IDSITER- a platform for automatic based threats detection

Adi Maliyanker *, **Avishalom Jan** * , **Maor Ovadia** * and **Oz maatuk** *

*University of Ariel

Yehuda Grester- lecturer in Ariel University

The offensive cyber environment and the rapid development of technology have always put obstacles to information security specialists and big companies. Private data, Infrastructures and services are all in danger in front of the remote attacker. One of the things made by security experts in a defensive step was setting group of resources which cannot be accessed directly from the outside world. In a response, the attackers created a new type of attacks, automatic instances able to survive without external commands and guide. These instances are based on a fixed deterministic set of activities which can be identified. This research is made in order to find out what can be the features of such instances and building a suitable system able to detect these instances and suggest rules in order to stop them. We also researched whether identifying mechanisms in the instances' activities can reduce the amount of false positives alerts and point with high reasonability on malicious program. We found a bunch of properties which might characterize the automatic attack, but focuses on time intervals while build our system. We used newton's interpolation formula in this field and achieved a relative success when trying identifying the attacks and suggest rules in order to stop them. We describe our work and results in the document.

Idsiter | mechanism | automatic-attacks

Introduction

In the old days, attacks were mostly committed individually, namely when an attacker wanted to steal data about the network or certain organization, his aim was reaching some computer over the network, sending some malware and gaining control over the victim - take by force some vital data or committing some other malicious activity.

As a response, companies spread its vital data over a group of secured computers, without any possibility of access from the outside world (it was not connected directly to the outer world). This way appeared useful but attackers managed to develop a new type of attacks , programs which manage them self without a need for commands from the outside world. They developed malwares able to spread over the network based on some deterministic-fixed mechanisms.

As for the motivation for systematic network attacks, as mentioned before, today many networks (except single computers) are not connected to the Internet, which makes it difficult for the attacker to attack it directly. To do this, the hacker attacks the computers that are connected to the Internet and by operating them to computers that do not see Internet. This is usually done by the attack system. The attack system uses a fixed algorithm that studies the network structure, then spreads continuously on the internal network by various network weaknesses, installs a copy of the attack system, then the attack systems collect information and send it to the parent system. The information collected is concentrated in the original attacker.

As expected, it turned very common when attacking companies as it is not trivial to detect them and prevent them by the resources exist today (antiviruses, firewalls, IDS etc) as it acts like other programs superficially, but when analyzing its essence and propagation its maliciousness may be detected. This insight that there is a malicious activity in the network identified by an ad hoc view, force us to recalculate a route again and evolve a new way of thinking looking on the whole network as one. We researched this field and came out with some insights as described in the article.

our project

[1]

Our research question is whether there are attacks which detected by neither local monitoring nor by network monitoring but can be identified by gathering information from all stations (we focus on hids-s) . We believe that using cross-referencing hids-s outputs, these attacks may detected. We first investigated whether there are signs that suggest the attacks that take place in any systematic manner (eg mathematical correlation between the time of the packets repeated in several systems) . Next we generated a proof of concept to the system and make it suggest rules in order to prevent these attacks.

We will establish a company's network with some hids that report in two levels: 1. Silent alert - that will reach only us (to our algorithm) 2. noisy warning - will also report the real siem server of the company. We describe our suggested algorithem in the attached github page.

we made this proccess during our graduation and we focused on three sections:

- researching identifying features of the attacks
- developing a system against these attacks which:
  1. able to detect and alert when this kind of attack is committed.
  2. able to suggest new rules (for systems' manager consideration) in order to prevent future attacks of this type.
- developing a "virus" able to survive without commands from a remote attacker

Reserved for Publication Footnotes

[1] these numbers on edges are times

search and characterization foundation

The theoretical research was done in 3 steps:

1. We reviewed various of professional literature about central attack sources discovered in the last years, especially:
   - confincker
   - Flame
   - Gauss
   - Stuxnet
   - Red October
   - CozyDuke

   those sources were an apt (Advanced Persistent Threat is a set of stealthy and continuous computer hacking processes).

2. Consulting with various professionals in the cyber-security world - we talked with the companies Kandiro, Illusive networks, NSO, Symmetry and Ministry of Defense. Those sources approved the existence of fixed mechanisms in attack systems and examples were given.

3. reviewing viruses in the network- inspecting some resources of trojan horses given in the network. Mechanisms were detected too.

During our research, we found the evidences of mechanisms in attack systems including:

1. mechanisms time based:
   - monotonic sending every fixed time (or fixed interval) to unknown addresses.
   - sending fixed payloads to different places in the network every fixed interval of time.
2. mechanisms behavior-based:
   - Exiting the network with unsigned ssl certification from the same port and process.
   - exiting from fake user agent given inappropriate process.
   - reading data from the network's server using addresses from abroad.
   - monotonic validation procedure from unknown server.

Those actions are not suspicious when happening on a single computer in contrast to the same procedure from different computers in the network as it might be malicious program.

Material and Methods

virus. The "virus" is simply a code snippet written in python language we made in order to perform a proof of concept to our research and in order to proof the existence of a system able to identify the activity of an instance based on fixed deterministic mechanisms (like our virus).

The idea of the virus is that it propagate to other computers in fixed interval of time (as mentioned, in practice, we focused in time intervals only. We will later discuss on the other possibilities when applying other features). The virus is split into 2 parts - a 'client' and a 'server'. The 'client' is the victim and the 'server' is in the attacker. We decided that there is no need to wrap the virus with a shell-code since it is not essential for the proof of concept - we tried to find the communication it create, not its installation.

The client (victim) side:

1. The client create the socket.
2. The size of a given file is sent to the attacker alongside with the file itself.

The server (attacker) side:

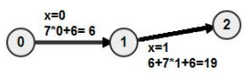
1. The server listen until it accepts the socket from the client.
2. The server accepts the size of the file and the file itself.

environment . In this part we are getting more practical, as our research shows that there is a mechanism which based on timing for some malicious attacks over our network. We focused of topologies with an hids on each computer. For simulating a flexible computers network, we used the environment services by CREATE. CREATELAB - Cyber Research, Experimentation And Test Environment is a cyber technologie project under the IUCC (International University Computation Center), which operated on Israel National Research and Education Network. This environment enables us to conduct advance research in cybersecurity through use of DeterLabs. This service provided us an isolate network with a full production virtual environment for any network topology or virtual machine operation system, setting up proper images for our experiment, and a fast recovery operation for the virtual environment. Our base topology was five NAT-connected computers with an ubuntu server operation system. One of them will act as an admin, and will run an IDS server side service, the three others will be armed with the client side, and three different antivirus. The last one will be the malicious attack getaway (this can be done also with an outside internet connection). The chosen IDS for the mission was OSSEC which is a free ,open-source, host-based intrusion detection system (HIDS). Thanks to its popularity and flexibility of this HIDS, and the fact it is well documented and open-source gave us a good system to experiment with, and a chance to explore prevention techniques. As followed, the client's contained a OSSEC client side and three different antivirus that chosen for our ubuntu operating system - CalmAV, FPROT and Comodo. All of them got good reviews and free for first use. Above all this protection wall, we imported our FTP Virus for any of the clients, with default arguments settings, and the listening process will run on the last computer in the network.

Experiment: After setting up the environment and making all installations and adjustments we could start testing. First, a naive approach with default rules in the OSSEC server side, and running latest versions of all antiviruses, the FTP Virus worked with no harm, and the information was liking to the attack getaway smooth. Out second step was trying to set the argument of the FTP Virus to be more aggressive, but still no notification from the HIDS or the antiviruses. For the prevention part, with modified the arguments setting of the FTP Virus for some fixed time. Next, adding proper roles for our HIDS to determinate this malicious attack. The results were interesting as the OSSEC HIDS did recognized a dangerous behavior over the network and notify about it.

database. We used Elastic Search as storage for the big amount of packets received from our environment. Elastic search is a no-sql datastore used for big data storage and analysis. It is a part of 3 parted - a data collector, a log-parsing engine, namely logstash and a visualizer of the data. We used the first two in order to keep a collection of the packets and parse the packets into a log passed to our system. The logs are based on json form for effective and simple analysis.

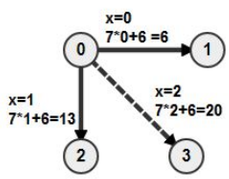
predictor. Predicting future attacks based on previous attacks can be really difficult given the big size of data you have to face with. As before, we focused on fixed time interval attacks only while we keep the rest for future researches and theoretical disscussion. What are fixed time interval attacks? Say that computer A gets infected first at time x, and it want to infect computers B and C based on the infection function 6x+7. Then for example if "patient zero" was infected at 6 pm , then it will infect computers B at 6:07 pm (x=0) and computer C at 6:20 pm (x=1). Here, we faced two models of propagation, the first one was described in the beginning of the paragraph and the other is theoretical and stayed without any answer for now. (see figure 1)



[1]

Fig. 1. first model-different propagation function to every node

The second model (see figure 2) determines that the propagation function is global and does not initialized on every computer infected.



[1]

Fig. 2. second model-there is one global propagation function

At the predictor, we got a sorted series of points, namely the times of the packets which identified as related and may be a part of an attack. We looked for a way to predict the next point in the series and decided to use newton's interpolation formula. interpolation is a method of constructing new data points within the range of a discrete set of known data points.

Definition 1. *Newton's interpolation function is a theory claiming that for any given finite set of data points, there is only one polynomial, of least possible degree, that passes through all of them.given a set of k+1 data points*

$$(x_0, y_0), (x_1, y_1), (x_2, y_2), ..., (x_k, y_k) \quad [1]$$

we mark divided difference of

$$x_i, x_{i+1} as f[x_i, x_{i+1}] = \frac{f(i+1) - f(i)}{x_{i+1} - x_i} \quad [2]$$

and

$$f[x_0, x_{i+1}, ..., x_j] = \frac{f[1, 2, ..., i, ..., j] - f[0, ..., i..., j-1]}{x_j - x_0} \quad [3]$$

where

$$j > i$$

. Then newton's formula is

$$f(x) = f(x_0) + (x - x_0)f[x_0, x_1] + ... + (x - x_0)(x - x_1)... \quad [4]$$

$$(x - x_k)f[x_0, ..., x_k] \quad [5]$$

The familiar use of this formula is to get a set of points on the same function looks like (x,y) and given some x point, return the y value of this x value on the function. We modeled it for our own use and given a set of times

$$x_0, x_1, x_2, x_3, x_4 \quad [6]$$

representing a sorted sent times of a group of packets suspected as related we predicted the next time a packet will be sent. We took every consequent pair of times and entered it by order to the formula and looked for the y value of the last time in the set. We found it difficult to relate and examine viruses which propagate to more than one station from single node. It creates different paths and harden the work of finding a single path for interpolation analysis. We tried to find the longest paths in the sending scheme but it is currently considered as an npc problem that has no polynomial solution for now (see figure 3). Pay attention that there might me repetitions to already infected stations and re-sending of the same packet to the same station.

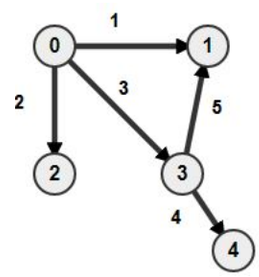


Fig. 3. We have to get one path for our interpolation analysis, but what is the right path? for example we might get here the series 1,2,3,4,5. what is the right path?

[1]

When implementing the predictor, we calculate the newton's interpolation with two arrays as we saw some fixed behavior in the formula.

An example of a rule created is that a packet from the kind .... with size .... is suspicious between the time 13:31:99 to 13:32:08 and should be dropped so the continuous propagating will stop.

Results

```
('point after 1270 is ', 2550.0, ' in ', '30+2*(X-10)')
('f(x)=', '30+2*(X-10)', ' from ', [10, 30, 70])
```

Fig. 4. function found after getting 3 points representing time of packets and prediction for the future

Our results proved that a mechanisms based viruses can not be identified by the security devices exist today as there was not any alert against our virus. We created an attack of this kind. We also created a proof of concept of a system able to recognize this virus given data from tools already attacked

[1] these numbers on edges are times

by the virus. We will discuss more about the difficulties we faced in the limitation and future work part. We think that the virus was not found by known security devices and softwares because it acts like known processes and software like google passing data about the user for improving its service or automatic updates of a program.

future work

limitations. We faced some difficulties while making our research. First, We categorized behavior patterns of fixed-mechanisms programs. Those features are based on past attacks and viruses, although we considered some features in our research which were not found in any virus we examined but might appear in future viruses. There's probably a possibility that some attacks will not be found thanks to new features the attacker want to depend on. Second, our system try to find some fixed features we defined. The attacker, given the list of features, will be able to trick the system with different tricks.

For example, we used newton's formula for interpolation in order to predict future times of the attack (based on repetitive time intervals), but it might not be accurate. why? Because newton's formula need a certain amount of points, depend on the propagation's time function. let's take look on a virus with propagation function of

$$f(x) = x^5 + 2x^4 + 30x^3 + 23x^2 + 31x \qquad [7]$$

will need about 6 points(although that with less points, it will give a close estimation). The more points it gets, the more accuracy it get. We also used linear interpolation only, which is unable to predict exponential functions like

$$f(x) = 2^{5x+2}, g(X) = 12^x \qquad [8]$$

Another problem is the existence of random functions in computer science and especially in malware writing. Even though it is not complete random, it depends on factors from the outer world (noises, the computer's clock, etc) and demands thousands of points in order to predict the next time of attack. There are also countless random and pseudo-random generators and each one of them demands different actions in order to understand how they work and predict their future points.That is why we focused on times based on functions only.

Additional problem is the delays in the network's infrastructure (sending, passing and receiving packets) made by various of reasons and might disturb the time prediction of new attacks. This leads to the need of using a buffer in order to find the exact sending time of a packet, not to mention that the processing time of every computer (while processing a packet) will probably be different. That is why we used interval of times in prediction instead of finding the exact time of future attack (we round all the time into tens of seconds).

The last problem we faced is to analyze a huge amount of packets and find an attack employees certain features from our list, so we had to find for example time relations and same types packets while there are not necessarily a relation from this kind.

despite all these facts, we want to mention that this is only an initial system, only to show a proof of concept of an principle and of course it can be proved dozen times and even use a machine learning in order to speed up the system's performance.

As a chance to improve our system's performance, We suggest to change the format of our function, for example from

$$f(x) = 2+3x*(x-3)+5x(x-3)(x-5)+43x*(x-3)(x-5)(x-6) \qquad [9]$$

to

$$f(x) = 2+((x-3)(3x+(x-5)(5x+(x-5)(43x(x-6))))) \qquad [10]$$

as it improves the complexity of calculating from factorial to polynomial-We did not faced with that problem because we only tried to create a proof of concept of a working system able to identified this kind of viruses.

ACKNOWLEDGMENTS. This work was supported by Ariel university. special thanks to Yehuda Grestel who guided us in the project. We also thank to all the professionals who helped us researching and categorizing the self propagating programs. Additional help is to Snir Levy and Adiel Am Shalom who helped us planning protection against multi-propagating programs.

1. IUCC - https://www.iucc.ac.il/ , (2017).
2. CREATELAB - https://createlab.iucc.ac.il/ .
3. DETERLAB - https://www.deterlab.net/ .
4. CLAMAV - https://www.clamav.net/.
5. COMODO - https://www.comodo.com/.
6. FPROT - http://www.f-prot.com/ .
7. https://en.wikipedia.org/wiki/Interpolation.
8. http://www.christof-strauch.de/nosqldbs.pdf.
9. https://www.cs.cmu.edu/ dga/papers/cuckoo-conext2014.pdf.
10. for more information about our project:https://github.com/maladi17/Idsiter.
11. https://www.icann.org/en/system/files/files/conficker-summary- review-07may10-en.pdf.
12. https://securelist.com/red-october- diplomatic-cyber- attacks-investigation/36740/.
13. https://securelist.com/stuxnet-zero- victims/67483/.
14. https://securelist.com/gauss-abnormal- distribution/36620/.
15. https://securelist.com/full-analysis- of-flames- command-control- servers-27/34216/.
16. https://securelist.com/full-analysis- of-flames- command-control- servers-27/34216/.
17. https://securelist.com/the-roof- is-on- fire-tackling- flames-cc- servers-6/33033/.
18. https://securelist.com/the-cozyduke- apt/69731/.
19. https://kasperskycontenthub.com/wp-content/uploads/sites/43/vlpdfs/kaspersky-lab-gauss.pdf.