The impact of predicting attacker tools in security risk assessments

Ezequiel Gutesman
CoreLabs, Core Security Technologies
Humboldt 1967, 1er piso
Cda. de Buenos Aires, Argentina
ezequiel.gutesman@coresecurity.com

Ariel Waissbein
CoreLabs, Core Security Technologies
Humboldt 1967, 1er piso
Cda. de Buenos Aires, Argentina
ariel.waissbein@coresecurity.com

ABSTRACT

In this article we present a new model and method for anticipating attacks against the networked computing infrastructures for an organization. The model combines existing models for cyber-attack representation and attack planning, with a new approach at anticipating what tools (e.g., exploits) an unknown attacker may hold. This approach allows us to take statistical samples of exploits an attacker could hold. Combining attack simulation and attack planning with a sampling algorithm, we show that we are able to derive information that can be used to anticipate threats. As a result, the model allows predictive risk assessments improving over older reactive models. We further present a software solution that implements this model by introducing a sampling algorithm and combining it with software for simulating attacks and automatically planning them. Finally, using this tool, we derive a mechanism to compute a security metric that describes which is the most fragile computer in the network.

Categories and Subject Descriptors

K.6.5 [**Security and Protection**]: Invasive software and Unauthorized access

Keywords

Security testing, risk assessment, metrics, networked computing infrastructure, cyber-crime

1. INTRODUCTION

The networked computing infrastructures for an organization hold some of the most important assets that the security-aware organization is bound to protect. In the past years, attackers to these networks have professionalized ([4]) and their tools of trade have diversified ([13],[18], [3]).

No matter what are the security defenses in place for a network, the community agrees that it will have security holes –eventually. In order to anticipate an attack and put in place the correct strategy, the *de facto* solution is to periodically perform one form of security assessment that will identify and prioritize the threats to which the network is exposed. The assessments will identify open threats and trends that will be used to manage IT risk of the organization.

There are many forms of Security Testing and Risk Assessment that range from methodologies such as vulnerability scanning to penetration testing (cf. [9]), to more formal frameworks such as COBIT ([7]) or FAIR ([8]). These methodologies result in varying precisions and, in general, have strengths and deficiencies. There is a deficiency common to all of them, which is that they are reactive, in the sense that they identify open threats and help to analyze their impact. However, threats are not bound to appear in such an orderly fashion. For example, an attacker could hold a so-called 0-day exploit (which the assessment team cannot identify *a priori*) and this will not be taken into account in the assessment (cf. [17]). Or, it may be the case that network users and the assessment process are synchronized in such a way that the assessment never takes place in the window of time when certain threats are open.

In this article we introduce a predictive risk-assessment model and platform for networked computing infrastructure security threats that solves the above problem. This platform allows us to anticipate, and therefore analyze the impact of, the potential attacks to a network that arise from present trends in attacker tools (and methods). Within this model, one can define the specifics describing the security-relevant details of a network, the tools that the attacker has access to in order to execute an attack, and the objective that these attackers may have. Next, the platform allows simulating the attacks that the attackers may be able to fulfill using different sets of tools (which are derived from a probability distribution that models trends) determining which objectives are easier to get and the attack path that leads to them. The model, thus, has several interesting applications in that it allows a predictive (e.g., non-reactive) approach to security assessment.

Moreover, we designed a methodology which combines some existing and novel tools for importing the specifics of a network into our model, defining what are the assets or attacker objectives that we are interested in guarding, and having the tool compute what are the threats that an attacker could exercise when using these tools. This is combined with a newly devised and implemented sampling algorithm that describes

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. CSIIRW '10, April 21-23, Oak Ridge, Tennessee, USA Copyright © 2010 ACM 978-1-4503-0017-9 ... $5.00

the list of tools that may be available to the attacker. Basically, the method will simulate attacks against a pre-defined network to check if a pre-defined set of (attacker's) objectives can be achieved when varying the tools (or actions) available to the attacker.

An application of this would be to fix a network and objectives, and iterate over the sampling process, deciding at each case if the sampled attacker can or cannot be successful, and the steps that he would take in this case. Aggregating this information allows us to have a better grasp of what are the threats that this network is exposed to even when we assume that we cannot guess what tools will the attacker hold, and also determining the components, which when vulnerable, lead to the most critical attacks.

For example, let us say that the security officer for a network infrastructure uses this methodology to test how exposed is the database that holds the credit card information for clients. After running a few thousand simulations, he discovers that the computer that is present in most of the attacks is a server sitting in the DMZ that is hardly patched since it runs a non-critical service: a custom built FTP server. The security officer will then realize that he must eliminate this threat immediately and that it can be easily done.

Finally, we introduce a metric that we derive from the above example. Namely, the metric assigns a fragility order to each of the computers in the infrastructure. The fragility order is computed by counting for each computer in the network in how many of the simulated attacks, which achieved the objectives, does it appear. The metric is then computed by simply ordering these computers from the one that appeared in more attacks to the one that appeared in less of them. The precision of the metric depends directly on how well does the sampling algorithm approximate real attackers, (the number of times we iterate and the precision of the simulation model).

The rest of the paper is structured as follows. Section 2 introduces the model and architecture of the implementation. The implementation is described in Section 3 and briefly analyzed in Section 4. Finally, Section 5 discusses future work.

2. A MODEL FOR RISK ASSESSMENT

The model consists in three components. First, a *network simulator*, capable of simulating networks and arbitrary attacks against the simulated networks. Second, an *attack planner* that shares the network attack model with the simulator and which given an objective, finds a list of steps to reach it when possible. Third, a *distribution of probabilities* that describes for different sorts of attackers what are the tools available to them.

A typical assessment cycle with the described kind of tools could be then outlined as follows: The first task is to import the information specifics needed into the simulator, this information can be collected from the network with automatic information gathering tools like [5] or through a manual penetration test process. Next, we take a sample of attacker tools using a pre-defined probability distribution that describes the tools that a given attacker is likely to hold.

Finally, we proceed to define one or more objectives, which are actions that the attacker may carry out successfully and that are supported by the simulator. For each attacker's tool sample that we draw, our planner and simulator can answer whether the objective can be achieved by the attacker. Further, we get the "attack path" which includes the systems compromised during the attack and all the attacker's actions.

3. MODEL IMPLEMENTATION

The network simulation can be done through an attack graph model (see [10] and [16]) or using the probabilistic simulator described in [6]. We choose the latter, which was built to simulate cyber-attacks against arbitrary target scenarios taking the attacker's standpoint, e.g., it allows the user to simulate an attack step by step, using simulated versions of the tools that the attacker holds. The scenarios might be composed by network devices, hardware devices, software applications, users and other parties. One of the most important features of this simulator is that it can simulate vulnerabilities and exploits, including 0-days, services running on hosts and even system calls executed in each simulated machine.

Different options exist while choosing an automated planner [15]. Some planners work over the attack graph trying to find the shortest path that leads to an objective in a deterministic fashion, other planners like the one used for the implementation [14] use probabilities in order to bound the explosion of possible states. We use [14]. The probabilistic attack planner takes a scenario, a set of exploits and modules (which are part of the attacker's arsenal), the probability of success for each module and an objective. It then calculates the plan for obtaining the desired objective based on the provided probabilities.

There is no publicly known tool for anticipating what tools may an unknown attacker hold. Explicitly, a tool for sampling a random variable that describes the tools (exploits, information gathering tools, post-exploitation tools) that a specific attacker holds.

We define a simple sampling algorithm which can be easily replaced. Assume we have a list of all the software installed in the network and its versions. We collect the information about publicly [1, 2] available exploits and count them as if the attacker had them with some probability. We then fix a number for the probability of an attacker having a 0-day exploit for the software versions from the software list. For each of these, we give a "simulated 0-day exploit" to the attacker. This information is added to the attack simulator and attack the planner.

4. ANALYSIS OF THE IMPLEMENTATION

We will now analyze the cost behind each step needed in order to perform an assessment. We provide experimental explanations as no formal estimates are available.

Importing the network scenario with automated tools [5] or building it by hand and defining the attacker's objectives are tedious tasks but they only have to be carried once. This is not in the scope of this work.

The simulator[6], has been tested and we have observed can handle 2000+ machines running on a single simulation. The planner[14] was tested with 5000 attacker modules and a few hundred machines (around 300) and it takes a few seconds to solve the given objectives.

To count how many samples can be drawn for each fixed number of 0-day exploits we can use the combinatorial number of the number of applications and the number of 0-day exploits (in 2009 there were 4600 vulnerabilities reported and 7200 in 2008 [2]). Let us assume there are 1000 applications including different versions in a single company. Let us assume the attacker has 3 0-day exploits. With this numbers we have $\binom{1000}{3} \approx 10^8$ combinations of applications available to the attacker, since we want to cover all possible 0-day vulnerabilities we might be facing. If one planner run takes 3 seconds to finish then covering all possible combinations would take approximately 15.7 years.

We coped with this limitation by performing Monte Carlo experiments [12] which allow the instantiation of a big amount of possibilities for a given parameter with some fixed probability distributions.

To compute the metric that assigns a fragility order to each of the computers in the infrastructure we proceed as follows. First, we do a tally that assigns to each computer in the network the number of successful attacks it appeared in during the Monte Carlo simulation. Next we order them from higher to lower and assigning the metric value 1 to the highest tally value, the value 2 to the computer in second place, etc.

4.1 Experimental Results

Since the presented work is part of and on-going research line we are still working on a fully-fledged prototype of the solution which includes integrating different planning technologies with the simulator and defining goals inside the simulator which can reflect the proposed methodology.

In order to evaluate examples we developed a simulation which implements the entire cycle with different probabilistic parameters.

First, we build a network with a fixed number of hosts and applications installed in each host (as described in section 3). Each host can have a different set of applications installed and the connection between hosts is made through the installed applications (e.g., host A is connected to host B through application *app1*, we denote this connection with $A \rightarrow B.app1$). There are two probability distributions which are used to randomly draw which connections and applications are to be added to the scenario. Over this network we choose a *starting point* for the attacker and a *target machine* which represents his objective.

Second, we proceed by assigning vulnerabilities in a random fashion. For the experiment shown in Figure 1 we combined the probability for a given application to have a vulnerability with the probability of being installed depending on the type of application that could be a server or a client application. This means that the probability of being able to connect from one host to another already includes the probability

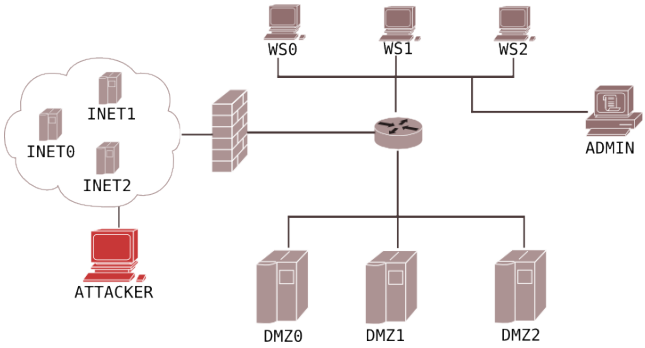

Figure 1: Experiment Scenario

that represents a vulnerability in *app1*. Thus, the correct way to illustrate the connectivity would be $A \xrightarrow{p} B.app1$, meaning "Host A can connect **and potentially exploit** a vulnerability in $B.app1$ with probability p".

After setting up the scenario the experiment continues by sampling the attacker tools. Two Gaussian distributions were used. The one that samples the exploitability for a certain application by a 0-day exploit has $\mu = 0.3$ $\sigma = 0.1$ and for publicly available exploits $\mu = 0.03$ $\sigma = 0.01$. These mean values can be interpreted as follows: "30 of each 100 installed applications can be exploited by a 0-day (if it exists) and 3 in 100 with a publicly available exploit". These values were the ones used for the experiment but can be easily modified. After defining these values we toss a coin with probability p_{public} to give the attacker public exploits (e.g., $p_{public} = 1 \Rightarrow$ attacker has all the public exploits) and with probability p_{0-day} to give him a 0-day exploit.

The experiment proceeds by iterating and sampling attacker tools as described above. For each iteration we calculate possible attack paths with the previously set probabilities, counting for each host, in how many successful attack paths it appeared. Figure 1 shows the topology used to run the experiment.

The experiment scenario was defined with some restrictions in order to make the simulation more realistic. The machine identified as $ATTACKER$ can connect to machines in the internet zone (cloud) and $DMZx$ and only to server applications installed on them. A server identified as $INETx$ or $DMZx$ can connect to any machine, but only to client applications (this could represent client-side attacks). The $ADMIN$ can connect to any application installed on any machine inside his network ($DMZx$ or WSx) and only to server applications of the $INETx$ machines. Finally, workstations identified as WSx can connect to server applications installed on $INETx$ or $DMZx$ servers, and to any kind of application (server or client) installed on any machine in its own segment (WSx and $ADMIN$).

We run 100000 iterations varying the 0-day probability, which represents how likely is the attacker to have a 0-day exploit for a given vulnerability present in a specific application. We used values inside $[0, 1]$ interval. Figure 2 shows the percentage of successful attack paths where each host was compromised as an intermediate step towards the pre-defined goal.

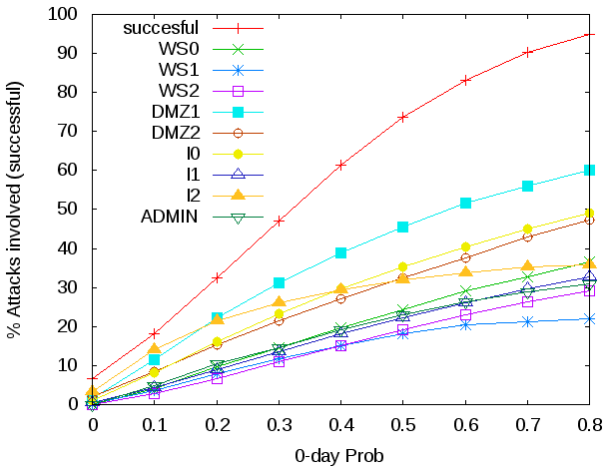

Figure 2: Simulation results. 100000 iterations. $p_{0-day} \in [0, 1]$.

In Figure 2 we observe that as the attacker becomes more likely to hold a 0-day exploit for any vulnerability, the relative relevance of each machine inside the same network changes. For example, machine $I2$ decreases its relevance compared to $DMZ2$, $DMZ1$ and $I0$. This phenomena is related to which applications are installed in each host. The percentage of successful attack paths is represented by the reference *successful* in Figure 2. Some reference values have also been included in Table 1.

p_{0-day}	Successful Attacks
0.01	7.61%
0.10	18.26%
0.20	32.37%
0.50	73.59%
0.70	90.27%
0.80	94.87%
0.90	97.66%
1.00	99.11%

Table 1: Successful attacks for 100000 Monte Carlo simulation

5. CONCLUSIONS AND FUTURE WORK

We have presented three different novel results: a model for doing predictive risk-assessment, an assessment method that underlies this model and a software implementation that allows us to carry out this method, and finally a new security metric.

The presented work was partially motivated by a research group formed to work in security metrics and attacker-centric metrics (cf. [11]). The search is for an estimator that can be calculated based on attack simulations that take the standpoint of the attacker.

There is a good space for improving over this metric, and in general the technologies we describe. The plan is to use our method and metric to gather information from realistic scenarios.

In order to simulate accurately attackers, we need to improve over our sampling algorithm and find how to better model the other tools that attackers use (e.g., how to model their information gathering tools and the information that is publicly available).

While most of our work is related to risk assessment, we understand the need to consider the whole risk management process and are interested in using our model and tools for analyzing risk mitigation strategies (e.g., the impact of changing the network topology).

6. REFERENCES

[1] Bugtraq mailing list - `www.securityfocus.com`.

[2] Common vulnerability ennumeration - `http://cve.mitre.org/`.

[3] Jerrie Abella. Hackers deface 5th govt web site, mock automated polls. GMANews.tv on 11/1/2010 at `http://www.gmanews.tv/story/181239/hackers-deface-5th-govt-web-site-mock-may-polls` (Last checked 17/3/2010), 2010.

[4] Pedram Amini. Mostrame la guita! adventures in buying vulnerabilities. In *Ekoparty 2009. Buenos Aires, Argentina. September 17-18.*, 2009.

[5] Yair Bartal, Alain J. Mayer, Kobbi Nissim, and Avishai Wool. Firmato: A novel firewall management toolkit. *ACM Trans. Comput. Syst.*, 22(4):381–420, 2004.

[6] Ariel Futoransky, Fernando Miranda, José Orlicki, and Carlos Sarraute. Simulating cyber-attacks for fun and profit. In *2nd Internation Conference on Simulation Tools and Techniques (SIMUTools'09)*, 2009.

[7] Governace Institute. *COBIT 4.1 - Framework Control Objectives Management Guidelines Maturity Models*, 4.1 edition, 2009.

[8] Jack A. Jones. An introduction to factor analysis of information risk (fair). Technical report, Risk Management Insight, 2008.

[9] Douglas J. Landoll. *The Security Risk Assessment Handbook*. CRC Press, Inc., Boca Raton, FL, USA, 2005.

[10] Richard Lippman and Kyle Ingols. An annotated review of past papers on attack graphs. Technical Report ESC-TR-2005-054, MIT Lincoln Laboratory, 2005.

[11] Fernando Miranda and Ezequiel Gutesman. Attacker-centric risk assessment and metrics, 2009.

[12] S. Ulam Nicholas Metropolis. The monte carlo method. *Journal of the American Statistical Association*, 1949.

[13] Joseph Pereira. How credit-card data went out wireless door. Wall Street Journal. May 4, 2007 edition. The online version `http://online.wsj.com/article/SB117824446226991797.html` was last checked 17/3/2010.

[14] Carlos Sarraute. New algorithms for attack planning. September 2009.

[15] Carlos Sarraute and Alejandro Weil. Advances in automated attack planning. In *PacSec Conference, Tokyo, Japan. November 12/13, 2008*, 2008.

[16] Bruce Schneier. Attack trees. Dr. Dobbs, December 1999.

[17] Ariel Waissbein. Your risk is not what it used to be. Toorcon X Conference. September 25, 2008. San Diego, CA, USA, 2008.

[18] Kim Zetter. Google hack attack was ultra sophisticated, new details show. Wired, January 14, 2010. Last accesed online at `http://www.wired.com/threatlevel/2010/01/operation-aurora/` on 17/3/2010, January 2010.