialized archetypes may be discovered here. This would not be surprising, as [6] uncovered several field-specific archetypes in industrial safety. In this paper, however, we limit ourselves to the archetypes most commonly discussed in the literature, in particular those described by [3].

In this paper we present one archetype that we modeled, namely "Tragedy of the Commons." In [7], [8], and [9], we have previously considered the following archetypes vis-à-vis computer security: "Escalation", "Shifting the Burden", and "Escalation" combined with "Limits to Growth", respectively.

2.2 "Tragedy of the Commons"

The concept of a "Tragedy of the Commons" was originally proposed by [10] over three decades ago; it has since proven useful in many other fields and been incorporated into the systems and archetype literature (see [2], [3], [4], and [5]). In this archetype, the "Commons" are an uncontrolled resource available to many users, who can use (or "consume") it without each other's knowledge or permission. The Commons can accommodate up to a certain capacity without loss of yield for its consumers. The "Tragedy" occurs when this capacity is exceeded; this can happen in two ways. In the first, each user consumes a small portion of the Commons, but the number of users eventually exceeds the Commons' optimal capacity. ([2] gives the example of too many cars limiting the flow of traffic on a highway.) In the second, there are few users, but in time each user begins to consume more and more of the Commons. (See [2] as well for an example regarding several automobile designers, each making increasing demands on the car battery.) In either case, until the optimal capacity is exceeded, additional consumption or use of the Commons leads to additional yields. Beyond this point of consumption, however, the yields begin to lessen. Each individual user of the Commons does not understand why their gains per effort are no longer as high, and may try to compensate by raising their efforts, further increasing consumption of the Commons and worsening the Tragedy. This may culminate in a total destruction of the Commons, or simply a peak and then decline in the gains from the Commons.

As described in [4], a partial solution to this Tragedy occurs if the Commons capacity can be raised. However, once the Commons capacity is fixed to some finite value, consumption will increase until a Tragedy occurs. Thus, the only true solution to this Tragedy is if the system is viewed as a whole, identifying the Commons, its limits, and its consumers. At that point, either the consumers must come to some agreement among themselves regarding proper use of the Commons, or a superior must place restrictions on its use.

The notion of human resources as Commons is not entirely new; in fact, [3] cites the example of one information technology department being "shared" by two different divisions of a company. Similarly, we describe a company's computer support staff as its Commons.

2.3 Illustrated Behavior

For a simple but plausible illustration in computer security, we paint a scenario in which a hypothetical company's computer system (or just "system") is continually falling prey to successful simple attacks, known as "kiddy-scripts." These attacks are launched by novice attackers (or "script-kiddies"), and generally only succeed if the system contains vulnerabilities such as software that is not up-to-date or an inadequate security policy.

Suppose that upon investigating the problem, the company's technology-related management realizes that it has neglected having software patches applied on a regular basis. (In this scenario, we assume that the software has

Received July 10, 2007

been completely unpatched until the beginning of the run, so additional patching will help significantly, but ignoring the patches will not lead to any further harm.) We suppose that these managers now demand that the staff begin spending X hours per day on patches. On finding this move yields significant gains, the demand on the support staff will be increased to X+1 within the next few days. This continues, with additional gains leading to additional efforts/demands/consumption, and so on. There certainly does reach a saturation point beyond which further allocation of time for patching will accomplish nothing, but this is again beyond the scope of our illustration of this archetype, so we assume this point is not reached during our time period. Alternatively, the management may conclude that significant increases in security are available if the support staff does a good job of system administration and enforcing the company's security policy, as described in [11]. These tasks include, to name a few: scanning for and fixing configuration vulnerabilities, which are effectively "doors" to the system that were inadvertently left open; applying proper access control to prevent unauthorized use; and monitoring the users to prevent them from unsafe practices such as downloading viruses and using "weak" passwords which are easily guessed. We group all of these tasks and all others that require no additional software or hardware *per se,* only a great deal of attention paid by the support staff (or system administrators) to what is already in place under the title of "enforcement actions." We thus suppose that the management initially demands Y hours per day of enforcement actions from the support staff. In several days' time, the managers find that the system's security has risen, and thus the demand is increased to Y+2 hours per day. Again, this leads to added performance, leading to added demands/efforts.

A third possibility is that the management decides to begin demanding both patches and enforcement actions, in quantities equal to the sum of the previous two cases. Meanwhile, the hapless support staff has a finite capacity of hours in the day it can spend on any given task. Well before its absolute maximum capacity however, there reaches a point where it is stretched beyond *optimal* capacity. Once demands beyond optimal capacity are placed on the support staff, especially if the demands are being made for different goals, the support staff is stressed and must now spend a nontrivial amount of time answering the non-stop requests from each directive from management, demanding its immediate attention. Thus, the gains it can provide a particular demand begin to decrease.

Throughout the systems literature, the possibility is acknowledged that the Commons could be consumed to the point of its total collapse, such as complete depletion of a natural resource; the analogy here would be the support staff's becoming disgruntled and ultimately resigning. For now, however, we assume the support staff, even if pushed beyond peak efficiency, is still operating in a "normal" mode and providing some gains.

The management does not realize that the *Commons* of the support staff is not capable of sustaining all of the demands placed upon it, and are thus puzzled when the increases in security are not as steep as they had once been; the system's security may even level off and decline. These effects are displayed in an influence diagram in Figure 1.

In the upper-center loop, increased efforts (i.e. demanding more staff-hours per day from the support staff) for patches cause increased gains in security from patches, increasing the motivation for additional efforts. Similarly, in the loop beneath it, increased efforts for enforcement actions produce gains in security, encouraging further efforts/demands. Each of these two loops, if viewed independently, would be described as reinforcing loops, which should increase indefinitely if nothing else influences the system.

Unfortunately, something else does indeed influence the system, and that is represented by the arcs to the left of the main loops: all efforts contribute to a rise in total activity, which depends on additional use of the Commons (in our case, the support staff); while there may be some delay (double hatch mark), ultimately the demands on the support staff beyond its optimal capacity will results in a reduced gain per individual activity (i.e. a given demand for more enforcement or patches).

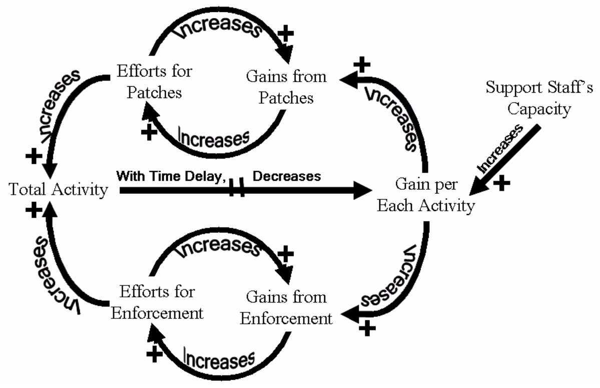

Figure 1. Influence diagram for a "Tragedy of the Commons" situation

Of course, if the support staff has a high capacity, it will take much longer for the point of optimality to be surpassed, and thus the gains per effort can remain higher for longer. Another pattern observable, therefore is as follows: increased efforts add to total activity, which reduces the gain per individual activity if the support staff's optimal capacity has been exceeded; this decreases the gain from enforcement and/or patches. At some point, it is believed that diminishing gains will lead to reduced efforts: this forms a "balancing loop", which will push towards the equilibrium of efforts matching the support staff's optimal capacity.

3. Proposed Model

To consider the quantitative strengths of the various influences in the above archetypes, and to see the results of changes and possible prognoses, computer simulation is used, as is recommended by [2]. The continuous simulation technique of the *Extend* simulation environment [12] was used. An overview of Extend can be found in [13], where Extend is described as providing "an integrated structure for building simulation models and developing news simulation

Received July 10, 2007

tools." Moreover, "model builders can user Extend's pre-built modeling components to quickly build and analyze systems without programming." The description provided here of our simulation model is excerpted from [14], which contains full details. The entities modeled consist mostly of those mentioned above: "attempted attacks", "staff size", "security-related efforts", "countermeasures and vulnerabilities", with human factors related to the management (or decision-makers regarding security investments), the support staff, and the attackers. As the model is applied to different specific contexts, this will allow data from those systems to contribute to tailoring and increasing the accuracy of the model. Using the model will reveal the data needs for executing it for a given system, thus guiding security measurement. For our current numerical values, we have incorporated empirical findings such as:

- The cyber security bulletins from [15]. For instance, in the first two years since its release, Microsoft Windows 2000 Server averaged 0.09 bulletins per day at the *Critical* or *Important* level. For Windows 2003 Server, the average was 0.04 per day. We used an average of these two values to describe the rate at which an unpatched system becomes steadily more vulnerable to attack.
- The FBI's annual computer security survey [16]. For example, out of the 600+ corporate, academic, and governmental organizations responding to the poll, 65% observed a virus on their systems last year, but only 25% observed a denial-of-service attack, making the former significantly more prevalent. It is thus assumed that between a countermeasure designed specifically against viruses and one designed specifically against denial-of-service attacks, the former will defeat a larger number of attacks typically seen on the Internet.
- "Honeypot" or "honeynet" research and analysis (such as [17] and [18]), which gives us some idea of how many attacks per day tend to be attempted against an average system.

For data unavailable in the literature, experts' judgment was used in assigning values. Thus, given these parameters, the current numerical results of the simulation reflect a combination of literature data where available and well-educated estimates where data were not available. Further benefit resides in the *overall trends* in behavior displayed by the simulation, which are used for problem-solving below.

3.1 Model Description

To see some quantitative results of the above "Tragedy of the Commons" scenario, an Extend model was built simulating a system containing on the order of 200 machines, sustaining 100 simple attacks per day. A certain percentage of simple attacks are modeled as exploiting vulnerabilities in the system's software; the better-patched the software has been, the less of these will succeed. Similarly, a certain percentage of simple attacks are designed to exploit user mistakes, poorly configured servers, and the like; therefore, a certain percentage of attacks will not succeed if proper enforcement actions are in place. Note that even if our simulation indicates "40% of the attempted attacks succeeded", the system's users may not observe for 100 attempted attacks, 40 separate failures, as many of these attempts might target a small set of specific vulnerabilities and exploit them in the same way. Similarly, no single task (or even patching combined with enforcement actions) should be expected to reduce the attack success rate to 0 by itself, as there are enough different types of attacks that any single security action or countermeasure can be defeated. (In real life, therefore, an organization would be wise to augment its patches and enforcement actions with countermeasures such as a firewall or antivirus software, but they are omitted in this archetype demonstration.) We use the percentage of successful attacks only as a measure of the system's vulnerabilities. For a given execution of the simulation, we specify some rule of how many staff-hours per day of efforts are demanded for each security-related task (such as applying software patches or enforcement actions). Given the optimal capacity of the support staff (as specified in the model), these demands are translated into actual staff efforts for each task, whereupon the staff efforts determine the quality (or lack-of-vulnerability) of the system's software and configuration. These daily measures of quality determine what percentage of each day's attempted attacks do not succeed; the remaining attacks are deemed "successful", and their tally is viewed as the model's output. Analyzing this output of "successful attacks per day" can give us a greater understanding of what the (often unforeseen) effects of our input choices have been; it also allows us to compare alternative scenarios run with different input values. The model was executed for the equivalent of 6 months (real time) with different scenarios. (Each execution of this type runs in under 30 seconds.)

3.2 Model Behavior

In the first scenario simulated, patches alone are used; a certain number x_i of staff hours are demanded initially for patches, and that number rises linearly with time, up to x_f staff hours at day 180, where x_f is still less than the optimal capacity of the support staff. (This is a quantitative description of behavior we assume to be realistic.) In the second scenario, enforcement actions alone are used, with an initial demand of y_i staff hours for enforcement actions, increasing linearly over 180 days to y_f. Again, at no point do the demands for enforcement actions exceed the support staff's optimal threshold. In the third scenario, the two previous scenarios are implemented simultaneously, beginning with net demand (i.e. staff-hours demanded for all tasks) $z_i = x_i + y_i$ and ending with demand $z_f = x_f + y_f$. We have used the values $x_i = 0.84$, $y_i = 1.2$, $z_i = 2.0$, and $x_f = 6.6$, $y_f = 11$, $z_f = 17.6$, believing these to be realistic descriptions of a transition from very modest security efforts vis-à-vis patches and/or enforcement actions to a full effort. To measure the effectiveness of efforts in the first two scenarios, we measure percentage of attacks *unsuccessful.* Figure 2 shows the results of scenarios one (*patches*) and two (*enforcement actions*), each one executed by itself. Note

Received July 10, 2007

that in our current model, effective software patching can lead to many more unsuccessful attacks than can effective enforcement actions; the goal of this archetype description is not to compare one against the other, but rather to note the overall shape of each trend and see what happens when they are combined.

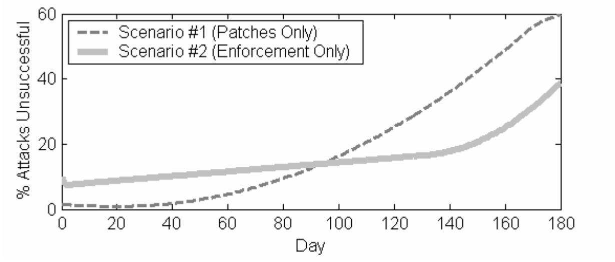

Figure 2. Percentage of attacks unsuccessful for the two countermeasures used individually

Figure 3 shows the results of scenario three, in which both efforts are made (and increased) simultaneously, without regard for the optimal output of the support staff. Thus, the total burden on the sysadmin is larger than it had been in each scenario of Figure 2.

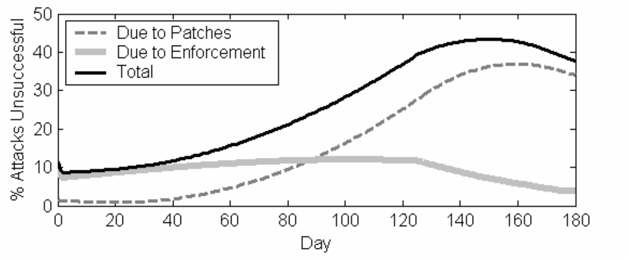

Figure 3. Percentage of unsuccessful attacks, both *Patches* and *Enforcement* used simultaneously

In scenario three, at some point (actually around day 90 here, though the effects are not felt for several more weeks), the optimal capacity of the support staff is slowly exceeded. Additional efforts are still made, but the support staff demand-beyond-optimal-capacity effect is engaged, reducing the number of effective staff-hours supplied to patches and enforcement. As a result, the gains from each begin to peak and then decline. For *enforcement*, which did not possess very aggressive growth when implemented independently, the percentage of unsuccessful attacks plateaus, and then descends quite rapidly. For *patches*, which witnessed a sharp increase in success over the last six months when run alone, the plateau and fall do not come until almost the end of the simulation (day 158), but they come nonetheless.

4. Discussion and Other Scenarios

4.1 Lessons and Alternate Scenarios for "Tragedy of the Commons"

From Figure 3, it can be learned that the effects of a combination of security-related tasks cannot easily be predicted only from the history of each one used in isolation. There must be one managing body considering all the demands being made on the *Commons*, then developing and enforcing a policy that leads to maximum overall gains from its use; such a policy will involve limiting consumption.

Armed with this knowledge, a manager can then ask what the optimal allocation of the *Commons* will be, and again simulation can help provide an answer. For any proposed "what-if" course of action, the model can be re-run with the appropriate change in input values; the (possibly counterintuitive) results of each proposed solution can then be studied to reduce risk before any changes are actually implemented in the real-world computer system. If the above scenario three represented our "problem" or "baseline" scenario, we consider "solution" scenarios four, five, and six. In these latter scenarios, the ratio of demand between patches and enforcement actions is modified for improved performance. The solution scenarios differ in how to manage the quantity of total effort demanded.

Scenario four is that of on "optimal" solution: the manager has near-perfect knowledge of the system, makes no additional demands once the optimal staff load has been reached, and allocates hours between the two tasks with this in mind.

Of course, the optimal scenario four assumed that this magical value of "optimal staff load" is known. More realistically, scenario five ("one day delay") acknowledges that it is not known in advance, but a manager keeping a close watch of the results (i.e. unsuccessful attacks) could implement the following (somewhat counterintuitive) rule: *if, at any day, a decline in unsuccessful attacks is seen, from the next day on, make no further demands, i.e. leave the staff demands at their current levels.* In the language of [4], any combination of several "mental barriers" might prevent a rule such as this from being implemented: the manager may not understand that the decline is being caused by excess demands; there may be an instinctive fear response to increase demands when problems appear; the decline may be too small to be observable; and the manager might not change course because he/she refuses to acknowledge the decline, rationalizing that "it was just one bad day." Given all of these difficulties, a more likely response is that of scenario six, where a delay greater than one day is present before additional demands are halted, i.e. *if, at any day, unsuccessful attacks are less today than they were θ days ago, then make no further demands from the next day on.* We consider $\theta = 3$ to be reasonably optimistic.

We simplify by plotting only the total percentage of attacks unsuccessful for each scenario. The results of scenarios "baseline", "optimal", "one day delay", and "three day delay" are shown in Figure 4.

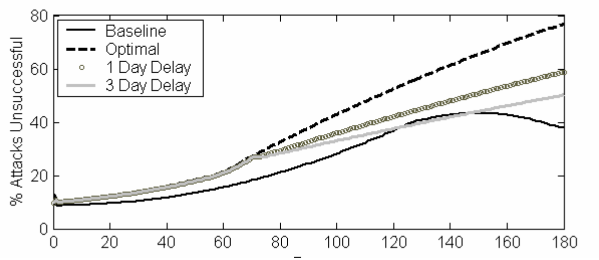

Figure 4. Results of "What-if" scenarios

Received July 10, 2007

Several trends can be observed here. Firstly, all of the scenarios other than baseline avoid the Tragedy to a large degree and thus do not experience a decrease towards the end of the simulation period. Secondly, note that the three alternative scenarios (optimal, one day delay, three day delay) all involve ceasing further demands by approximately day 90, and yet gains continue to rise: once the company begins doing a good job with patches and enforcement, a constant effort will increase gains over time. A saturation point will most probably be reached at some point in the future, but that is not considered here; a description of that situation is addressed by the “Limits to Growth” archetype described in [9]. Thirdly, it is not surprising that the longer the delay to action, the less the plot will look like “optimal” and the more it will look like “baseline.” Lastly, even in the “optimal” scenario, the greatest gain reached is approximately 80% attacks unsuccessful. While this value is impressive, we might also turn back to Figure 2, which considered the gains of using each task individually. If we were to sum the two plots of Figure 2 at day 180, we would arrive at a supremum of over 90%. Indeed, if the *Commons* were infinite, then the gains due to two tasks performed simultaneously would be equal to the sum of each task’s individual gains. Realistically, however, the best we can do is to acknowledge the *Commons* as finite and optimize accordingly; this is reflected in the “optimal” 80% of Figure 4.

Here we have presented only one instance of “Tragedy of the Commons” as it can occur on the enterprise security level. This archetype has also been used in other fields of computing, for example, resource allocation between processes in centralized system design. Another instance on the enterprise level that may be modeled in future work is that of common memory, processing power and the like, where security-related tasks such as encryption and virus scanning can consume the *Commons*.

4.2 Applying Archetypes and Simulation

We have thus shown the feasibility of using known system archetypes for explaining situations that can occur within an organization with respect to its computer security. In our modeling of such situations, both the system’s technological aspects and human factors are taken into account. Each archetype describes a "story", explaining symptoms that can be observed while examining relevant variables (in our case, for example, the “attacks unsuccessful" served as an indicator of a chosen security strategy’s effectiveness). While one archetype alone may not depict a broad picture of the entire system’s behavior, a combination of archetypes can do so. Depending on a specific organization’s characteristics, a particular known archetype may or may not present itself. Furthermore, besides the generic system archetypes referenced in this paper, security-specific archetypes (as mentioned in the introduction to Section 2) may be identified in a given organization. It is up to the analyst of each given system to determine what lessons of system dynamics are applicable to his/her concrete situation. Here, the paradigm of system thinking and archetypes is offered as a tool to analysts who may include, for example, security engineers, managers, and security policy makers. The other tool proposed here is that of simulation: once archetypes are used to identify the cause of an undesirable outcome, simulation can then be used to consider the results of different proposed solutions. Used in this way, simulation can thus support decision making in selecting and implementing the most effective security strategy; by modeling dollar-value costs for phenomena such as successful attacks or staff-hours of security labor (see [9] and [14]), the most cost-efficient strategy can be determined and chosen as well.

4.3 Simulating Other Organizational Contexts

The scenarios presented above were executed for a hypothetical organization. For practical use of the simulator, it has to be run in the context specific to the organization under study. For making the model and the simulator applicable to different contexts, we parameterized the model and the simulator. The context can be varied by changing the values for these parameters. The parameters reflect the profile of company, system size (i.e. number of machines), system vulnerability factor, the attack profile, and the countermeasure strategy.

In addition to the context parameters, the model and simulator also have another set of parameters, namely input parameters, whose values are varied for executing different scenarios for a chosen context, corresponding to different candidate solutions. Examples of such input parameters are staff size, delays for making changes, and change quantities.

For executing the simulator for a specific organization and system, the simulator has to be calibrated first to that specific environment (by setting the context parameters to specific values). Different scenarios can be then executed by setting the values for the input parameters. The *Extend* simulation environment that we used provides features such as sensitivity analysis, for one or more parameters at a time.

As described in [14], the current version of the simulator has a graphical user interface containing sliders and switches through which the user of the simulator can set the desired values of the input parameters. The values of the context parameters are set (and can be adjusted) in an Excel spreadsheet that is automatically read at the beginning of the simulation. The outputs of the simulation from one run or multiple consecutive runs can be recorded as graphs (as shown in figures above), or exported to Excel and processed after the simulation. They can be processed for example for comparing outputs of different scenarios using multiple decision variables (e.g., number of attacks and effort/cost invested in security) and supporting decisions regarding the most desirable security strategy, according to the criteria established for the organization under study.

The value of the simulation’s output depends on the quality of its inputs. For the illustration in this paper we used values taken from the literature and experts’ opinions. To obtain maximum gain from simulation, a company must have its own specific values with which to run the simulation. The list of parameters of the model can constitute a start for defining what the organization should collect as part of their security measurement activity.

5. Related Work

5.1 Security Validation

Traditional, non-quantitative approaches to security validation, such as [19] and [20], have focused on prescribing procedures for system design. Actually, in selecting the entities to be represented in our model, we followed the security concepts and relationships from [20], and then extended it with dynamic, behavioral, and quantitative aspects.

Where quantitative methods have been used, they have either been quite formal, such as [21], proving how certain security properties hold given a specified set of assumptions; quite informal, using a "red team" of experts (e.g. [22]) trying to compromise a system; or risk-oriented, assessing the risks to which an organization is exposed (e.g. [23]). Risk assessment is also included in [24], which describes guidelines for initiating, implementing, maintaining, and improving information security management in an organization or in information security governance that delivers certifications like the Certified Information Security Manager (CISM) certification. An alternative approach has been to try to probabilistically quantify an attacker's behavior and its impact on a system's ability to provide certain security-related properties. Attempts have been made to build models that take into account both the attacker and the system being validated. A general 9-state model of an intrusion-tolerant system is proposed by [25] to describe security exploits by considering attack impacts; the system state is represented in terms of failure-causing events.

[26] proposes a combination of state-level modeling, formal logic, and Bayesian analysis to quantify system survivability. The authors first model the network nodes and links of a networked system using state machines, and then faults are injected in the models. The third step consists of specifying a survivability property, e.g., the system enters a faulty state, using a temporal logic. This is used to generate a scenario graph, which is then used for evaluating the overall system reliability or the latency using Bayesian networks. [27] proposes modeling known vulnerabilities in a system using a "privilege graph" similar to the scenario graph described above. By combining a privilege graph with assumptions concerning an attacker's behavior, the authors then obtain an "attack state graph." Parameter values for such a graph have been obtained empirically; once obtained, an attack state graph can be analyzed using standard Markov techniques to obtain probabilistic measures of security. [28] uses a probabilistic model for validating an intrusion-tolerant system that combined intrusion tolerance and security, allowing the designers to make choices that maximize the intrusion tolerance before they implement the system.

The presented models vary from high-level models with 9 states [25] to much more detailed models [26], [27], [28]. The model that we have presented has an intermediate level of abstraction. Our model is also more generic in its inclusion of other human elements like users and system administrators. Finally, with the exception of [27]'s model, which uses some data collected experimentally to assess certain parameter values, the other models are not developed to easily be linked to empirical data.

5.2 Related Security Simulators

Cyberciege [29], [30], developed by the Naval Postgraduate School, is a computer game with a very engaging user interface and virtual world, intended for training students to understand security engineering. Cyberciege focuses on detailed access control, user-by-user, for a small number of users. Each piece of hardware is hand-selected from a list of fictional brands, and physical security measures are then implemented on a user-by-user basis. The determination of whether an attack succeeds is by comparing asset desirability and how well standard procedures have been followed. Cyberciege's level of detail models the role of an individual security officer who might oversee a dozen computers at most, while our model abstracts one level higher, to the manager who oversees several hundred machines. Many of the tasks we include under "enforcement actions", for example, are performed in detail in Cyberciege.

In a similar vein, [31] describes a security simulator available on their website (http://all.net/games/index.html). This simulator gives examples of how a single attack of varying sophistication might succeed against different computers with different countermeasures. The defender strength, i.e. to what degree the defender does the right thing, is specified as a percentage by the user before running the simulator. If an attack succeeds, the dollar loss due to the attack is estimated based on the attacker profile, e.g. how much will a successful attack by a private investigator cost? Our approach attempts to add in more empirical data. Additionally, our work extends the "defender strength" idea by allowing for strengths of each countermeasure: a system may have a 90% effective firewall but only a 70% effective IDS. Furthermore, rather than specify a value for defender strength, the user of our model inputs managerial decisions such as how much effort is allocated to which security tasks and how skilled the staff is – the model then uses these inputs to determine the resulting defense strength for each countermeasure.

5.3 Related Empirical Studies

For sources of empirical data, we have security-incident anecdotes such as [32] and corporate surveys such as [16]. Hypothesized behavior of an individual attacker is empirically described by [33], but more data are needed on describing the aggregated effect of multiple attackers. Therefore, most data on attacks are gathered from analyzing "honeypots" or "honeynets", systems designed to be attacked. Such studies include [17], [34], [35], and [36], as well as our own laboratory's [18] and [37]. A great deal of this work is ongoing and will continue to yield additional empirical data.

To help meet the dearth of empirical data regarding security, nine teams are collaborating on the projects DETER and EMIST [38]. DETER involves building a massive (currently approximately 200 machines, intended to reach 1000 machines) "researcher- and vendor-neutral" network testbed for emulating various types of attacks, countermeasures, and network topologies. Meanwhile, the EMIST project seeks to formalize methodologies for measuring these effects. Combined, these projects should provide a wealth of useful, unbiased, and well-accepted emulated attack data. Both studies will enrich our model with quantifiable values, e.g. honeynet findings might show that 20 buffer-overflow attacks of a certain type are attempted each day, and the DETER/EMIST findings would tell us that the attack will succeed 80% of the time if the network has Topology A but only 60% of the time with Topology B.

Regarding user factors, [39] uses surveys to understand Internet usage, and [40] conducts studies with test websites to investigate users' privacy behaviors online. The authors of these papers have indicated that their future work will analyze user behavior regarding network security, which should be applicable to our user model.

As far as the interaction of economics and computer security, [41] considers the effects of public disclosure regarding security breaches on a company's stock prices. [42], [43], and [44] use economic analysis in determining how much security investment is worthwhile for a company, given its priorities; however, details are not provided as to what should be done specifically with the investments. This provides the connection point to our model.

Economic requirements are also used to lead to assumptions or specifications for related computer security, e.g. determining the subjective cost and total welfare regarding network routing [45] or requirements on trusted platforms placed by digital rights management [46], [47].

6. Conclusions

We have shown in this paper how a system archetype can be used in computer security to aid understanding and diagnosis, and making decisions for risk mitigation. Moreover, systems thinking, combined with simulation, can assist an organization in placing its efforts in the places that will give the most "leverage" to their goals, and in diagnosing and solving problems. We have specifically shown that even if the results are known for two different courses of action, combining the actions can produce vastly different results.

System dynamics simulation is thus a very valuable tool for enterprise computer security, and can be used in the following way:

An organization identifies the indicators whose evolution needs to be studied for assessing achievement of its goals ("attacks unsuccessful" in the example presented here) or existence of problems. If the trends of these indicators point to an archetypal behavior, then the simulation corresponding to that archetype (or archetypal combination, see [9]) is selected and the simulator is calibrated to the organizational context, as described in section 4.3. After the observed behavior is reproduced by simulation, different *what-if* scenarios will be executed in order to determine how to fix the problems identified by the archetypes. Different solutions will be simulated by varying the values of the simulation parameters. As with any simulation modeling, the results depend on the data used for developing and calibrating the model: the data used for model development were drawn from the current literature and experts' opinions, gathered by our interviews, for average information systems. The model is then tuned with calibration data provided by the individual organization using it.

The simulation model can also be used for training of students and less-experienced security personnel, who can simulate and analyze the effects of potential actions without affecting the real system. This *learning by doing* is enabled by the knowledge encoded in the model; thus the model serves as a tool for knowledge transfer from experts to the less-experienced.

In sum, the archetype and simulation results presented here show the value of systems dynamics modeling for enterprise security. Monitoring behavior for a long enough time allows observation of the genuine, long-term trends. Combined with a systems approach, this supports proper diagnosis, understanding, and solution-finding for observed problems. We illustrated the value of applying systems thinking in modeling the domain of enterprise security; the value of thinking in terms of links, delays, and feedback loops; and of keeping in mind that something happening locally in one part of the system might have an unexpected, possibly delayed, global effect. We also showed the benefits of simulation, which supports execution of "what-if" scenarios. The number values assumed in our discussions were for a hypothetical situation to illustrate some archetypal trends, but the patterns outlined here are expected to hold in other contexts as well.

Acknowledgment

This material is based upon work supported by the National Science Foundation under Grant No. 0237493.

References

[1] J. W. Forrester. *Industrial Dynamics*, The Wright-Allen Press, Cambridge, 1961.

[2] P. Senge, A. Kleiner, C. Roberts, R. Ross, and B. Smith. *The Fifth Discipline Fieldbook*, Doubleday, New York, 1994.

[3] W. Braun. "The System Archetypes", 2002. Available at: wwwu.uni-klu.ac.at/gossimit/pap/sd/wb_sysarch.pdf

[4] E. F. Wolstenholme. "Toward the Definition and Use of a Core Set of Archetypal Structures in System Dynamics", *System Dynamics Review*, vol. 19, no. 1, pp. 7-26, 2003.

Received July 10, 2007

[5] P. M. Senge. *The Fifth Discipline: The Art and Practice of the Learning Organization*, Doubleday Currency, New York, 1990.

[6] K. Marais and N. Leveson. "Archetypes for Organizational Safety". In *Proceedings of the Workshop on Investigation and Reporting of Incidents and Accidents (IRIA)*, 2003.

[7] S. N. Rosenfeld, I. Rus, and M. Cukier. "Modeling and Simulation of the Escalation Archetype in Computer Security". In *Proceedings of the 2006 Symposium on Simulation and Software Security (SSSS)*, 2006.

[8] S. N. Rosenfeld, I. Rus, and M. Cukier. "Modeling the Symptomatic Fixes Archetype in Computer Security". In *Proceedings of the 30th Annual International Computer Software and Applications Conference (COMPSAC)*, 2006.

[9] S. N. Rosenfeld, I. Rus, and M. Cukier. "Archetypal Behavior in Computer Security", to appear in *Journal of Systems and Software*, 2007.

[10] G. Hardin. "The Tragedy of the Commons". *Science*, vol. 162, pp. 1243-1248, 1968.

[11] D. Danchev. "Reducing 'Human Factor' Mistakes". WindowSecurity.com, 2003. www.windowsecurity.com/articles/Reducing_Human_Factor_Mistakes.html

[12] Imagine That, Inc. *Extend* (Version 6.07), CD-ROM, Windows 98/ME/NT4/2K/XP, San Jose, CA, 2005.

[13] D. Krahl. "The Extend Simulation Environment". In *Proceedings of the 2000 Winter Simulation Conference*, pp. 280-289, 2000.

[14] S. N. Rosenfeld. "System Dynamics Modeling and Simulation of Enterprise Computer Security". Master's thesis, University of Maryland at College Park, 2006.

[15] CERT (United States Computer Emergency Readiness Team). "Technical Cyber Security Alerts", 2007. http://www.us-cert.gov/cas/techalerts

[16] L. A. Gordon, M. P. Loeb, W. Lucyshyn, and R. Richardson. "Eleventh Annual CSI/FBI Computer Crime and Security Survey". Computer Security Institute, San Francisco, 2006.

[17] M. Dacier, F. Pouget, and H. Debar. "Honeypots: Practical Means to Validate Malicious Fault Assumptions". In *Proceedings of the 10th IEEE Pacific Rim International Symposium on Dependable Computing (PRDC)*, pp. 383-388, 2004.

[18] S. Panjwani, S. Tan, and K. Jarrin. "An Experimental Evaluation to Determine if Port Scans are Precursors to an Attack". In *Proceedings of the International Conference on Dependable Systems and Networks (DSN)*, Yokohama, Japan, pp. 602-611, 2005.

[19] U.S. Department of Defense. "Department of Defense Trusted Computer System Evaluation Criteria" ("Orange Book"). DOD 5200.28-STD, Library No. S225,7ll (November), 1985. http://www.radium.ncsc.mil/tpep/library/rainbow/5200.28-STD.html

[20] ISO/IEC. International Standards (IS) 15408-1:1999, 15408-2:1999, and 15408-3:1999, "Common Criteria for Information Technology Security Evaluation": Parts 1-3, Version 2.1, (August) (CCIMB-99-031, CCIMB-99-032, and CCIMB-99-033).

[21] C. Landwehr. "Formal Models for Computer Security", *Computer Surveys*, vol. 13, no. 3, September 1981.

[22] J. Lowry. "An Initial Foray into Understanding Adversary Planning and Courses of Action." In *Proceedings DARPA Information Survivability Conference and Exposition II (DISCEX)*, pp. 123-133, 2001.

[23] D. J. Landoll. *The Security Risk Assessment Handbook: A Complete Guide for Performing Security Risk Assessments*, Auerbach Publications, Boca Raton, Florida, 2006.

[24] ISO/IEC. "Information Technology – Security Techniques – Code of Practice for Information Security Management". *ISO/IEC 17799*, 2005.

[25] K. Goseva-Popstojanova, K. Vaidyanathan, K. Trivedi, F. Wang, R. Wang, F. Gong, and B. Muthusamy. "Characterizing Intrusion Tolerant Systems Using a State Transition Model". In *Proceedings of the DARPA Information Survivability Conference and Exposition II* (*DISCEX)*, 2001.

[26] S. Jha and J. M. Wing. "Survivability Analysis of Networked Systems". In *Proceedings of the 23rd International Conference on Software Engineering (ICSE)*, pp. 307-317, 2001.

[27] R. Ortalo, Y. Deswarte, and M. Kaaniche. "Experimenting with Quantitative Evaluation Tools for Monitoring Operational Security", *IEEE Transactions on Software Engineering*, vol. 25, no. 5, pp. 633-650, 1999.

[28] F. Stevens, T. Courtney, S. Singh, A. Agbaria, J. F. Meyer, W. H. Sanders, and P. Pal. "Model-Based Validation of an Intrusion-Tolerant Information System". In *Proceedings of the 23rd Symposium on Reliable Distributed Systems (SRDS)*, pp. 184-194, 2004.

[29] Naval Postgraduate School and Rivermind, Inc. Cyberciege, (Version 1.5b), 2006. See: http://cisr.nips.navy.mil/cyberciege/index.htm

[30] C. E. Irvine, M. F. Thompson, and K. Allen. "CyberCIEGE: Gaming for Information Assurance", *IEEE Security and Privacy Magazine*, 3(3), pp. 61-64, 2005.

[31] F. Cohen. "Simulating CyberAttacks, Defenses, and Consequences". Fred Cohen & Associates, Livermore, California, 1999.

[32] S. Gibson. "The Strange Tale of the Denial of Service Attacks Against GRC.com". Gibson Research Corporation, Laguna Hills, California, March 2002. www.grc.com/files/grcdos.pdf

[33] E. Jonsson and T. Olovsson. "A Quantitative Model of the Security Intrusion Process Based on Attacker Behavior", *IEE Transactions on Software Engineering* 23 (4), pp. 235-245, 1997.

[34] F. Pouget and M. Dacier. "Honeypot-Based Forensics". In *Proceedings of the AusCERT Information Technology Security Conference*, 2004.

[35] F. Pouget, M. Dacier, and V. H. Pham. "Understanding Threats: A Prerequisite to Enhance Survivability of Computing Systems". In *Proceedings of the International Infrastructure Survivability Workshop*

(IISW), in conjunction with 25th International Real-Time Systems Symposium (RTSS), 2004.

[36] F. Pouget, M. Dacier, and V. H. Pham. "Leurre.com: On the Advantages of Deploying a Large Scale Distributed Honeypot Platform". In *Proceedings of the E-Crime and Computer Conference (ECCE)*, 2005.

[37] R. Meyer and M. Cukier. "Assessing the Attack Threat due to IRC Channels". In *Proceedings of the International Conference on Dependable Systems and Networks (DSN)*, 2006.

[38] R. Bajcsy, T. Benzel, M. Bishop, B. Braden, C. Brodley, S. Fahmy, S. Floyd, W. Hardaker, A. Joseph, G. Kesidis, K. Levitt, B. Lindell, P. Liu, D. Miller, R. Mundy, C. Neuman, R. Ostrenga, V. Paxson, P. Porras, C. Rosenberg, J. D. Tygar, S. Sastry, D. Sterne, and S. F. Wu. "Cyber Defense Technology Networking and Evaluation", *Communications of the ACM* 47 (3), pp. 58-61, 2004.

[39] R. Larose and M. S. Eastin. "A Social Cognitive Explanation of Internet Uses and Gratifications: Toward a New Theory of Media Attendance". In *Proceedings of the International Communication Association, Communication and Technology Division*, 2003.

[40] R. Larose and N. Rifon. "Your Privacy is Assured – of Being Invaded: Web Sites With and Without Privacy Seals". Presented at *IADIS International Conference*, 2003.

[41] K. Campbell, L. A. Gordon, M. P. Loeb, and L. Zhou. "The Economic Cost of Publicly Announced Information Security Breaches: Empirical Evidence from the Stock Market", *Journal of Computer Security*, 11, pp. 431-439, 2003.

[42] L. A. Gordon and M. P. Loeb. "The Economics of Information Security Investment", *ACM Transactions on Information and System Security*, 5 (4), pp. 438-457, 2002.

[43] L. A. Gordon and M. P. Loeb. *Managing Cybersecurity Resources: A Financial Perspective*, McGraw-Hill, New York, 2005.

[44] L. Bodin, L. A. Gordon, and M. P. Loeb. "Evaluating Information Security Investments Using the Analytic Hierarchy Process", *Communications of the ACM*, 28 (2), pp. 79-83, 2005.

[45] J. Feigenbaum, D. Karger, V. Mirrokni, and R. Sami. "Subjective-Cost Policy Routing". In *Proceedings of the Workshop on Internet and Network Economics*, Lecture Notes on Computer Science, vol. 3828, Springer, Berlin, 2005.

[46] D. Bergemann, J. Feigenbaum, S. Shenker, and J. M. Smith. "Towards an Economic Analysis of Trusted Systems". In *Proceedings of the Third Annual Workshop on Economics and Information Security (WEIS)*, 2004.

[47] D. Bergemann, T. Eisenbach, J. Feigenbaum, and S. Shenker. "Flexibility as an Instrument in DRM Systems". In *Proceedings of the Fourth Annual Workshop on Economics and Information Security (WEIS)*, 2005.

Received July 10, 2007

Author Biographies

Shalom N. Rosenfeld is currently a graduate student in Electrical and Computer Engineering at the University of Maryland. He holds a B.A. in mathematics from Queens College – CUNY. His research interests include archetypes and computer security, modeling, and machine learning.

Ioana Rus is a research scientist at the Fraunhofer Center for Empirical Software Engineering Maryland. Her current research interests include software process modeling and simulation, software security, dependability engineering, measurement and empirical studies in software development and technology transfer, and experience and knowledge management. She has a Ph.D. in computer science and engineering, and work experience in software development, research, and teaching.

Michel Cukier is an Assistant Professor at the Center for Risk and Reliability in the Mechanical Engineering Department of the University of Maryland. He earned his doctorate from LAAS-CNRS, Toulouse, France in 1996 on coverage estimation of fault-tolerant systems. From 1996 to 2001, he was a member of the Perform research group in the Coordinated Science Laboratory at the University of Illinois, Urbana-Champaign, working on adaptive fault tolerance and intrusion tolerance. His current research interests include security evaluation, intrusion tolerance, distributed system validation, and fault injection.