Towards Understanding the (In)security of Networked Systems under Topology-directed Stealthy Attacks*

Paul Parker Shouhuai Xu
Department of Computer Science, University of Texas at San Antonio
{pparker,shxu}@cs.utsa.edu

Abstract

Consider a networked system of interest, where "networked" may be in a physical sense, meaning that the nodes are physically connected by point-to-point communication channels, or in a logical sense, meaning that the nodes are connected via edges that reflect certain relationships between the nodes (e.g., trust relationships). In such a system, once some nodes have been compromised, the attack would be directed by the network topology because compromise of a node may cause the compromise of its neighbors. Furthermore, the attack could be crafty or stealthy, meaning that it would always try not to trigger the intrusion detection alarm *of the networked system. In such a setting, a question of particular interest to the system administrator is:* What is the quantitative security assurance of the networked system? *This problem is notoriously known to be difficult, and the state-of-the-art is that we know very little about it. This paper aims to move a step towards resolving this problem.*

1 Introduction

Consider a networked system of interest, where "networked" may be in a physical sense, meaning that the nodes are physically connected by point-to-point communication channels, or in a logical sense, meaning that the nodes are connected via edges that reflect certain relationships between the nodes (e.g., trust relationships). Example types of such systems are virtual private networks and critical information infrastructures. In these systems, once some nodes have been compromised, the attack would be directed by the network topology (or *topology-directed*), because compromise of a node may cause the compromise of its neighbors. Furthermore, an attack could be crafty or *stealthy*, meaning that it always tries not to be detected.

In the above setting, a question of particular interest to the system administrator is: *What is the quantitative security assurance of the networked system?* This question is notoriously known to be difficult in general [4], and the state-of-the-art is that we know little about it (see Section 4 for related prior work).

1.1 Our Contributions

This paper aims to make a significant step towards resolving the above challenging problem by adopting a stochastic modeling approach. The rationale of this approach is that attacks and their detections appear to be probabilistic because, for instance, an attack may be based on an unknown or zero-day vulnerability. Ideally, such a model should simultaneously consider both topologies and intrusion detection capabilities of the networked systems.

Specifically, we present a model that can accommodate arbitrarily heterogeneous systems, where by "heterogeneous" we mean that different nodes or edges may exhibit different attributes (e.g., nodes may have different numbers of neighbors, and edges may have different capabilities in carrying out attacks). However, it turns out to be more challenging to accommodate the intrusion detection capabilities. This is mainly because such capabilities are often measured in a qualitative, rather than quantitative, fashion. Since the degree of an attack's stealthiness fully depends on the intrusion detection capabilities of the networked system, this also means that the former may not be quantified. Nevertheless, we manage to accommodate two important aspects of the intrusion detection capabilities in the model. The first is the threshold number of compromised nodes that sufficiently leads to the detection of the attack. We call such

*Supported in part by ARO and UTSA CIAS.

a threshold the *intrusion detection alarm* of the system. The second is the communication the attacker utilizes to launch or coordinate its attack. As we will see, our model does not require the attacker to do any such communication because, otherwise, it would help the intrusion detection system detect the attack — even if the individual communications are protected by cryptographic means.

Our model is validated through a simulation study. Furthermore, our model suggests a strategy whereby an *offline* attacker can compromise nodes without triggering the intrusion detection alarm. The strategy is simply that the attack stops just before it would trigger the intrusion detection alarm. However, such a strategy may not always be sufficient, because the model does not capture some other aspects of real-life intrusion detection systems. This leads us to investigate, although mainly in a qualitative fashion, some means that the attacker may exploit to make its attack more stealthy. As we will see, the attacker could easily make its attack much more stealthy. Still, the model would serve as a base for future studies for more accurately capturing intrusion detection capabilities, and thus security of networked systems under topology-directed stealthy attacks.

Outline: In Section 2 we explore our model. In Section 3 we investigate heuristic ways to reduce unnecessary traffic, which may lead to the detection of the attack. In Section 4 we discuss related prior work. We conclude this paper in Section 5.

2 Modeling Topology-directed Attacks

System setting. We model a physically or logically networked system as a finite graph $G = (V, E)$, where $V = \{1, 2, \ldots, n\}$ is the set of nodes (or hosts, users), and E is the set of edges. A graph G may be directed or undirected. If $(u, v) \in E$, then compromise of u may help the adversary compromise v.

A node $v \in V$ is always in one of the three states: secure, compromised (or infected), or removed (or cured, vaccinated, or patched). A removed node always remains in this state. We denote the aforementioned *intrusion detection alarm* by Δ that is publicly known, where $0 < \Delta < 1$. This means that when $\Delta \cdot n$ nodes become compromised, the intrusion detection alarm is triggered. Once the attack is detected, a vaccine or a cure may become available. Let θ_v be the parameter of the geometric distribution for node v to obtain the cure after it becomes available.

Adversary. We consider an attacker whose goal is to compromise nodes without triggering the intrusion detection alarm. We assume that the attacker knows the topology $G = (V, E)$ of the system as well as the configurations of its components via the parameters specified below.

Model. Let $s_v(t)$ be the probability that node v is secure at time t, $c_v(t)$ be the probability that node v becomes compromised at time t, and $r_v(t)$ the probability that node v becomes removed at time t. Let γ_{vu} be the probability that a secure node v is compromised because of its compromised neighbor u, where $(u, v) \in E$. Furthermore, let S_t be the random variable indicating the total number of secure nodes at time t, C_t be the random variable indicating the total number of compromised nodes at time t, and R_t be the random variable indicating the total number of removed nodes at time t. We may assume that $\forall\ v \in V$, $s_v(0) = 1$ and $c_v(0) = r_v(0) = 0$. We note that $\forall\ t$, $s_v(t) + c_v(t) + r_v(t) = 1$ and $S_t + C_t + R_t = n$, and that, by definition, $\mathsf{E}[S_t] = \sum_{v \in V} s_v(t)$, $\mathsf{E}[C_t] = \sum_{v \in V} c_v(t)$, and $\mathsf{E}[R_t] = \sum_{v \in V} r_v(t)$.

Figure 1. State transition of a vertex v

The state transition for a node $v \in V$ is specified in Figure 1. It basically says the following.

1. At each discrete time step $t + 1$, vertex v changes state from secure to compromised because of (1) its own behavior with probability α_v or (2) its neighbors' infection with probability $\eta_v(t)$, where

$$\eta_v(t) = 1 - \prod_{(u,v) \in E} [1 - \gamma_{vu}\, c_u(t)]. \tag{2.1}$$

   Note that $\eta_v(t)$ captures the probability that at time $t + 1$ node v becomes compromised because of its neighbors that were compromised at time t.

2. At each discrete time step t, vertex v changes state from compromised to removed because of its own reason with probability β_v, or because of the vaccine with probability $\sigma_v(t)$, where

$$\sigma_v(t) = \begin{cases} 0, & 0 \le t < \tau, \\ \theta_v, & t \ge \tau. \end{cases} \tag{2.2}$$

and

$$\tau = \max\{t \geq 0 : \sum_{v=1}^{n} [c_v(t) + r_v(t)] \leq n \cdot \Delta\}. \quad (2.3)$$

Note that τ captures the time when the attack is detected and vaccine becomes available.

3. At each discrete time step t, a vertex v, if in state secure, changes state to removed also with probability $\sigma_v(t)$.

Furthermore, if there are multiple applicable state transitions, vaccination overrules the others. Then, we have, for $v = 1, 2, \cdots, n$ and $t = 1, 2, \cdots$,

$$\begin{cases} s_v(t+1) &= [1-\alpha_v]\,[1-\sigma_v(t)]\,[1-\eta_v(t)]\,s_v(t), \\ c_v(t+1) &= [\alpha_v + \eta_v(t) - \alpha_v\eta_v(t)]\,[1-\sigma_v(t)]\,s_v(t) \\ &\quad +[1-\beta_v]\,[1-\sigma_v(t)]\,c_v(t), \\ r_v(t+1) &= \sigma_v(t)\,s_v(t)+ \\ &\quad [\beta_v + \sigma_v(t) - \beta_v\sigma_v(t)]\,c_v(t) + r_v(t). \end{cases} \quad (2.4)$$

2.1 On the Accuracy of the Model

In order to validate the accuracy of Eq (2.4), we conduct a simulation study based on the PGP certificate graph [18]. The PGP certificate graph is relevant because it may be seen as a logically networked system that may somewhat reflect the trust relationships between the users. The data is based on the version of October 2, 2005 available at `http://dtype.org/keyanalyze/`, which consists of 183,197 nodes and 401,179 edges. Without loss of generality (as the model is equally applicable to both directed and undirected graphs), we treat the edges as undirected.

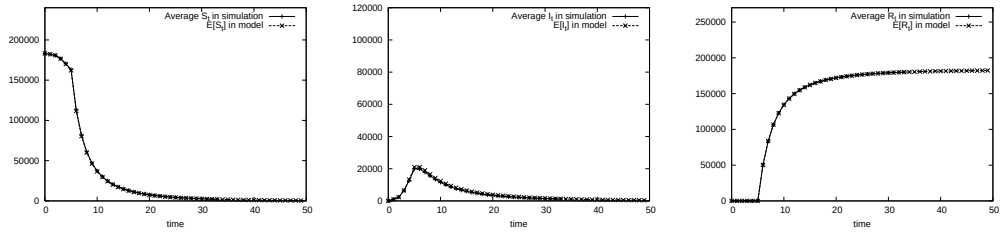
We assign some arbitrary values to the parameters; in practice, such values should be derived from history (e.g., incident) data of the system. We set α_v to be uniformly chosen from $[0.001, 0.01]$ for each $v \in V$; $\beta_v = 0$ for all $v \in V$, meaning that vaccine is not available until the attack has been detected (this is for simplicity); θ_v to be uniformly chosen from $[0.05, 0.5]$; and γ_{vu} to be uniformly chosen from $[0.05, 0.10]$ for each edge $(u, v) \in E$. Furthermore, we consider two cases: $\Delta = 0.10$ and $\Delta = 0.30$. In Figure 2, we compare the $\mathsf{E}[S_t]$, $\mathsf{E}[C_t]$, and $\mathsf{E}[R_t]$ obtained via Eq (2.4) with the same metrics obtained by averaging over 30 simulation runs. After the attack is detected when $\Delta \cdot n$ nodes become compromised, compromised nodes may become removed.

It is clear that the model prediction and simulation match quite well. Therefore, one may use Eq (2.4) to compute $\mathsf{E}[S_t]$, $\mathsf{E}[I_t]$, and $\mathsf{E}[R_t]$ for *any* t without running a significant number of simulations (e.g., 30). In particular, one can use Eqs (2.3) and (2.4) to determine τ, the time the attack is detected.

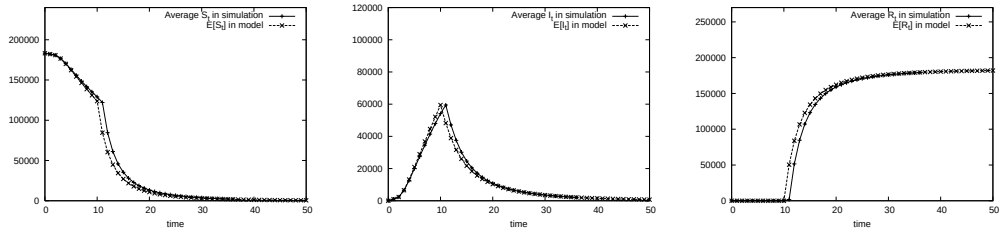
3 Making Attacks More Stealthy

The above model captures two important aspects of intrusion detection capabilities, namely the *intrusion detection alarm* (which reflects the threshold number of compromised nodes that lead to the detection of the attack) and the fact that the attacker does not need to stay online (in communication) during the attack process. There are some other important aspects of intrusion detection systems, including (1) an attack may be detected because of the content of its attack messages, and (2) an attack may be detected because of the high bandwidth consumption incurred by the compromised nodes when attempting to compromise other nodes (which is different from the communication between the attacker and the compromised nodes — something actually avoided in our model). Attackers might circumvent above point (1) by exploiting some unknown or zero-day vulnerabilities. Furthermore, the attack messages between compromised nodes and yet-to-be compromised ones could be protected using cryptographic means that are out of the control of the intrusion detection system. This is particularly relevant when the system is designed for secure communication between nodes.

The above point (2) is important because even if the attack payloads are encrypted, the bandwidth consumed by the attack traffic may still be detected by the intrusion detection system. However, it is far from trivial to see how an attacker would deal with the above point (2). Since the model is stochastic, it is unknown a priori which nodes will become compromised before triggering the intrusion detection alarm. Furthermore, since the attacker does not communicate with the compromised nodes, the compromised nodes themselves need to decide which nodes they should try to attack. Since the compromised nodes would have the tendency to attack all their neighbors — even if they have already been compromised, there might be a large volume of *useless traffic*. By "useless traffic" we mean the traffic consisting of the *useless attack messages* that do not *directly* lead to the successful compromise of any secure node. Let us define an *infection attempt* as a probe of an infected node A to any other node B (i.e., A trying to attack B). If B is



(a) $\Delta = 0.10$



(b) $\Delta = 0.30$

Figure 2. Model vs. simulation with $\alpha_v \in [0.01, 0.1]$, $\gamma_{vu} \in [0.05, 0.1]$, $\beta_v = 0$ and $\theta_v \in [0.05, 0.5]$

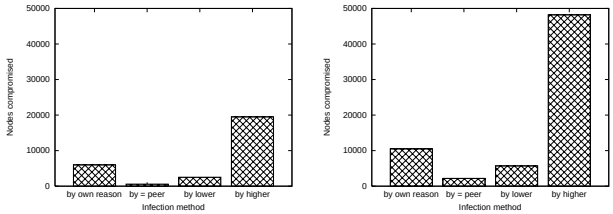
already compromised, or B is not yet compromised but the probe does not successfully attack B, then the attempt is considered as a *useless attack message*. Therefore, in order to deal with the above point (2), a sophisticated attacker would try to reduce the volume of *useless traffic*.

3.1 Some Heuristic Strategies

Thus we explore some heuristics an attacker may adopt to reduce the volume of useless traffic. We start by analyzing the spectrum of useless traffic, which gives us some useful hints in designing strategies that will be validated.

Reasons for nodes to get compromised. Figure 3 plots the number of compromised nodes in the following four categories: "by own reason" – compromised because of its own reason, "by = peer" – compromised by a neighbor of the same degree, "by lower" – compromised by a lower degree neighbor, and "by higher" – compromised by a higher degree neighbor. It clearly shows that most nodes get compromised because of their higher degree neighbors. This can be understood as those higher degree nodes have a larger chance of getting compromised, and have more neighbors to attack. This suggests that many attack messages from low degree nodes to their high degree neighbors may turn out to be useless. Therefore, we conjecture that it may be effective to reduce the attack messages from lower degree nodes to their higher degree neighbors. This is one of the strategies that we explore below.

Strategies to reduce *useless traffic*. Based on the above analysis we consider the following heuristics, which the adversary may adopt to reduce useless traffic.



(a) $\Delta = 0.10$ (b) $\Delta = 0.30$

Figure 3. Reasons nodes become compromised

Strategy 0: No special measures are taken. That is, this is the normal attack process.

Strategy 1: If a node v_2 was compromised because of an attack launched from v_1, then v_2 will not attempt to attack v_1, because v_1 must have already been compromised.

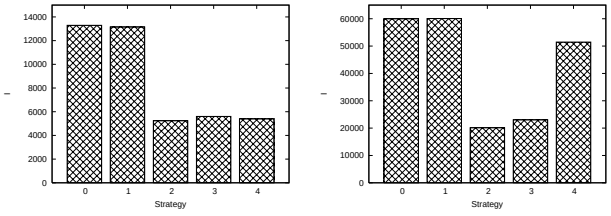
Strategy 2: If $deg(v_1) > deg(v_2)$, then v_2 never attempts to attack v_1, where $deg(v)$ denotes the degree of a node v.

Strategy 3: For each compromised node u, for each of its neighbors v, u flips a coin uniformly at random to decide whether to attack v. This decision is made independent of whether v is compromised and whether v attacked u.

Strategy 4: For each compromised node u, for each of its neighbor v, u flips a biased coin on whether to attack v. The bias is to attack at probability 0.75. This decision is made independent of whether v is compromised and whether v attacked u.

3.2 On the Impact of the Strategies

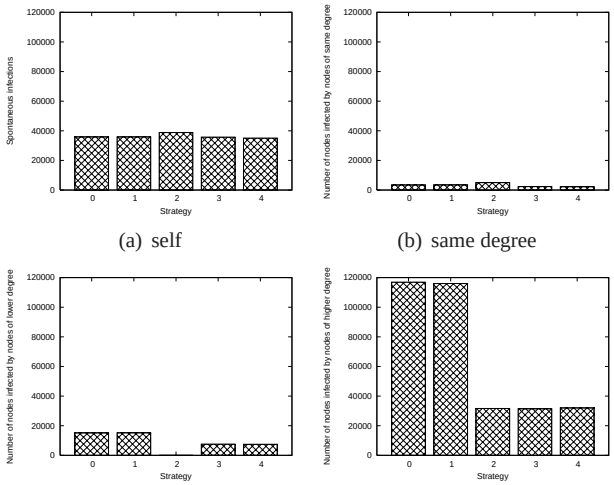
First, we explore the number of compromised nodes when the attack stops at time τ_0, which is the time the number of compromised nodes reaches the intrusion detection alarm Δ in **Strategy 0**, and can be obtained from Eqs (2.3) and (2.4). Although the simplest method is always to stop an attack at time τ_0, it may be far from the optimal. Figure 4 shows the number of compromised nodes in different strategies until time τ_0. It shows that **Strategy 1** does not lead to a decrease of the number of compromised nodes in both cases (because it only prevents redundant attacks). However, **Strategies 2** and **3** lead to a sharp decrease in the number of compromised nodes in either case. (Indeed, simulation indicates that, in the case of $\Delta = 0.10$, both **Strategies 0** and **1** take 6 time steps to reach the threshold of Δ, whereas **Strategies 2**, **3** and **4** take 12 time steps to reach the same threshold.) However, **Strategy 4** in the case of $\Delta = 0.1$ has a significant impact, but not in the case of $\Delta = 0.30$. This means that **Strategy 4** is effective in the case of $\Delta = 0.30$, if it could also significantly reduce the volume of useless traffic.



(a) $\Delta = 0.10$ (b) $\Delta = 0.30$

Figure 4. Number of compromised nodes until time τ_0, which is calculated from Strategy 0.

Second, we explore the impact of the strategies on the spectrum of reasons the nodes are compromised. For this purpose, we compare the strategies with respect to the following categories: (a) the number of nodes getting compromised because of their own behavior, (b) the number of nodes getting compromised by their neighbors of the same degree, (c) the number of nodes getting compromised by their neighbors of lower degree, and (d) the number of nodes getting compromised by their neighbors of higher degree. This is consistent with the strategies adopted in Section 3.1.



(a) self (b) same degree

(c) lower degree (d) higher degree

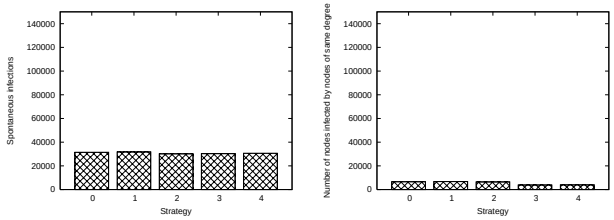
Figure 5. Reasons nodes become compromised until reaching threshold $\Delta = 0.10$

Figures 5 and 6 plot the impact of different strategies on the number of nodes getting compromised in different categories for $\Delta = 0.10$ and $\Delta = 0.30$, respectively. Figures 5.(d) and 6.(d) clearly show that the reduced number of compromised nodes is actually due to the drop of the number of nodes that are attacked by their higher degree neighbors. Note that Figures 5.(c) and 6.(c) show that for **Strategy 2** no nodes are attacked from their lower degree neighbors.

Comparing Figures 5 and 6, we notice that the percentage of nodes getting compromised because of their own reasons in the case of $\Delta = 0.30$ is less than its counterpart in the case of $\Delta = 0.10$. This is because we used a small parameter $\alpha_v \in [0.01, 0.1]$, and when the simulation runs longer, nodes are more likely to be attacked by their neighbors. We also notice that the number of nodes getting compromised by their higher degree neighbors in the case of $\Delta = 0.30$ is more than in the case of $\Delta = 0.10$. This is because in the former case the vaccination is available at a later point in time.

Third, we explore the reduction in the volume of useless traffic from the following three perspectives: reduction in the total volume, reduction at each time step, and reduction in average volume where the average is taken over the respective time steps.

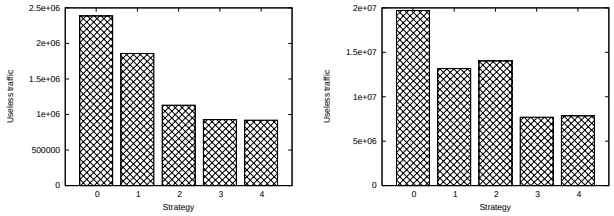
Figure 7 plots the total amount of useless traffic until the



(a) self (b) same degree

(c) lower degree (d) higher degree

Figure 6. Reasons nodes become compromised **until reaching threshold** $\Delta = 0.30$



(a) $\Delta = 0.10$ (b) $\Delta = 0.30$

Figure 7. Total useless traffic in different strategies

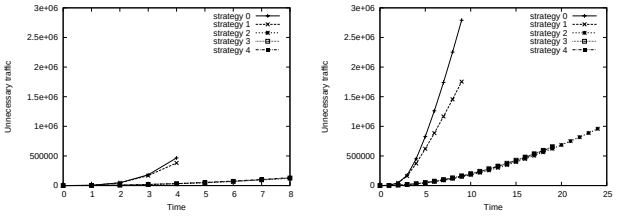
number of compromised nodes reaches $\Delta \cdot n$. It shows the following:

- For $\Delta = 0.10$, **Strategy 1** reduces useless traffic about 25%, **Strategy 2** reduces useless traffic about 50%, and **Strategies 3** and **4** each reduce it about 60%.
- For $\Delta = 0.30$, **Strategy 1** reduces useless traffic about 30%, **Strategy 2** reduces useless traffic about 30%, and **Strategies 3** and **4** each reduce it about 60%.

It is clear that **Strategy 0**, merely stopping the attack when reaching $\Delta \cdot n$, is not optimal.

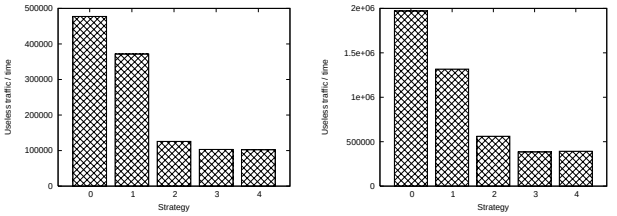
Figure 8 plots the amount of useless traffic at each time step. It shows that **Strategies 2**, **3** and **4** eliminate a large amount of useless traffic, which could otherwise make detection of the attacks easy.

Figure 9 plots the average amount of useless traffic until reaching $n \cdot \Delta$. This metric may be of interest because it



(a) $\Delta = 0.10$ (b) $\Delta = 0.30$

Figure 8. Useless traffic at each time step



(a) $\Delta = 0.10$ (b) $\Delta = 0.30$

Figure 9. Average rate of useless traffic

captures the average rate of useless attack messages. Since different strategies may take different time periods to reach $\Delta \cdot n$, the average reductions in useless traffic, as shown in Figure 9, are not necessarily proportional to the total reduction in useless traffic, as shown in Figure 7. Furthermore, it shows the following:

- In the case of $\Delta = 0.10$, **Strategy 1** reduces average useless traffic about 20%, **Strategy 2** reduces average useless traffic about 70%, and **Strategies 3** and **4** both reduce average useless traffic about 75%.
- In the case of $\Delta = 0.30$, **Strategy 1** reduces average useless traffic about 30%, **Strategy 2** reduces average useless traffic about 70%, and **Strategies 3** and **4** both reduce average useless traffic about 75%.

Summary. There are simple strategies that an attacker may exploit to make the attacks more stealthy. For example, among the above studied ones, **Strategies 3** and **4** are effective.

4 Related Work

Existing graph-based analysis approaches vs. our modeling approach. There are three types of graph-based analysis approaches.

1. The first is based on *privilege graphs* [5, 13], where a node represents a set of privileges on some objects and

an arc represents a vulnerability. An arc exists from one node to another if there is a method allowing a user owning the former node's privileges to obtain those of the latter's.

2. The second is based on *attack graphs*, where a node represents the state of a network (i.e., the values assigned to relevant system attributes such as specific vulnerabilities on various hosts and connectivity between hosts), and an edge represents a step in an attack (cf. [14, 9, 1] and their references). A designated node (or set of nodes) represents the initial state(s), and each transition represents a specific exploit that an attack can carry out. This technique can identify the set of attack paths that have a high probability of success for the attacker.

3. The third is based on *key challenge graphs* [3], where a node represents a host, and an arc represents a *key challenge* that is an abstraction to capture access control. A key challenge is, for instance, a password authentication prior to accessing to a resource. The starting point of an attack could be one or more vertices, which are assumed to be initially in the control of the attacker. The target of an attack could be one or more vertices, for which the attacker knows the location and the paths to reach them. A successful attack is a sequence of zero or more vertices not in the initial set but eventually containing all the target nodes. The cost of an attack is measured as the sum of the effort required to compromise individual vertices by attempting to counter the key challenges on the edges. For these a problem of particular interest is to find an attack path of minimum cost.

The overall goal of the above-mentioned approaches and our modeling approach is similar, namely to understand security from the perspective of considering the system of interest as a whole, rather from the perspective of some isolated attacks against building-block components. However, there are important differences between these approaches and ours.

The main difference between privilege/attack graphs and ours lies in the model *assumptions* and *scalability*. First, privilege/attack graphs are based on the systems' *known* vulnerabilities that have not been patched. While this is certainly a realistic threat, we believe that it would be better resolved using vulnerability-specific countermeasures (e.g., [15]). In contrast, our modeling approach does not assume known vulnerabilities; instead, it is quite appropriate for modeling attacks that may be based on *zero-day or unknown* vulnerabilities — this further justifies our use of stochastic models. Second, privilege/attack graphs based approaches suffer from limited scalability, because of their inherent exponential state explosion; whereas our approach is scalable.

The main difference between the key challenge graph approach and ours lies in the model *purpose* (i.e., the questions targeted by the models) and *capability*. First, the key challenge graph approach emphasizes the algorithmic aspect of finding an optimal attack path, namely that the adversary can achieve its goal with minimal effort or cost. In contrast, our modeling approach aims to understand the dynamic behavior of system evolutions, and to answer questions such as the number of compromised private keys in a public key infrastructure. Second, the key challenge graph can only capture the attack behaviors with respect to some specific starting points (i.e., the nodes that have been initially compromised); otherwise, it will encounter the exponential explosion problem. In contrast, our modeling approach does not need to know the initially compromised nodes, nor even to have any.

Existing epidemic models vs. our model. Existing epidemic models (e.g., [10] and its numerous follow-ons) are typically adapted from the biological epidemiology models for *homogeneous* systems [11, 2]. Such models assume that every individual has equal contact to everyone else in the population, and that the rate of infection is largely determined by the density of the infected individuals. Homogeneous models may be useful in some cases, e.g., when a worm spreads via *truly random scanning*. However, homogeneous models do not apply to heterogeneous systems, which are harder to model because topology-based spreading does not rely on random scanning.

There are a few investigations on non-homogeneous models. Semi-heterogeneous networks are investigated in [17, 16] with discrete time models, and in [7] with an analogous continuous time model. The models of [17, 16] are perhaps the closest to ours. However, they have the following drawbacks. (1) They cannot express that the number of compromised nodes is persistently greater than zero, even if the ratio between the attack death rate and the attack birth rate is above the threshold. (2) They can only model semi-heterogeneous systems, where different nodes may have different degrees. However, both capabilities are offered in our models.

Finally, we notice that, to our knowledge, there is no prior work, even in the context of homogeneous models, that investigated the possibility and usefulness of incorpo-

rating the intrusion detection capabilities of the systems of interest. We should remark that [12] investigated the possibility of self-stopping worms, which can be seen as a special case of our model with $\Delta = 1$. However, this work neither explored the issues of topology-directed attacks, nor considered the intrusion detection capability of the system.

5 Conclusion and Future Work

We explored the (in)security of networked systems under topology-directed stealthy attacks via a stochastic modeling approach. The model accommodates arbitrarily heterogeneous network topologies, and some aspects of the intrusion detection capabilities of the system. The accuracy of the model is validated via a simulation study. Besides the hints an attacker may draw from the model (e.g., compromising as many as possible nodes without triggering the intrusion detection alarm), we further investigated heuristic methods that an attacker may adopt to make the attack more stealthy.

This paper is presented more from the perspective of an attacker. This serves as a first step towards effectively dealing with the attacks in future works. There are many interesting future research directions. Examples include: How can we precisely quantify intrusion detection capabilities in a way that can help us understand what countermeasures would be more effective than others? Can we derive some analytic results on the effectiveness of countermeasures in heterogeneous systems? What is the optimal strategy that an attacker can utilize to reduce the useless traffic?

References

[1] P. Ammann, D. Wijesekera, and S. Kaushik. Scalable, graph-based network vulnerability analysis. In *Proc. ACM CCS'02*, pp 217–224.

[2] N. Bailey. *The Mathematical Theory of Infectious Diseases and Its Applications*. 2nd Edition. 1975.

[3] R. Chinchani, A. Iyer, H. Ngo, and S. Upadhyaya. Towards a theory of insider threat assessment. In *Proc. of IEEE DSN'05*, pp 108–117.

[4] The Computing Research Association. Four grand challenges in trustworthy computing. http://www.cra.org/Activities/grand.challenges/, 2003.

[5] M. Dacier and Y. Deswarte. Privilege graph: an extension to the typed access matrix model. In *Proc. ESORICS'94*, pp 319–334.

[6] M. Dacier, Y. Deswarte, and M. Kaaniche. Models and tools for quantitative assessment of operational security. pp 177–186, 1996.

[7] A. Ganesh, L. Massoulie, and D. Towsley. The effect of network topology on the spread of epidemics. In *Proceedings of IEEE Infocom 2005*, Vol. 2, pp 1455-1466, 2005.

[8] S. Jha, O. Sheyner, and J. Wing. Two formal analyses of attack graphs. In *Proc. of IEEE Computer Security Foundations Workshop (CSFW'02)*, pp 49–65.

[9] S. Jha and J. Wing. Survivability analysis of networked systems. In *Proc. ICSE'01*, pp 307–317.

[10] J. Kephart and S. White. Directed-graph epidemiological models of computer viruses. In *IEEE Symposium on Security and Privacy*, pp 343–361, 1991.

[11] A. McKendrick. Applications of mathematics to medical problems. *Proc. of Edin. Math. Society*, 14:98–130, 1926.

[12] J. Ma and G. Voelker and S. Savage. Self-stopping worms. *Proc. of ACM Worm'05*, pp 12 –21.

[13] R. Ortalo, Y. Deswarte, and M. Kaaniche. Experimenting with quantitative evaluation tools for monitoring operational security. *IEEE Trans. Softw. Eng.*, 25(5):633–650.

[14] C. Phillips and L. Swiler. A graph-based system for network-vulnerability analysis. In *Proc. of workshop on New security paradigms (NSPW'98)*, pp 71–79.

[15] H. Wang, C. Guo, D. Simon, and A. Zugenmaier. Shield: vulnerability-driven network filters for preventing known vulnerability exploits. In *Proceedings of the ACM SIGCOMM 2004*, pages 193–204, 2004.

[16] J. Wang, L. Lu, and A. Chien. Tolerating denial-of-service attacks using overlay networks – impact of topology. In *Proc. of ACM workshop on survivable and self-regnerative systems*, 2003.

[17] Y. Wang, D. Chakrabarti, C. Wang, and C. Faloutsos. Epidemic spreading in real networks: An eigenvalue viewpoint. In *Proc. of IEEE SRDS'03*, pp 25–34.

[18] P. Zimmermann. *The official PGP user's guide*. MIT Press, 1995.