Adversary-Driven State-Based System Security Evaluation

Elizabeth LeMay[†], Willard Unkenholz[‡], Donald Parks[‡],
Carol Muehrcke*, Ken Keefe[†], William H. Sanders[†]

[†]Information Trust Institute, Coordinated Science Laboratory,
Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign
[‡]U.S. Department of Defense, Fort Meade, MD, *Cyber Defense Agency, Wisconsin Rapids, WI
{evanrui2, kjkeefe, whs}@illinois.edu, {wunkenho, dparks}@restarea.ncsc.mil,
cmuehrcke@cyberdefenseagency.com

ABSTRACT

To provide insight on system security and aid decision-makers, we propose the ADversary VIew Security Evaluation (ADVISE) method to quantitatively evaluate the strength of a system's security. Our approach is to create an executable state-based security model of a system. The security model is initialized with information characterizing the system and the adversaries attacking the system. The model then simulates the attack behavior of the adversaries to produce a quantitative assessment of system security strength. This paper describes the system and adversary characterization data that are collected as input for the executable model. This paper also describes the simulation algorithms for adversary attack behavior and the computation for the probability that an attack attempt is successful. A simple case study illustrates how to analyze system security using the ADVISE method. A tool is currently under development to facilitate automatic model generation and simulation. The ADVISE method aggregates security-relevant information about a system and its adversaries to produce a quantitative security analysis useful for holistic system security decisions.

Keywords

Security Quantification, State-based Model, Simulation, Adversary Attack Behavior

1. INTRODUCTION

Making sound security decisions when designing, operating, and maintaining a complex system is a challenging task. Analysts need to be able to understand and predict how different factors affect the overall system security.

During system design, before the system is built, security analysts want to compare the security of multiple proposed system architectures. After a system is deployed, analysts want to determine where security enhancement efforts should be focused by examining how the system is most likely to be successfully penetrated. And when several security enhancement options are being considered, analysts would like to evaluate the relative merits of each.

In each of these scenarios, quantitative security metrics could provide insight on system security and aid security decisions. Quantitative metrics enable ranking the alternatives to determine the best option. Quantitative assessments of system security are also valuable for risk management trade-off decisions.

Many low-level security metrics exist, but the challenge is to produce a quantitative assessment of the security of the system as a whole. This requires an understanding of how the components interact within the system. We propose to construct a security model of the system to facilitate a quantitative holistic security assessment.

An effective security model should contain system information relevant to a security analysis, including the penetrability of security-enforcing devices such as firewalls and the existence of possible connection paths between disparate parts of a system. Our method provides a formal structure to aggregate relevant system details in an organized fashion.

An effective security model should also describe the adversaries threatening the system. One key component of our approach is the inclusion of adversary attack behavior models. We assert that meaningful measurements of system security cannot occur in a vacuum devoid of information about the system's adversaries. A system defense should be evaluated in the context of the anticipated opponents.

Different systems face different adversaries. Government-owned systems are likely to attract attacks from nation-state adversaries. Corporate networks are likely to draw the attention of other types of attackers, such as competitors, criminals, or disgruntled employees. To produce meaningful analysis results, security should be analyzed in the context of the specific adversaries likely to attack the system.

For this adversary-behavior-modeling approach, a security model should include profiles of the adversaries that enable predictions about their likely attack behavior. Our method specifies how to generate an adversary profile containing attack goals, attack preferences, and other factors that influence attack behavior decisions. Our method also includes algorithms for making probabilistic statements about attack behavior based on an adversary profile.

Our method is intended to produce holistic quantitative security assessments of systems in order to evaluate alternatives to aid system-level security decisions. Other tools already exist for detailed configuration analysis of deployed systems; this is not the purpose of our method.

Our method goes beyond existing security risk management methods, such as NIST's Risk Management Guide for Information Technology Systems [8]. Other methods, like our method, aggregate information into a system-level as-

MetriSec2010 September 15, 2010, Bolzano-Bozen, Italy

sessment of security. However, these methods are often limited to static qualitative results and are unable to provide much insight on how attackers are likely to interact with the system as they attempt attacks. For example, in the Mission Oriented Risk and Design Analysis (MORDA) [1] and Network Risk Assessment Tool (NRAT) [10] methods, the adversarial decision is a one-time selection of a full attack vector. Our method dynamically models step-by-step adversarial decisions, in which the outcome of previous attack step decisions impacts the next decision.

Our method leverages existing work on attack graphs [7, 6, 9] and privilege graphs [2, 3, 5], but extends the attack graph concept by creating executable models driven by the attack preferences of individual adversaries. Thus, an analysis using our method can be customized to reflect the attack behavior of different types of adversaries with different attack objectives, attack preferences, resource levels, attack skill levels, system access, and system knowledge.

The risk assessment method described in [4] offers a means for characterization of adversaries, which we have leveraged for this work. In that method, relative adversary preference scores for possible attacks are calculated, allowing the analyst to focus on the highest scoring attacks. ADVISE goes beyond this approach by simulating adversary behavior based on the adversary's attack preferences.

Previously, security analysts using modeling and simulation would design a unique security model for each system analysis, but our method introduces a precise, repeatable technique to create state-based security models.

This paper presents a new adversary-driven state-based system security evaluation method called ADversary VIew Security Evaluation (ADVISE) that includes the following contributions:

- a precise adversary characterization format that enables simulation of how a particular adversary is likely to attack a system,
- a precise system characterization format that describes the possible attack paths into a system and includes security-relevant system details,
- simulation algorithms for computing adversary attack decisions and the probability that an attack attempt will be successful, and
- executable models that produce mission-relevant quantitative security metrics.

2. ADVISE METHOD OVERVIEW

To obtain a high-level holistic security assessment from a collection of low-level pieces of information, we propose the ADVISE method to quantitatively evaluate a system's security. Our approach is to create an executable state-based security model of the system. The security model is initialized with information characterizing the system and the adversaries attacking the system. The analyst specifies the security metrics of interest for the system and generates metrics data by running discrete-event simulations of the adversaries attacking the system.

The ADVISE method for system security analysis consists of three main phases. Phase one is the characterization of the system and its adversaries and the specification of the desired security metrics. Phase two is the generation of a dynamic executable security model created from the characterization information in phase one. Phase three is the execution of the security model generated in phase two. Quantitative security metrics are produced as model outputs.

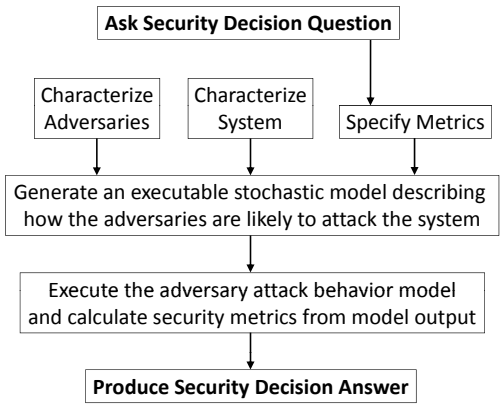

Figure 1: The ADVISE Method for Producing Quantitative Model-Based System Security Metrics

Figure 1 illustrates how the three phases of the ADVISE method assist security analysts in generating answers to security decision questions by incorporating information about the system, its adversaries, and the security metrics.

The ADVISE method precisely specifies the format of the system and adversary characterization data. The adversary data describes the system-specific attack goals of a particular adversary as well as a more general statement on how the adversary prefers to conduct attacks (e.g., risk-averse or risk-tolerant). The adversary data also includes ratings of the adversary's skill levels in conducting a variety of types of attacks and also an initial assessment of what access to the system and knowledge about the system the adversary possesses before beginning any attacks.

The system characterization data in an ADVISE analysis is aggregated into an attack execution graph, which is similar to an attack graph but augmented with additional data needed to create an executable model. Attacks against a system are constructed of chains of small attack steps. Each attack step, if successfully executed, can increase the adversary's access or knowledge of the system and move him or her closer to achieving attack goals, such as loss of confidentiality of specific data, loss of integrity of specific data, or loss of availability of specific services and/or data within the system. The attack execution graph defines and describes all possible attack steps an adversary could try against the system.

The metrics specification enables the executable model to produce output relevant to the security question posed. The metrics enable an analyst to study the average time for an adversary to reach an attack goal, the mostly likely attack path to an attack goal, and the average total cost to the adversary to reach an attack goal.

The executable security model simulates the adversary attack behavior. The simulation involves three computations for each attack step: the Boolean expression evaluation of the attack step precondition, the attack step attempt deci-

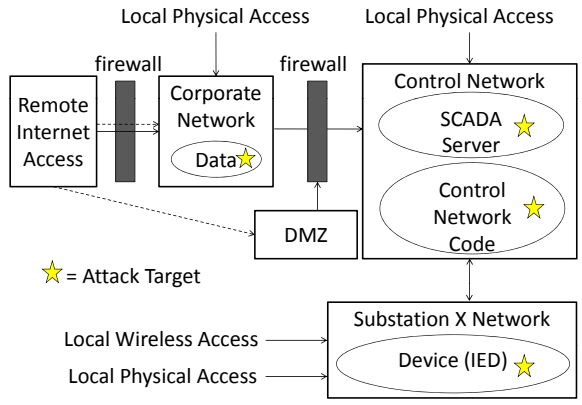

Figure 2: Case Study of Adversaries Attacking an Electric Power SCADA System

sion, and the attack step attempt outcome. These values are computed using the system and adversary characterization data, as well as current state information provided by the security model.

In this way, the ADVISE method produces mission-relevant (as specified by an analyst) security metrics from the simulation output.

3. CASE STUDY INTRODUCED

The ADVISE system security analysis method is applicable to a variety of systems at varying levels of abstraction and with varying levels of detail. However, to illustrate the concepts introduced in the ADVISE method, the definitions in this paper are accompanied by examples from a generic electric power SCADA system.

The example electric power SCADA system is depicted in Figure 2. The rectangles represent different domains within the system, and the arrows represent possible connections between access domains. The dotted lines are VPN connections requiring a VPN user account and password. There are two firewalls in the system: one between the Internet and the corporate network, and another between the corporate network and the control network. Local physical access points and local wireless access points are also noted in the system diagram.

In Figure 2, stars denote possible attack targets within the system. Not all adversaries seek all the attack goals. One adversary may only desire to view the confidential data stored on the corporate network. Another adversary may seek to crash the SCADA server in an attempt to disrupt electric power service.

A security analysis of this type of system is useful for identifying possible attack paths into the system and determining which paths are most attractive to attackers. An ADVISE security analysis can show the most likely attacks by an insider adversary with a particular type of system access or by an outsider adversary with connections to an economic competitor.

The case study is intended to demonstrate how the ADVISE method concepts can be applied to a specific system analysis.

4. ADVERSARY CHARACTERIZATION

The ADVISE method of security analysis is driven by the particular adversary or set of adversaries attacking the system of interest. Each adversary exhibits attack behavior that reflects his or her internal attack goals and attack preferences. We strive to characterize adversaries in a way that enables modeling of their attack decisions and the results of their attack attempts. Figure 1 shows that the adversary characterization provides input for generating the executable model.

ADVISE adversary characterization includes system-independent and system-dependent components. The system-independent components are the *attack preference weights* and the *attack skill levels*. The system-dependent components are the *attack goals*, *system knowledge*, and *system access*.

Attack preference weights describe the relative attractiveness of unit changes in each of the four core criteria that adversaries consider when deciding among their attack options. We theorize that attack decisions are based on the following criteria: (1) cost to the adversary in attempting the attack step, (2) payoff to the adversary for successfully executing the attack step, (3) probability of successfully completing the attack step, as perceived by the adversary, and (4) probability of being detected by the system during or after attempting the attack step. A utility scale is constructed for each of the four preference weights such that each preference weight is a real number between zero and one.

Different adversaries may have different attack preference weights. A well-funded nation-state may care little about the cost of an attack but may tolerate only very low probability of detection in the attack steps it chooses to attempt. However, a resource-constrained lone hacker may try riskier attack steps with a low probability of success and high probability of detection, but the cost to attempt must be low. These differences are reflected in the attack preference weights.

Attack skill levels describe the proficiency of the adversary in executing specific types of attacks. For each attack skill, each adversary is assigned a skill level represented as a real number between zero and one. An attack skill level of zero means that the adversary is incapable of successfully executing an attack that requires this attack skill. An attack skill level of one means that the adversary is fully proficient in executing attacks requiring this attack skill. In the SCADA system example, one relevant attack skill is the ability to exploit VPN software. Another attack skill is the ability to defeat a remote network log-in process. For each attack skill, a rubric can be developed to assist security analysts in assigning meaningful attack skill level numbers for each adversary.

Attack goals are system-specific and describe what end goal the adversary is seeking as he or she attacks the system. Different adversaries may be working toward different attack goals. Some adversaries may be motivated to pursue multiple attack goals. Each attack goal takes on a binary value: zero if the adversary is not motivated by this attack goal, one if the adversary is. In the SCADA system example, one attack goal is to steal proprietary pricing data stored in the corporate network. Another attack goal is to crash the SCADA server in the control network to disrupt the proper operation of the SCADA system.

System knowledge describes the pieces of system-specific

knowledge that can aid the adversary in successfully executing an attack. System knowledge can include user account names and passwords (e.g., VPN account information), system architecture details (e.g., the location in the network of the server hosting a vital service), and system administration procedures (e.g., administrators routinely ignore certain intrusion detection alerts due to an overwhelming number of false positives). For each key piece of system information, each adversary is assigned a binary value: one (meaning he or she possesses this information) or zero (he or she does not possess it). During a simulation of adversaries attacking a system, the adversaries may attempt to increase their system knowledge.

System access describes the system-specific network domains or physical locations through which adversaries can attack the system. In the SCADA system example, types of system access include network domains such as Internet access, corporate network access, and substation network access. System access also includes physical access such as local physical access to the corporate network or physical proximity to the substation wireless access point. For each distinct type of system access, each adversary is assigned a binary value: one (meaning he or she possesses this access) or zero (he or she does not possess it). During a simulation of adversaries attacking a system, the adversaries are likely to attempt to increase their system access.

The five categories of information above—attack preference weights, attack skill levels, attack goals, system knowledge, and system access—characterize adversaries in a way that enables simulation of their attack behavior decisions and computation of the likelihood of the success of their attack attempts.

To facilitate the efficient re-use of adversary information collected for an ADVISE analysis, a tool currently under development will provide functionality to enable analysts to develop a library of adversary characterization profiles. When a security analyst is assessing the security of multiple systems facing the same adversaries, the adversary characterization data can be pulled from the library and modified as needed to fit the specific system.

5. SYSTEM CHARACTERIZATION

In the ADVISE method of security analysis, attacks against a system are studied as chains of attack steps. To achieve an attack goal, an adversary must successfully execute a series of attack steps that lead to the attack goal state. Recall that stating an adversary's attack goals is a part of the adversary characterization process. Figure 1 shows that the system characterization provides input for generating the executable model of adversary attack behavior. ADVISE system characterization takes the form of an attack execution graph composed of many precisely defined attack steps.

5.1 Attack Execution Graph

The system is represented as the set of all possible attack steps from the perspective of an adversary. Because the execution of some attack steps depends upon successful completion of other attack steps, the set of all attack steps forms a graph structure we call an *attack execution graph.* Figure 3 shows an attack execution graph. The ovals at the bottom of the attack execution graph are the attack goal states an adversary may pursue. Each long rectangular box represents an attack step. The triangles, circles, and squares

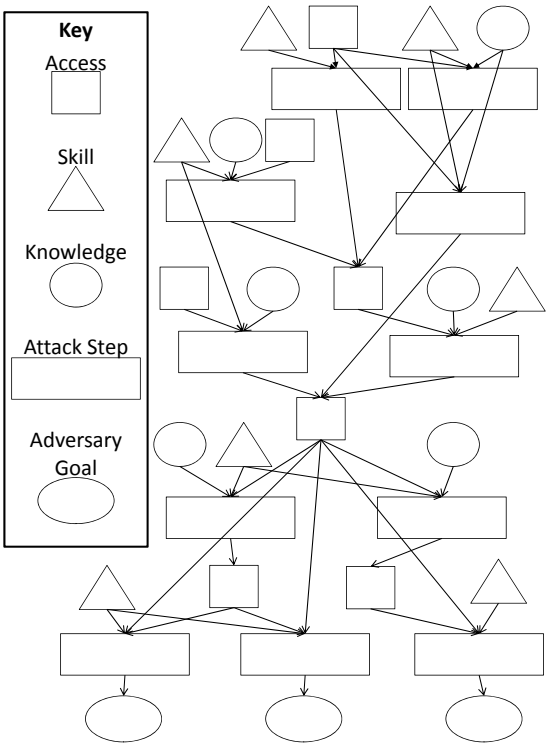

Figure 3: An Attack Execution Graph

represent different types of attack skills, system knowledge, and system access, respectively.

When attempting an attack step requires an adversary to possess some set of attack skills, system knowledge, and system access, arrows pointing from the appropriate triangles, circles, and squares toward the attack step rectangle indicate this dependency within the attack execution graph. An arrow pointing from the attack step rectangle indicates how executing this attack step can change the model state. For example, an arrow from an attack step rectangle pointing to a particular system access square indicates that executing this attack step can affect the adversary's access to this particular system access domain.

Figure 4 provides a closer look at one attack step and its precondition dependency on certain pieces of system access, system knowledge, and/or attack skills. The exact nature of the precondition dependency and the attack outcome effects is specified within the attack step definition, as described in the following section. The attack execution graph picture reveals the overall structure of possible system attack paths, but the creation of an executable model to simulate adversaries attacking a system requires more detailed information about each attack step.

5.2 Attack Step Definition

Each attack step within the attack execution graph is precisely defined to enable the simulation of an adversary's attack decisions and the computation of the likely outcome of attack attempts. An attack step definition includes the *attack precondition, execution time, cost, set of outcomes, outcome distribution, detection distribution, payoff*, and *state*

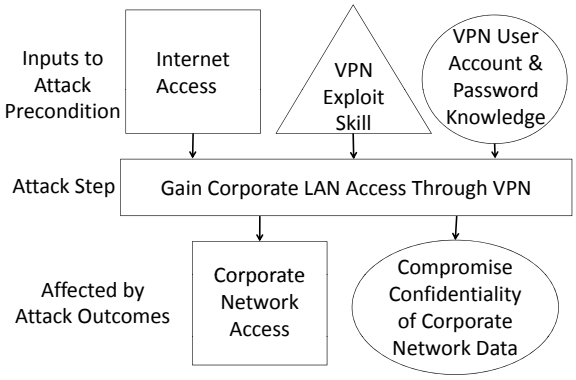

Figure 4: An Example Attack Step Depicted with Precondition and Outcome

variable updates. For some attack step parameters, a utility scale is constructed in which the least desirable value (from the adversary perspective) is assigned a utility of 0, and the most desirable value is assigned a utility of 1. A utility function specifies the translation from the raw scale to the utility scale [0,1].

To illustrate ADVISE attack step characterization, as each parameter is defined, we apply the method to the attack step "Gain Corporate Network Access Through VPN" from the SCADA system example. The ADVISE method stipulates that the security analyst gather the following information about each attack step.

The *attack precondition* precisely states the attack skill levels, system knowledge, and system access that the adversary must possess before attempting the attack step. The attack precondition is specified as a Boolean expression. If the precondition evaluates to FALSE, the adversary will not attempt the attack step. If the precondition evaluates to TRUE, the adversary may attempt the attack step. A full explanation of the factors that influence the attack attempt decision is provided in Section 7. In the SCADA system example attack step "Gain Corporate Network Access Through VPN," the precondition requires that the adversary have Internet access AND either a certain level of skill in exploiting VPN software OR knowledge of an existing VPN user account and password. This precondition is expressed as $(InternetAccess)\&\&((VPNSoftwareExploitSkill > 0.6)||(VPNUserAccountAndPasswordKnown))$.

The *execution time* describes the time needed for the adversary to execute the attack step. The execution time is a random variable defined by a probability distribution function over the positive real numbers. For example, the execution time for an attack step could be normally distributed with mean 300 minutes and variance 50 minutes (but truncating the distribution at time zero to ensure the execution time is always non-negative). The execution time probability distribution may be a function of other values in the model, such as the attack skill level of the adversary.

The *cost* captures the total resource cost to the adversary for attempting the attack step. The costs are incurred whether or not the attack succeeds. A utility scale is constructed for cost by defining the endpoints: the minimum and maximum possible costs in this model. The cost is a random variable defined by a probability distribution function over the utility range [0,1]. For a static attack attempt cost, a deterministic distribution is used. For example, the attack attempt cost could be a deterministic \$600, which could be assigned a utility value of 0.2.

The *set of outcomes* defines the nature of the possible outcomes after an adversary attempts the attack step. For the example attack step "Gain Corporate Network Access Through VPN," the possible outcomes are success (adversary gains access to corporate network), quiet failure (adversary does not gain access but leaves no evidence of the attempt), or noisy failure (adversary does not gain access and leaves evidence of the failed attack attempt). The model may include any positive integer finite number of outcomes for each attack step.

The *outcome distribution* specifies the likelihood of each outcome in the set of outcomes. Each outcome is assigned a probability (a real number between zero and one, inclusive), and the sum of the probabilities of all outcomes equals one. This outcome probability distribution can be a function of other values in the model, such as the attack skill level of the adversary. For the example attack step, an outcome distribution could look as follows: $P[success] = 0.15$, $P[quiet\ failure] = 0.60$, and $P[noisy\ failure] = 0.25$.

The *detection distribution* specifies the likelihood that the system detects the attack step execution. A utility scale is constructed over the detection probability range [0,1]. Note that the utility scale also has the range [0,1], and a utility function translates between a detection probability and its utility value. Detection is then a random variable defined by a probability distribution function over the utility range [0,1]. The detection probability distribution may be a function of other values in the model, such as the attack skill level of the adversary. For the example attack step, a detection distribution could look as follows: for a successful outcome $P[detected] = 0.05$, which is equivalent to *utility* 0.95; for a quiet failure outcome $P[detected] = 0.05$, which is equivalent to *utility* 0.95; and for a noisy failure outcome $P[detected] = 0.8$, which is equivalent to *utility* 0.2.

The *payoff* describes the value to the adversary of achieving a particular attack step outcome. A utility scale is constructed for payoff by defining the endpoints: the minimum and maximum possible payoffs in this model. Undesirable outcomes may have negative payoff values. The payoff is a random variable defined by a probability distribution function over the utility range [0,1]. Each possible attack step outcome is assigned a payoff random variable, and this random variable may be a function of other values in the model. For the example attack step, a payoff distribution could resemble the following: for a successful outcome $Payoff = \$1000$, which is represented as *utility* 1.0; for a quiet failure outcome $Payoff = \$0$, which is represented as *utility* 0.5; and for a noisy failure $Payoff = -\$600$, which is represented as *utility* 0.2.

Finally, the *state variable updates* define how the state of the model changes due to a particular attack step outcome. A state variable update consists of a set of pairs <*state variable, new state variable value*>, where the particular state variable determines the range of possible values. For the example attack step, the success outcome could result in the adversary gaining access to the corporate network and its data, which is expressed as $<CorporateLANAccess, 1>$

and <*CompromiseConfidentialityCorporateData*, 1>. The noisy failure outcome could result in the system administrators disabling the compromised VPN account, which is expressed as <*VPNAccountAndPasswordKnown*, 0>.

All of this collected information enables the ADVISE method to produce an executable model to simulate an adversary or set of adversaries attempting attacks against a system. The input data can be either empirical data measured from similar past attacks or subjective data produced by subject matter experts capable of estimating the input values. Any thorough analysis should include a sensitivity analysis of the input data regardless of the source.

6. METRICS SPECIFICATION

The ADVISE method enables security analysts to specify customized metrics relevant to the security objectives of a particular system. The metrics specification determines what output is collected from the model. Figure 1 shows that the security decision question determines which security metrics are needed and that the metrics specification provides input for generating the executable model.

Our method produces quantitative metrics that are best used for relative comparisons rather than absolute measurements. Model output can be difficult to validate as an absolute measurement, but relative assessments are often sufficient for supporting security decisions. A system assessment is usually either system-focused or adversary-focused.

A system-focused security assessment compares the security strength of different system configurations. An initial assessment is performed on the baseline system. Then the system configuration is modified, the security assessment is repeated, and the results are compared with the baseline system assessment. Configuration changes can affect the system architecture (i.e., the arrangement and connection of components) or the individual components (e.g., firewall settings or VPN software version). Security analysts can choose to test the security of the system against a single adversary or multiple different adversaries.

Metrics for a system-focused security assessment may include the average time for the adversary to achieve a particular attack goal or the most likely attack path (i.e., series of attack steps) used to achieve a particular attack goal. Other variants are possible, such as the probability of an attack goal being achieved by time t or which attack goal is likely to be achieved first when there are multiple attack goals.

An adversary-focused security assessment examines how changes in the characterization of an adversary or differences between adversaries affect the apparent security strength of the system. Typically, the system configuration remains constant as the adversary characteristics are varied. The adversary's attack skill levels, attack goals, attack preferences, initial system access, and/or initial system knowledge may be varied.

When considering multiple adversaries, metrics may include determining which adversary is able to achieve a particular attack goal in the shortest average time or which attack path is most likely when considering all adversaries. An assessment may reveal that different adversaries are likely to choose different attack paths into a system based on their different strengths and attack preferences. Metrics may also measure the total average cost for an adversary to achieve a particular attack goal.

In the SCADA system example, one relevant security objective is to protect the availability of substation network devices (denoted as an attack target in Figure 2). To determine how to best protect the availability of the devices, a security analyst could conduct a system-focused security assessment and examine how modifying the substation network wireless access security settings and the device security settings impacts the average time required for an adversary to make the device unavailable.

To determine the relative advantage of an insider adversary over an external adversary, the security analyst could perform an adversary-focused security assessment. The insider adversary would possess more initial system access (such as local physical access to the substation) and more initial system knowledge (such as wireless network passwords) than an external adversary. The analyst could study the differences between the two adversaries in the average total attack cost and the average time to make the substation device unavailable. After determining the magnitude of the insider threat, the security analyst could also use the model to assess the effectiveness of security mechanisms intended to protect against the insider threat.

7. EXECUTION OF ADVERSARY ATTACK BEHAVIOR MODEL

Figure 1 shows that an executable model is generated from the information gathered in the adversary characterization, system characterization, and metrics specification. To practically analyze the security of a system using the ADVISE method, we convert the characterization information into an executable simulation model and produce security metrics from the model outputs. The ADVISE characterization is precise enough that the conversion process can be fully automated.

The executable model simulates an adversary (or set of adversaries) attacking a system, following the attack cycle illustrated in Figure 5. The following discussion will explain the attack cycle for a single adversary. During each run of the simulation, an adversary repeatedly chooses and attempts an attack step and then succeeds or fails.

An adversary may be faced with several potential attack step options. The adversary must first determine which attack steps are available options and then determine which available attack step option is most attractive. The success of an attempt is determined by the capability of the adversary to execute such an attack step and the system's ability to defend against that type of attack step. We now examine the three steps of the adversary attack cycle.

7.1 Attack Step Precondition Evaluation

The first step in the attack cycle is the attack step precondition evaluation, as shown in Figure 5. Before attempting an attack step, an adversary must possess some minimum level of system access, system knowledge, and attack skill. Within the security model, an attack step precondition formally states the minimum combination of specific access, knowledge, and skills needed, as perceived by the adversary. The attack step precondition is a necessary, but not sufficient, condition for an adversary to attempt an attack step.

The attack step precondition for each attack step is specified directly during the system characterization, as described in Section 5. The precondition can be a function of model

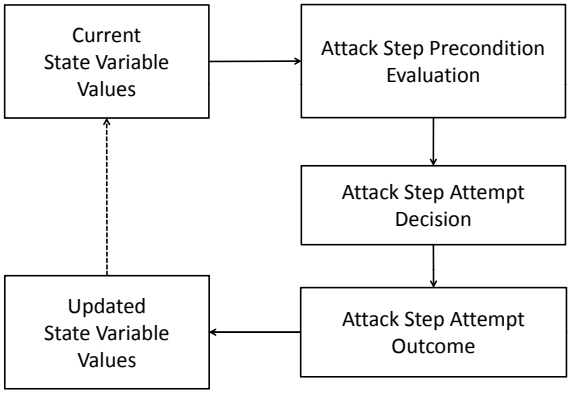

Figure 5: Execution of Adversary Attack Behavior Model

state variables. For example, when the attack step consists of an adversary gaining corporate network access from the Internet through the company's VPN, the precondition might specify that the attacker must possess Internet access and either knowledge of VPN account log-in information or VPN software exploit skill. An adversary who does not meet the precondition requirements will not attempt the attack step. If the precondition is met, then the adversary may or may not attempt the attack, depending on how the attractiveness of this attack step compares with others.

7.2 Attack Step Attempt Decision

The second step in the attack cycle is the attack step attempt decision, as shown in Figure 5. After the adversary has the checked the precondition for each attack step in the attack execution graph, the adversary chooses to attempt one (or some subset) of the available attack steps for which the precondition was satisfied. Within the security model, this choice is represented by the probabilities of the adversary attempting each attack step. Note that the probability of attempt is implicitly zero for the attack steps for which the adversary does not meet the precondition.

The probability estimate that an adversary will attempt a specific attack includes an implicit assumption of the attack step decision cycle time. For some analyses, the time period may be as long as a decade; in other instances, the time period may be a few minutes or even approaching instantaneity.

An adversary examines the available attack options and chooses one to attempt next. The adversary's probability of attempting a particular attack step depends on the relative attractiveness of that attack compared with other attack step options. The adversary first considers all available attack step options, including the option to attempt no attack (the "do-nothing" attack step). The adversary then rates the attractiveness of each option using his or her personal attack goals and attack preferences. Recall that the attack goals and attack preferences were characterized in Section 4.

After the available attack step options have been rated with respect to attractiveness to a particular adversary, the adversary chooses one attack step to attempt. In certain circumstances, the "do-nothing" attack may be the most attractive option. Although the "do-nothing" attack step has no payoff, it also has zero cost, no probability of detection, and no probability of failing. In fact, when the "do-nothing" attack is consistently the most attractive attack step option for adversaries, this is a sign of a strong deterrence defense against that adversary. This situation means that the available attack options are too costly, with too little payoff, too high a probability of detection, and too low a probability of success, in the opinion of the attackers.

7.3 Attack Step Attempt Outcome

The third step in the attack cycle is the attack step attempt outcome, as shown in Figure 5. After the adversary has chosen a particular attack step to attempt, the model must determine the outcome of the attack attempt. Most attack steps have at least two outcomes—success and failure—but there may be other outcomes (e.g., multiple failures modes may be possible).

Within the security model, the attack step attempt outcome is quantified as the probability of the adversary successfully executing the attack step given that it has been attempted. This probability is computed based on the balance of the attack skill of the adversary versus the defensive strength of the system. The defeat of strong system defenses requires the adversary to possess more advanced attack skills.

The outcome of an attack step attempt is determined by the outcome distribution, whose shape is influenced by the attack skills of the adversary and the defense skills of the system. There are many ways to define the outcome distribution. One way is to create a probability graph for a fixed defense skill level, as shown in Figure 6. At each adversary skill level, the graph shows the outcome probabilities for each possible outcome. This example attack step has three possible attack outcomes: noisy failure, quiet failure, or success. If the adversary possesses less than attack skill level X, the probability of success is zero. If the skill level is between X and Y, increases in attack skill result in increases in the probability of success. If the skill level is above Y, additional increases in attack skill do not further increase the probability of success. In this example, the probability of failure (one minus the probability of success) is divided evenly between noisy failure and quiet failure. Figure 6 shows one possible attack outcome probability graph; many different types of graphs are possible.

8. FUTURE WORK

Future work on this project will include case studies with real system data and the development of security model validation methods. We will apply the ADVISE method to evaluate the security of a specific company's system architecture. We will use secondary security assessment methods to show that the ADVISE method aggregates system and adversary data in a way that produces reliable security metrics.

Future work also includes extending the basic ADVISE method in several ways. One possible extension is to allow what is currently constant during model simulations (attack skills, attack goals, and attack preferences) to vary; this could be useful for analyses over longer time periods. For example, when considering longer time scales (i.e., attacks developed and executed over months or years, instead of hours or days), the adversaries may invest in attack step

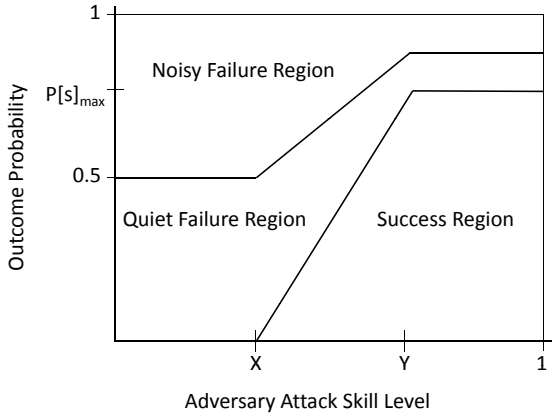

Figure 6: An Attack Outcome Probability Graph as a Function of Adversary Attack Skill Level for a Fixed System Defense Skill Level

skill development, which could boost their attack skill level values.

The basic executable security model can also be extended by the addition of algorithms to simulate dynamic defense behavior. These algorithms would determine when and how response steps should be deployed. Analogous to the adversary attack step precondition, there is a response step precondition to detect when a response may need to be deployed. The response step deployment decision and the outcome of the response step deployment are two other components of the dynamic defense behavior. When dynamic defense behavior is implemented in a security model, the model output can be used to help evaluate the relative effectiveness of different dynamic defense strategies.

In the original model, when an adversary has a nonzero probability of succeeding in an attack, the adversary will reach the attack goals eventually. Modeling dynamic defense behavior makes the steady state of the model more insightful. Steady state security metrics could include the fraction of time that the system spends in a compromised state, which assesses the effectiveness of the response and recovery mechanisms in the system as well as the static defense mechanisms. For example, a steady state security metric could assess the average time length between when the adversary succeeds in crashing the SCADA server and when the system dynamic defenses detect the crash and bring the server back online.

9. CONCLUSION

To obtain a high-level holistic security assessment from a collection of low-level pieces of information, we propose the ADversary VIew Security Evaluation (ADVISE) method to quantitatively evaluate a system's security. This method guides a security analyst in creating an executable state-based security model of the system. Our method includes precise adversary and system characterization formats, a process to generate an executable model from the characterization data, and simulation algorithms for computing adversary attack decisions and the probability that an attack attempt will be successful. One key component of our approach is the inclusion of adversary attack behavior models. A security analyst can generate security metrics data by running discrete-event simulations of the adversaries attacking the system. A security analysis tool using the ADVISE method is currently under development.

10. ACKNOWLEDGMENTS

The work depicted here is performed, in part, with funding from the Department of Homeland Security under contract FA8750-09-C-0039 with the Air Force Research Laboratory. We particularly wish to thank Douglas Maughan, Program Manager, Cyber Security R&D Center, Department of Homeland Security Science and Technology Directorate.

11. REFERENCES

[1] D. Buckshaw, G. Parnell, W. Unkenholz, D. Parks, J. Wallner, and O. S. Saydjari. Mission oriented risk and design analysis of critical information systems. *Military Operations Research*, 10(2):19–38, 2005.

[2] M. Dacier and Y. Deswarte. Privilege graph: an extension to the typed access matrix model. In *ESORICS '94: Proceedings of the Third European Symposium on Research in Computer Security*, pages 319–334, London, UK, 1994. Springer-Verlag.

[3] M. Dacier, Y. Deswarte, and M. Kaâniche. Models and tools for quantitative assessment of operational security. pages 177–186, 1996.

[4] S. Evans and J. Wallner. Risk based security engineering through the eyes of the adversary. In *Proceedings of the 2005 IEEE Workshop on Information Assurance*. United States Military Academy, West Point, NY, June 2005.

[5] R. Ortalo, Y. Deswarte, and M. Kaâniche. Experimenting with quantitative evaluation tools for monitoring operational security. *IEEE Trans. Softw. Eng.*, 25(5):633–650, 1999.

[6] O. Sheyner, J. Haines, S. Jha, R. Lippmann, and J. M. Wing. Automated generation and analysis of attack graphs. In *SP '02: Proceedings of the 2002 IEEE Symposium on Security and Privacy*, page 273, Washington, DC, USA, 2002. IEEE Computer Society.

[7] O. M. Sheyner. *Scenario graphs and attack graphs*. PhD thesis, Pittsburgh, PA, USA, 2004. Chair-Wing, Jeannette.

[8] G. Stoneburner, A. Goguen, and A. Feringa. *Risk Management Guide for Information Technology Systems (SP 800-30)*. National Institute of Standards and Technology, Gaithersburg, Maryland, July 2002.

[9] L. Wang, A. Singhal, and S. Jajodia. Toward measuring network security using attack graphs. In *QoP '07: Proceedings of the 2007 ACM workshop on Quality of protection*, pages 49–54, New York, NY, USA, 2007. ACM.

[10] B. Whiteman. Network risk assessment tool (NRAT). *IAnewsletter*, 11(1):4–8, Spring 2008.