Testbed Automation for Network Security and Security Analytics

Aragats Amirkhanyan

Internet Technologies and Systems
Hasso-Plattner-Institut
aragats.amirkhanyan@hpi.de

The report summarizes my work in the past semester in the IT security team of the chair "Internet Technologies and Systems".

The testbed is an important part of any research work in the area of network security and security analytics. Common testbeds are data and test network environments. The absence and the need to create a new testbed make an overhead of research work and, as a result, this overhead slows down research progress. Therefore, the issue of automation of the testbed creation is a crucial goal to accelerate research progress. In this report I show a method, which I used for solving a problem of the lack of data for the certain research project in the security analytics area.

The goal of the report is to show a method and identify research challenges in solving the problem of testbed data automation for network security and security analytics.

1 Introduction

When researching in network security and security analytics areas, it is very important to have an appropriate testbed. The appropriate testbed includes appropriate data in needed amount and an appropriate test network environment. But often researchers do not have one of them or both of them.

In case with data, the lack of data is a result of the fact that mostly the huge amount of data belongs to industrial companies, but they are usually not able to share them because of security and privacy reasons. The limitation with data makes data for security research is more inaccessible. And even if there is possibility to get such information, this information should be anonymized before exporting for analysis. Therefore, researchers have to simulate scenarios and generate data. And it means that we have new challenges, such as how to simulate a scenario, how to generate data, how to design a scenario based on modeling user behavior.

In case with a test network environment, researchers need to create and configure a test network environment manually. Even if the installation is manual, it is still a challenge, because it is needed to set up a test network environment of enterprise or campus level. It means that we have to set up different complex systems, for instance, a domain controller.

Any test network has its own specifics. In case with research in the area of network security and security analytics, this specifics means that often needed to have a

potentially vulnerable environment and that means that old and vulnerable software applications must be installed. This restriction complicates the ability to use many existing IT automation systems, such as Chef[1] and Puppet[2].

There are several ways to generate data. One of them is to generate synthetic data. This method implies that you know about the structure of data, the correlation between data and you roughly know what the result of analysis will be on the generated data. The advantage of this method is that you do not need to have a test network environment and you can generate any data if you know the structure of them. If the structure is quite simple then this way is more preferable. But if the structure of data and the correlation between data are very complex and varies, then this approach becomes complicated, because in this case, you have to implement algorithms for each type of data and for each correlation.

Another approach to generate needed data is based on the simulation of some activities needed to produce this data [1]. This approach is more complex than the first one in case of simple data, because you need to set up a test network environment before applying the simulation. Despite the fact that this approach is more complex, it is more flexible and it can be applied for generation data with complex structure, because we do not need to care about the structure of data or the correlation between data. We need to care only about the simulation of specific activities. Also, this approach is suitable for real-time research, for example, for generating data for real-time intrusion detection systems.

Within the IT security team of the chair "Internet Technologies and Systems" we also face with problems of the lack of testbeds. We are trying to solve them by automation the process of the testbed creation. In the second section of the report, I show how we solve the problem of the lack of data by the simulation of user behavior.

2 Preparing testbed data to analytics

In the past semester the IT security team of the chair "Internet Technologies and Systems" worked on the project "Machine Learning for Security Analytics powered by SAP HANA". Within this project the team aimed to implement and test the machine learning approach for security analytics based on SAP HANA. Under this approach the team focused on the analysis of user events, particularly login and logout events. To test the machine learning analytics module, we have generated several simple brute-force attacks. Also, there was the idea to try the machine learning approach powered on SAP HANA for security analytics with more sophisticated cases. But the main problem of trying the approach with sophisticated cases is the lack of data. As I mentioned in the introduction section, mostly only industrial companies have such data in needed amount, but they do not share them because of security and privacy reasons. For example, login and logout events contain information about users and this information cannot be passed to third parties. I was faced with the task to set up a test network environment and implement a tool to simulate particular user scenarios, user behavior,

[1] Chef. http://www.getchef.com
[2] Puppet. http://puppetlabs.com

and as a result, generate necessary data for further analysis. This simulation tool must provide capability to simulate normal and abnormal scenarios to be able to analyze user behavior by the machine learning approach to detect anomalies.

Challenges to create such testbed are setting up the domain controller of enterprise or campus level, creating large amount of users, generating random user activities and designing a model of user behavior, especially abnormal behavior.

2.1 Network architecture

For the first challenge, we started with designing and describing the test network. Our test network contains the domain controller (DC) with installed Windows Server 2012, the wiki and database servers with Windows Server 2003 and four client computers with Windows 7 Professional 64-bit. To complete the installation, we created four user accounts. All the above information is presented below in the form of compact lists.

Description of the network:

- 4 client computers with Windows 7 Professional 64-bit
- Domain controller with Widows Server 2012
- Wiki server with Windows Server 2003
- Database server with Windows Server 2003

Users:

- Ivanov
- Petrov
- Smirnov
- Admin

2.2 User behavior

2.2.1 Normal scenario

In our configuration, all users have access to client computers, the wiki server and the database server, but only admin has direct access to the domain controller. In the case of the normal scenario, users log into client computers by their credentials and they do it several times per day. Users Petrov and Smirnov during the workday visit the wiki server, but Ivanov does not visit the wiki server despite that he has access. All users have access to the database server, but no one uses it in the normal scenario. In turn, admin usually logs into to his computer and during the workday he logs into to the domain controller, the database server and the wiki server by the remote desktop connection (RDP). The description of the normal scenario you can see in Figure 1. The lines show all allowed access to computers. The solid lines illustrate access, which are used by users, the dash lines show access, which are not used by users in the normal behavior despite that they are allowed to.

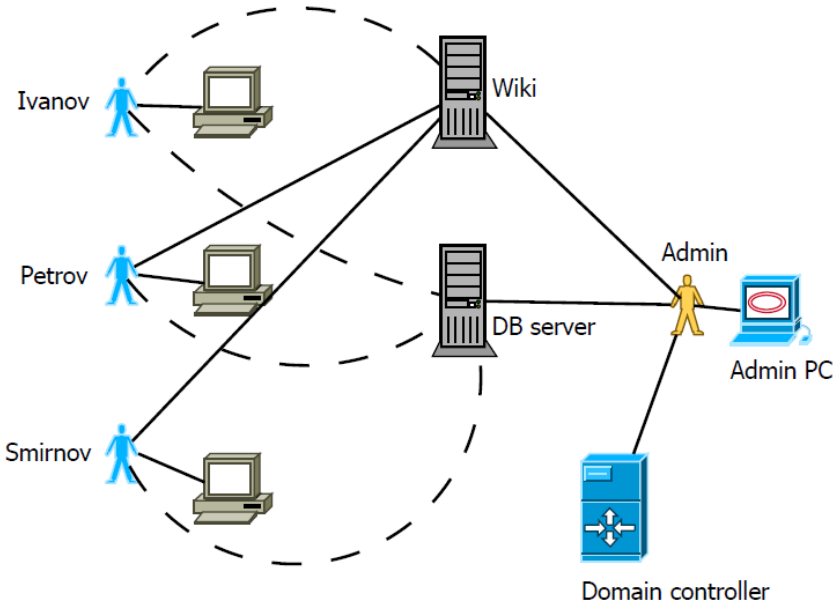

Figure 1: Normal scenario

2.2.2 Abnormal scenario

The abnormal scenario is the scenario that differs from the normal scenario by some unusual but acceptable behaviors. In our case, the abnormal scenario includes additional user behaviors. The first abnormal behavior is that the user Ivanov uses the wiki server. This behavior is abnormal but not suspicious, because other users use it every day. The second abnormal behavior is that the user Petrov uses the database server but in the normal scenario no one uses it. This user behavior is more suspicious and could be determined as an attack. The abnormal scenario is illustrated in Figure 2. The straight solid lines show normal user behavior, the curve solid line from Ivanov to the wiki server shows the first abnormal behavior and the curve solid line from Petrov to the database server shows the second abnormal behavior that is suspicious and could be determined as an attack.

2.3 Implementation

The first part of the implementation is the deployment and the configuration of the test network environment. To set up the network, we used the Future SOC Lab[3] with installed VMware ESXi. Firstly, we created the network to isolate our virtual machines from others hosted on the same ESXi server. Afterwards, all machines were deployed and configured as described in the section 2.1. We used Windows Server 2012 for the domain controller, Windows Server 2003 for the wiki and database servers and Windows 7 Professional 64-bit for client computers.

The architecture of the simulator is illustrated in Figure 3. This simulation tool is

[3]Future SOC Lab. http://hpi.de/en/research/future-soc-lab.html

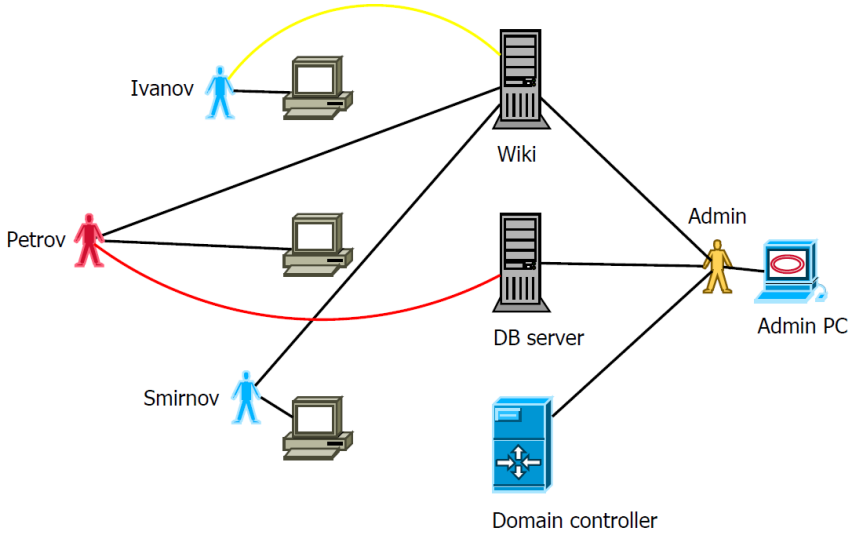

Figure 2: Abnormal scenario

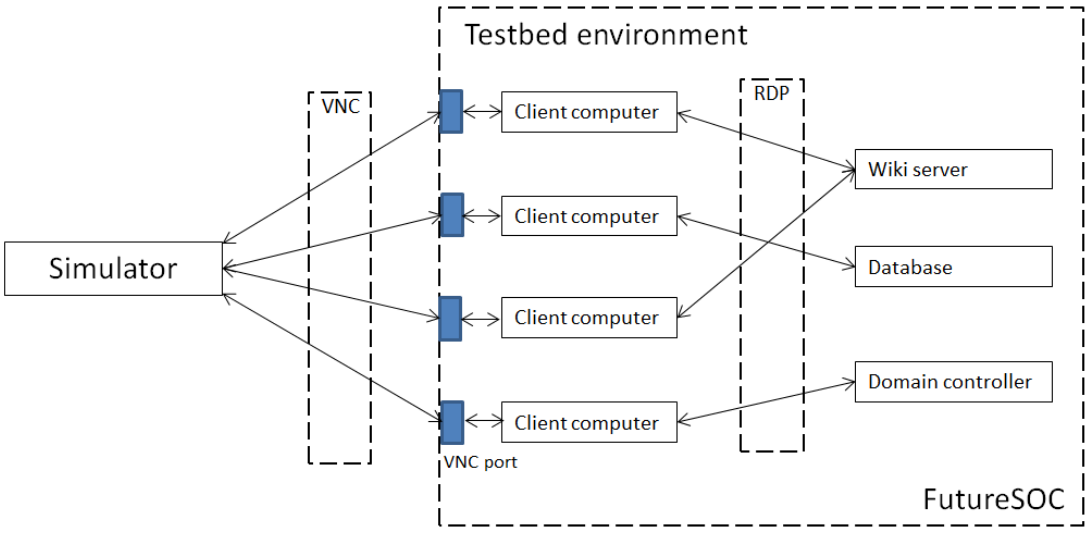

Figure 3: Simulator's architecture

implemented on the python programming language and uses the virtual network computing (VNC) protocol to connect to virtual machines as the ESXi server supports the VNC protocol. But to use the VNC protocol, it is needed to enable VNC on the ESXi server, specify the VNC port for each virtual machine and support the VNC protocol on the application side. To use VNC in the simulator, we used the vncdotool library [3]. This library also can take a screenshot of the remote desktop. We used this screenshot functionality for checking a state of the virtual machine by comparing image histograms with predefined states. Once we determined the current state of the virtual machine, we can specify the activities to be undertaken in each case according to the scenario description, such as sending the *ctrl+alt+delete* command to the virtual machine, entering the username and the password, opening the RDP connection and others. The RDP connection from the client computer to the database and wiki servers

Action	Command
Open the logon window	['ctrl-alt-del', 'alt-w', 'right', 'right', 'right', 'right', 'enter']
Enter username, password and press the enter key	[':' + kwargs['username'], 'tab', ':' + kwargs['password'], 'enter']
Log off	['lsuper-r', ':shutdown /l /f', 'enter']
Run the powershell script to open the RDP connection	['cd /', 'powershell rdp.ps1]

Table 1: Actions and commands

is implemented by invoking the PowerShell script hosted on the client computer. This script can accept parameters and it means that we can use the script to establish the connection with any server by specifying parameters, such as the IP address, the username and the password. In the Table 1 you can see examples of relations between actions that must be undertaken according to the scenario and commands passed to VM to perform it.

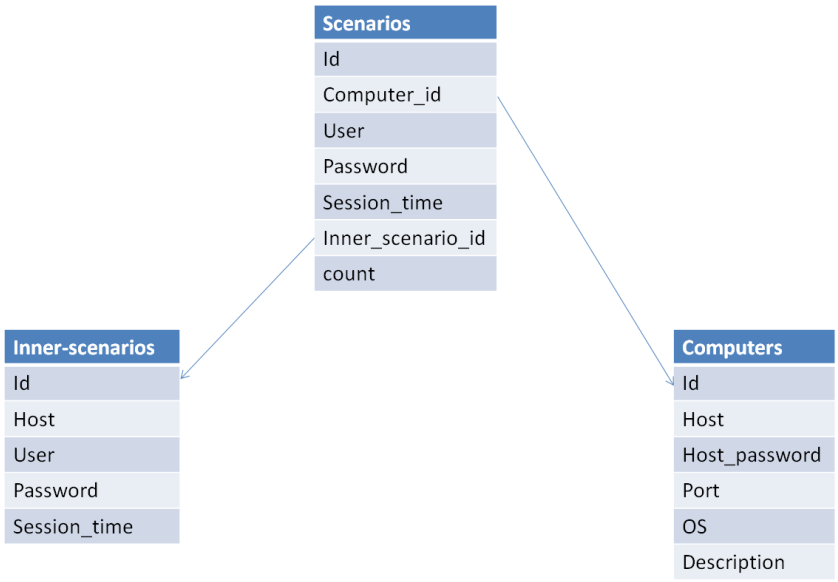

Figure 4: Scenario relations

In this paragraph, I show how we described the scenario of user behavior. The scenario descriptions are stored in *csv* files. There are three *csv* files. The first of them is called *computers.csv*. It describes all computers that are involved in scenarios and it contains the information about the computer identifier, the IP address or the host, the VNC port, the VNC password and the type of operation system. The second file is called *scenarios.csv*. It describes the main user activity, which is the connection to the client computer. The file contains the computer identifier referring to the identifier in the *computer.csv* file, the username, the password, the session time, the count of sessions and the identifier referring to the identifier in the *inner-scenarios.csv* file. The third file

Level	Date and Time	Source	Event...	Task Category
Information	9/4/2014 11:18:15 PM	Security-Auditing	4672	Special Logon
Information	9/4/2014 11:18:14 PM	Security-Auditing	4634	Logoff
Information	9/4/2014 11:18:14 PM	Security-Auditing	4624	Logon
Information	9/4/2014 11:18:14 PM	Security-Auditing	4672	Special Logon
Information	9/4/2014 11:18:13 PM	Security-Auditing	4624	Logon
Information	9/4/2014 11:18:13 PM	Security-Auditing	4672	Special Logon
Information	9/4/2014 11:17:47 PM	Security-Auditing	4624	Logon
Information	9/4/2014 11:17:47 PM	Security-Auditing	4672	Special Logon
Information	9/4/2014 11:17:37 PM	Security-Auditing	4634	Logoff
Information	9/4/2014 11:17:35 PM	Security-Auditing	4624	Logon
Information	9/4/2014 11:17:35 PM	Security-Auditing	4672	Special Logon
Information	9/4/2014 11:17:35 PM	Security-Auditing	4769	Kerberos Service Ticket Operations
Information	9/4/2014 11:17:27 PM	Security-Auditing	4634	Logoff
Information	9/4/2014 11:17:27 PM	Security-Auditing	4624	Logon
Information	9/4/2014 11:17:27 PM	Security-Auditing	4672	Special Logon
Information	9/4/2014 11:17:18 PM	Security-Auditing	4634	Logoff

Figure 5: Active Directory's logs

called *inner-scenarios.csv* describes the actions of users after logging into the client computer. For example, it can describe the connection to the wiki server, database server, domain controller or even the connection to another client computer. The file contains the identifier, the host, the username, the password and the session time. There are two sets of *csv* files. The first set is used to perform the normal scenario and the second is used to perform the abnormal scenario, respectively. The described relations between *csv* files are illustrated in Figure 4.

I have successfully used the simulator to simulate simple normal and abnormal scenarios. The normal scenario takes about 3,5 hours with 41 login and logout events into client computers and 25 RDP connections to the wiki and database servers. The abnormal scenario takes about 5,5 hours with 45 login and logout events to client computers and 34 RDP connections to the wiki and database servers. The result of running the simulator is set of logs in the active directory of the domain controller. This log dataset is presented in Figure 5.

3 Conclusion

I have presented a method of solving a problem of testbed automation for research in the area of network security and security analytics regarding the problem of the lack of data. My proposal is based on the simulation of user behavior. To prove the concept of the method, I provided the implementation of the idea and, additionally, I presented the network architecture used for the simulation, the description of simulated scenarios and the architecture of the implementation. As result, and I have successfully used the simulation tool to generate necessary data for research and presented the example of dataset produced by the simulator.

As future work, I plan to try the simulator with large amount of users, with the network of enterprise or campus level and random user behavior.

References

[1] A. Garg, V. Sankaranarayanan, S. Upadhyaya and K. Kwiat. USim: A User Behavior Simulation Framework for Training and Testing IDSes in GUI Based Systems. In 39th Annual Simulation Symposium, 2006.

[2] Emilie Lundin Barse, Hakan Kvarnstrom and Erland Jonsson. Synthesizing Test Data for Fraud Detection Systems. In 19th Annual Computer Security Applications Conference, 2003.

[3] VNCdotool. https://github.com/sibson/vncdotool.