A visual analytic approach for analyzing SSH honeypots

Jop van der Lelie - jop.vanderlelie@os3.nl
Rory Breuk - rory.breuk@os3.nl
University of Amsterdam
System and Network Engineering (MSc)
(Dated: July 8, 2012)

An SSH honeypot can be used to study the activities of an attacker by logging the full SSH session. In this paper we present an interactive visualization system that can be used by network security experts to visually analyze large sets of SSH honeypot data. By using different visualizations and interaction techniques the expert can explore SSH sessions and quickly find related sessions which will help in identifying attackers.

I. INTRODUCTION

Unauthorized access to servers occurs on a daily basis. Attackers are constantly searching the internet for servers that they can use for their malicious activities. An easy target for an attacker is a server on which an administrator has set up a service which allows users to control the server remotely but have a weak password set. This may be done unintentionally when a local application specific user account is configured to be used for remote login, like database accounts. An attacker aiming to find such a server will scan network ranges, probe the service specific TCP port and try to connect to the service. Whenever a service is found, the attacker will try to authenticate to the service by guessing user credentials. This process is often automated by a program called a scanner, which can do a dictionary attack or brute force attack on the service. After a successful login attempt, the attacker can use its credentials to login to the server and install his malware.

It is the task of a network security analyst to analyze these attacks and identify the attacker(s). Multiple attacks might be related and lead back to one attacker as looking at the characteristics of the attacks may show. A strong relation between attacks could indicate the same attacker.

To study the activities of attackers on a compromised server, one can set up a honeypot server[1][2]. A honeypot could, for instance, emulate a remote login service and will log all interactions with an attacker. Honeypots are often deployed within particular IP ranges to attract attackers looking for vulnerable targets in specific networks. Distributing honeypots over multiple networks is likely to increase the number of attackers connecting to the honeypot.

analyzing the data gathered from the honeypots can be a real challenge. As the number of attacks grows very large over time, it becomes impossible to manually compare all the sessions. Interpreting these in an automated way is very difficult, as behavior of different attackers varies substantially and is often unpredictable. This is where the aspects of *visual analytics*[3] can help the network security analyst in gaining insight in the data. An interactive visualization system can support the decision maker by automating parts of the data analysis, representing the data in a clear way and provide the expert ways of interacting with the data.

The current methods that are available for supporting the network security analyst in the decision making process do not fully use the capabilities of visual analytics. In this research we will present an interactive visualization system to perform data analysis on SSH sessions logged on a honeypot and to help the expert identify attacks that need to be investigated further.

II. RELATED WORK

A lot of research is done into visualizing attacks on computer networks, but most focus on visualizing attacks that are logged using an Intrusion Detection System (IDS) with a combination of NetFlow data. For instance, the tool *NFlowVis* that was presented by Fischer et al. [4] in 2008. *NFlowVis* can be used to visually analyze attacks in large-scale networks. The tool visualizes the NetFlows from the attacks from an IDS log. Their example provides a visualization is given of all SSH attacks on their network, but only connections to the SSH server are shown. It would be more interesting to see what an attacker does once an SSH session with the server has started.

Another tool to visualize NetFlow data is VIAssist, developed in 2009 by Goodall et al. [5]. This tool visualizes NetFlow data on a large dashboard and lets the expert easily browse the data using multiple views. It can provide details-on-demand when a specific flow needs to be investigated further. The tool integrates with the SiLK network flow analysis tools [6] to allow multiple analysis tools for input. However, since VIAssist only visualizes NetFlow data, it is not applicable for the analysis of attacks on SSH honeypots as we want to find possible relations between attacks by looking at the contents of each attack.

A visualization more related to SSH honeypots is the one of Jaime Blasco [7]. He created a visualization on data gathered with a Nephentes honeypot[8]. This honeypot is used to gather malware samples by emulating known Windows vulnerabilities which an attacker will exploit and sent his malware. The visualization that he proposes maps the source IP address of the attacker to a country. By doing so it is clear which country spreads a specific piece of malware. However, geographical information could be misleading when it comes to source of the attack because attackers often use proxy servers to hide their actual location. A visualization will only map the proxy servers instead of the real source of the attack.

When multiple honeypots are linked together managing all logs becomes a real challenge. In 2011 Visoottiviseth et al. [9] propose an architecture to store the logs from multiple honeypots in a centralized manner. In their experiment they created a geographical visualization of all attacks on their honeypots, but no further analysis of the attacks is done.

Another tool that is capable of managing multiple honeypots is SURFcert IDS[10], an open-source Distributed Intrusion Detection System based on passive sensors. The logs can be analyzed using a web-interface. However, only basic visualizations are used and no interaction with the data is possible.

Kippo-graph [11] is an application for creating basic visualizations from data gathered with SSH honeypot Kippo. Although the application creates numerous visualizations on the data, no interaction is offered.

In our research we will use SURFcert IDS honeypot data gathered with multiple sensors. We will create an interactive visualization system that assist the expert in his decision making process. The visualization will not focus on the NetFlow information but on the commands executed in the SSH sessions on the honeypot.

III. THEORY

In this section, we first explain the theory behind SSH honeypots. We introduce a number of concepts that we used for designing different ways of visualizing SSH honeypot data, which we will explain in the end of this chapter.

A. Secure Shell Honeypots

A common way to provide remote shell access on a *nix operating system is using the Secure Shell (SSH) network protocol [12]. By using the SSH protocol one can perform a secure remote login over an insecure network to access the remote shell. In order to login to the remote shell the user has to successfully authenticate itself to the SSH server. Public key authentication and password authentication are by far the most widely used authentication methods. With public key authentication, the user sends a signature encrypted with the user's private key and the server verifies this signature with the user's public key [13]. This requires the public key to be stored in the SSH server's allowed list.

An easier, but considered less secure method is a user name and password combination to authenticate the user. There are multiple ways of intercepting or guessing user credentials, of which brute force attacks are often used in non-targeted attacks. As long as there are system administrators who fail at securing their systems using a secure password there will be attackers trying to guess the credentials and compromise the system.

In order to study the activities performed by attackers after they compromised a system an SSH server is set up waiting for attackers to break in. But giving attackers a full system to (mis)use, is a potential security risk and therefore it is better to use a honeypot system.

A honeypot is an information system resource whose value lies in unauthorized or illicit use of that resource [14]

A honeypot system is exclusively set up for investigation purposes. It does not have any value other than providing services for attackers to interact with. In theory all traffic to the honeypot is considered malicious or unauthorized, it does not have any legitimate activity.

There are many types of honeypots, but they are usually categorized based on the amount of interaction that is offered; low-interaction and high-interaction honeypots [1]. A low-interaction honeypot will only emulate a service or operating system and thus offers a limited amount of interaction. The attacker can only perform the commands that are implemented in the emulation. A high-interaction honeypot emulates nothing, the honeypot is running a full operating system and real services and thus offers the attacker a high amount of interaction.

In order to set up an SSH service on a honeypot we need to emulate the whole SSH session and implement a response for all commands that a user is allowed to execute. Basic commands such as 'whoami' and 'w' are easy to implement, but most often a user wants to interact with the file system by using 'mkdir' and 'ls'. This could be all emulated, but it is easier to provide the honeypot with a virtual file system and copy some real executables and place them in the sandbox. It emulates all commands and gives the attacker full control of the virtual file system. Because this type of honeypot is not a full operating system, but does provide a fairly large amount of interaction to an attacker, it is considered a medium-interaction honeypot. The level of interaction hopefully convinces attackers in the honeypot being a real system. The commands executed in a session can be stored and if the attacker tries to download malware, it is possible to copy these files outside the virtual file system for later analysis.

The data gathered from the SSH sessions on the honeypot consists of the source and destination IP address of the attack and the commands issued in the session. Note that no NetFlow information is needed, because information such as the amount of bytes send is not relevant for this analysis. To investigate the attacks we focus on the interaction between the attacker and the SSH shell.

once a session is started, the source IP address is logged along with a time stamp and all commands send in that session. This allows the honeypot administrator to fully reconstruct every SSH session logged by the honeypot.

B. Visual Analytics

The goal of visual analytics might be described as:

to gain insight into homogeneous, contradictory and incomplete data through the combination of automatic analysis methods with human background knowledge and intuition. [3]

The goal of deploying SSH honeypots is to gain broad insights into threats. This can be in many fields, for instance active attackers, groups and the use of malware. The large amount of data makes it almost impossible to hand search the data. Especially when SSH sessions are compared on different levels like content and metadata.

By using a combination of visualization and data mining techniques we create the possibility to explore large amounts of SSH honeypot data in an interactive way. As a starting point we have to distinguish the sessions on different characteristics. This can, for instance, be done on the length of the session and by looking at specific commands in the sessions such as the number of downloads.

Once an interesting session is found, looking for similar sessions is important to identify attackers or groups. This can be done by simply looking at the source IP address of the attack, but since many attackers use proxy servers to hide their identity it is better to look inside the session itself. By linking sessions that issue the same download we can identify attackers regardless of the IP address used. Other commands that can identify an attacker are user creation commands or user modification commands. Changing the password to a unique password that is used in different sessions might indicate the same attacker.

In 2010 Bertini and Lalenne [15] presented an overview of the different research done in the field of visual analytics. They distinguish three categories: Computationally Enhanced Visualization (V++), Visually Enhanced Mining (M++) and Integrated Visualization and Mining (VM). Sessions gathered by SSH honeypots are often unpredictable, particulary because they are often performed by human attackers. The best way to analyze large amounts of SSH honeypot data, is to aid an expert in investigating the data using a combination visualization and computation methods. Therefore, visualization of SSH honeypot data would be categorized under a Computationally Enhanced Visualization (V++). The focus of the visualization will be manual exploration of the dataset and application of different filters to limit the search space for identification of anomalies.

To assist the expert in the visual analytics process, we can partly automate the data analysis process and present the results using an interactive visualization. Since 2001 research has been done into the field of interactive machine learning [16] and interactive data mining [17]. They describe an approach in which the expert can influence the partly automated data mining algorithm by interacting with a representation of the data. Later, Keim et al. [3] describe a more matured view on visual analytics by combining human and machine capabilities in a visualization to tackle data analysis. They present the visual analytics feedback loop where both the machine and the expert generate new hypotheses that the expert can verify using the visualization. This is a continuous process, as the expert will gain new insight and new hypotheses by creating different views of the data. This new analysis method has been summarized in the visual analytics mantra:

Analyse first
Show the important
Zoom, filter and analyse further
Details on demand

This mantra can loosely be applied to visual analytics on SSH honeypot data. However, we will take a new approach as we will first zoom in on our dataset to repeatedly find a session and then zoom out again to link it to other attacks. The approach we take involves the following steps:

Analyze first
Show the important
Details on demand
Zoom, filter and analyze further

1. *Analyze first*

As a first step in the visual analytics process an automated analysis of the data has to be performed. Because analysis by the expert will be the predominant focus of the visualization of SSH honeypots, this will be a basic step. The analysis starts by selecting a subset of the data by using a timeline. On the timeline two points are choosen and only attacks between this timespan are selected. The start and end of the timeline corresponds with first and last element from the dataset. Setting a timespan over the whole timeline would give the expert an overview of the entire dataset.

To help the expert in selecting the right timespan line charts should be shown on the timeline. Each line represents a different value in the dataset which can be shown over time. For instance the amount of attacks or the total commands used can help the expert by setting the right timespan. An automated analysis on the timeline could help the expert, by automatically setting the timespan based on the values in the linechart.

After selecting a subset of the dataset using the timeline, another visualization should be used to filter out attackers that have a small amount of commands executed on the honeypots. This can be done by creating a linechart that displays the amount of IP addresses that have a certain amount of commands send. The expert can then define a minimum for the total amount of commands send by an attacker. The linechart can help the expert by giving him an overview of the subset and will give an indication of the size of the subset that will be left after selecting a minimum.

2. *Show the important*

In this step of the mantra the expert should select a session from the remaining attacks for further analysis. In only two steps we drill down to one attack and then zoom out to relate this session to the rest of the dataset. Details about the remaining attacks should be shown to help the expert select one. The following details should be shown:

- The source of the attack.
- The target of the attack.
- The time of the attack.
- The length of the session.
- The total amount of commands executed by the attacker.

The attacks could be displayed in a list with smart sorting capabilities, making the expert able to sort the list on the previous named items. When sorting the list after an attack was selected, the list should be automatically scrolled to the selected attack again. This allows the expert to relate the attacks close to the selected one on the resorted list.

3. *Details on demand*

By selecting an attack from the attack overview the expert will be presented more details about the attack. In this step we will look at the commands send in the attack rather than the information about the attack. Some automated analysis will be performed in order to extract information from important commands. We distinguish five data fields that can help identify an attacker in a SSH session:

- Host names from which files have been downloaded.
- Filenames of downloaded files.
- Name of executables that are run.
- Usernames inserted on user creation.
- Passwords inserted on user creation or password change.

After selecting the attack these five data fields should be automatically extracted from the attack. These fields could be visualized using a graph where the root node represents the attack and all children the characteristics of the attack.

If the expert wants to analyze the attack further, hovering over the attack should provide more details. Such as the length of the attack. Yi et al. [18] describe this as *Elaborate*. If even more details are required the full SSH session should be shown when clicking the attack. The detailed view is also connected to the timeline. When the detailed view for a session is shown the attack is plotted on the timeline. This enables the expert to see if the related attacks in the graph are also related by time.

4. *Zoom, filter and analyze further*

Once an interesting session has been found the expert should use this session as a starting point for further exploration and investigation of the data. From the selected session the child nodes are created based on the characteristics found in the session. Our goal here is to use these characteristics to find related other attacks. Based on these characteristics we can search the sessions of other attacks and connect them to these nodes. By clicking on one of the related attacks the process can be repeated and more related attacks can be found.

Because the data itself is not hierarchical, a node can be its own ancestor. In order to keep the visualization from cluttering only forward links should be allowed. Whenever a related attack is found that already exists higher in the graph it should be excluded from the analysis process and not be used to find more related attacks. This will prevent looping between the same attacks.

To find related attacks on for example the hostname used in a file download we can create a node connected to the root node representing the hostname. All other attacks with the same hostname present in one of the commands will be connected to this node. We now call this a relation node. Colours can be used to distinguish attack nodes from the relation nodes. The difference between the relation nodes can be seen by the color and the name on the node. The name represents the variable that is used to find related attacks, in this example the hosename.

Using the information gathered during the exploration of the data, the expert can decide to change the way in which the data is filtered in the first step. It should be possible to change the timespan and the required amount of commands executed by the attacker of the attacks shown in the second step. Also a word filter should be implemented to exclude or include attacks in the dataset that have a word in one of their commands. By adjusting the filters the dataset is changed and the list of attacks is refreshed. The visualization can now start over from the point where the expert has to choose an attack from the list of attacks.

IV. EXPERIMENTS

A. Dataset

For our experiments, we used a dataset that was gathered over a period of twenty months by the Dutch National Cyber Security Centre (NCSC-NL). This agency is the center for expertise on cyber security and incident response of the Dutch government. It is aimed at preventing ICT and internet related incidents and coordinates response to these incidents. They monitor multiple networks in The Netherlands by using honeypots. All the data is gathered in a central database using SURFcert IDS[10], an open-source Distributed Intrusion Detection System based on passive sensors. The tool is developed by SURFnet, a Dutch ISP that provides ICT services to research institutes in The Netherlands. Multiple honeypots can be set-up on a central server and can be reached via a sensor. Each sensors is placed in a network that needs to be monitored. The sensor will open a VPN tunnel to the central server where the honeypot is running and the logging takes place. Most low-interaction honeypots are supported such as Dionaea[19] and Amun[20], which can simulate multiple known Windows vulnerabilities for gatherng malware. Another honeypot that is supported is Kippo. Kippo[21] is an SSH honeypot server written in Python and inspired by Kojoney[22]. We will use the alerts generated by the Kippo honeypot in SURFcert IDS.

The dataset we used is a subset of the SURFcert IDS database. We filtered out all alerts generated by honeypots other than Kippo. This leaves us with a total of roughly 6,5 million attacks performed on 27 different sensors in a timespan of twenty months preceding our research. After filtering out all login attempts, we were left with a total of 6,273 SSH sessions consisting of a total of 56,607 commands. Because the location of the sensors is considered confidential, we obfuscated the IP addresses in the database by replacing them with a local network address in the 192.168 range.

The information available for each alert generated by SURFcert IDS is:

- The IP address and port number of the attacker.
- A timestamp of the attack.
- The sensor on which the attack took place.
- The commands that the attacker issued in the SSH session.

B. Implementation

To retrieve the information from the SURFcert IDS database, we used a PHP script to query the PostgreSQL database. These PHP scripts return JSON encoded query results. The visualization will be on an HTML page and Javascript will be used for interaction and data gathering. We will use a visualization framework written for Javascript, D3. This framework helps creating visualisations in HTML, SVG and CSS by manipulation of the DOM (Document Object Model) using a data-driven approach [23].

We combined all the concepts we introduced in section III B into a proof of concept of an interactive dashboard. Screenshots of the dashboard are shown in appendix A. The dashboard consists of six elements which are described further on. The numbers on the figures in appendix A correspond with the numbers of the element descriptions.

1. **Timeline** The timeline is the first filter step in the dashboard. Using the brush, an expert can choose the timespan for the shown attacks. Three line charts in which the amount of *sessions*, *attacks* and *commands* per week are plotted over time, can help an expert identify interesting events in order to choose a timespan.

2. **Commands per IP filter** A line chart shows the number of IP addresses plotted against the minimum amount of commands executed by each IP address. The expert can easily see how quickly the amount of shown attacks will change, when he changes the required amount of commands by using the brush on the chart.

3. **Attacks list** A table shows all the attacks left after applying the filter settings from *1*, *2* and *4*.

4. **Word filter** The expert can use the word filter in order to only show sessions in the *attacks list* containing words from the include list and without words from the exclude list.

5. **Attack graph** The graph as described in section III B 3 has been made into an interactive graph, allowing the expert to explore the data and finding relations between different sessions. Hovering on attack nodes shows the time stamp of the attack on the timeline with a vertical line, hovering on the other nodes shows the timestamps of all attack nodes connected to that node.

6. **Session view** The session display is the highest level of detail the expert can get about a session. A full list of all the commands issued by the attacker is given.

C. Results

The resulting dashboard is a powerful tool, which assists a network security analyst in analyzing large amount of SSH honeypot data. By using our visualization the expert can quickly apply filters, find an interesting session and explore the data by finding related attacks. If needed, the expert can request all commands issued during a session by using the session view of an attack. This enables the expert to fully analyze each attack in detail. An example use-case scenario on the system is shown in appendix A.

In the example, the expert wants to look at new sessions from active attackers. He selects a timespan surrounding the last peak of attacks in the timeline (*1*) and decides to use the automatic filter settings of the "commands per IP" filter (*2*). The expert wants to study locations from which malware is downloaded, so he adds `wget` and `tar` to the include list in the word filter (*4*). He adds `microsoft` to the exclude list, so attacks in which files were downloaded from the Microsoft website are excluded. This leaves the expert with three attacks in the attacks list (*3*).

Going through the remaining attacks, the expert sees in the attack graph (*5*) that the attack is related to another attack by two nodes; the download host `psycopath.ucoz.ru` and the file `relax.tgz`. Looking into these sessions, he finds that in both cases `relax.tgz` has been downloaded from the host given in the host node. By hovering his mouse pointer over the host node, he sees on the timeline that the timestamps of the attack nodes are about a month apart, meaning that the host might still be providing the file `relax.tgz`. The expert decides to mark the host as a source of malware, which can be used for further investigation.

In the attack graph, the expert also discovers that the attacker downloaded a file called `narcoman.tgz`. After expanding the attack connected with this file, the expert sees that this same file was hosted at `best-hack.clanteam.com` and that this host was used in 28 sessions in our dataset, distributed over more than a year. This host might be a very actively used malware provider, and could lead the expert to create an incident report and start an investigation into this host.

V. CONCLUSION

We developed a visualization system that can be used by experts to visually analyze attacks on Secure Shell (SSH) honeypots. Since the whole session is logged by the honeypot we can perform an in-depth analysis of the attack and study the attackers activity. Our approach is to first drill down to one interesting SSH session and from there explore the whole dataset.

Using our visualization system enables the expert to perform relevant filtering on the dataset and then select an attack for further investigation. From the SSH session associated with this attack multiple characteristics can be extracted that may help in identifying the attacker. Next, related sessions are identified by looking for sessions that have the same characteristics. Each session is plotted on a graph and is connected with their relation nodes representing the characteristic they have in common. By making the graph interactive the expert can explore the dataset by clicking on a session. Details-on-demand are provided by showing the characteristics found or the complete session. This way the complete dataset can be explored by the expert by following the relations between attacks starting by the selected attack. In our opinion, this visualization system provides the expert with a powerful tool in analyzing attacks on SSH honeypots.

VI. DISCUSSION AND FUTURE WORK

In our study we focused merely on the SSH honeypot data. However, the visualization techniques we developed could also be applied to other honeypots. We think that a visual analytic approach can greatly help an expert in his decision making process. A visualization system for malware analyzes would be a nice addition to the field of visual analytics.

The visualization that we presented is implemented in HTML and JavaScript. It would be nice to add this visualization system to existing honeypot systems such as SURFcert IDS and Kippo.

Another valuable addition to Kippo would be the use of a logon script that accepts any password after a random amount of login attempts. This would provide the possibility to link a brute force attack back to an SSH session that is most likely to be opened from another IP address. This is currently not possible with a list of credentials as these are too generic.

VII. ACKNOWLEDGEMENTS

We would like to thank the Dutch National Cyber Security Centre for giving us the opportunity to conduct this research and making their data available for us. Especially Bart Roos and Tarik El-Yassem for supervising the project. And Marcel Worring from the University of Amsterdam for giving us valuable feedback on our visualization and report.

[1] N. Provos. A virtual honeypot framework. In *Proceedings of the 13th USENIX security symposium*, volume 132, 2004.

[2] N. Provos and T. Holz. *Virtual honeypots: from botnet tracking to intrusion detection.* Addison-Wesley Professional, 2007.

[3] D. Keim, F. Mansmann, J. Schneidewind, J. Thomas, and H. Ziegler. Visual analytics: Scope and challenges. *Visual Data Mining*, pages 76–90, 2008.

[4] F. Fischer, F. Mansmann, D.A. Keim, S. Pietzko, and

M. Waldvogel. Large-scale network monitoring for visual analysis of attacks. *Lecture Notes in Computer Science*, 5210:111, 2008.

[5] J.R. Goodall and M. Sowul. Viassist: Visual analytics for cyber defense. In *Technologies for Homeland Security, 2009. HST'09. IEEE Conference on*, pages 143–150. IEEE, 2009.

[6] Silk network flow analysis tools. `http://tools.netsa.cert.org/silk/`.

[7] J. Blasco. An approach to malware collection log visualization. *Journal*, 2005.

[8] Nepenthes honeypot. `http://nepenthes.carnivore.it/`.

[9] V. Visoottiviseth, U. Jaralrungroj, E. Phoomrungraungsuk, and P. Kultanon. Distributed honeypot log management and visualization of attacker geographical distribution. In *Computer Science and Software Engineering (JCSSE), 2011 Eighth International Joint Conference on*, pages 23–28. IEEE, 2011.

[10] Surfcert intrusion detection system. `http://ids.surfnet.nl/`.

[11] Kippo-graph visualization. `http://bruteforce.gr/kippo-graph`.

[12] Rfc#4251: The secure shell (ssh) protocol architecture. `http://www.ietf.org/rfc/rfc4251.txt`.

[13] T. Ylonen and C. Lonvick. The secure shell (ssh) authentication protocol. 2006.

[14] L. Spitzner. Honeypots: definitions and value of honeypots, 2003.

[15] E. Bertini and D. Lalanne. Investigating and reflecting on the integration of automatic data analysis and visualization in knowledge discovery. *ACM SIGKDD Explorations Newsletter*, 11(2):9–18, 2010.

[16] J.A. Fails and D.R. Olsen Jr. Interactive machine learning. In *Proceedings of the 8th international conference on Intelligent user interfaces*, pages 39–45. ACM, 2003.

[17] M. Ware, E. Frank, G. Holmes, M. Hall, and I.H. Witten. Interactive machine learning: letting users build classifiers. *International Journal of Human-Computer Studies*, 55(3):281–292, 2001.

[18] J.S. Yi, Y. ah Kang, J.T. Stasko, and J.A. Jacko. Toward a deeper understanding of the role of interaction in information visualization. *Visualization and Computer Graphics, IEEE Transactions on*, 13(6):1224–1231, 2007.

[19] Dionaea honeypot. `http://dionaea.carnivore.it/`.

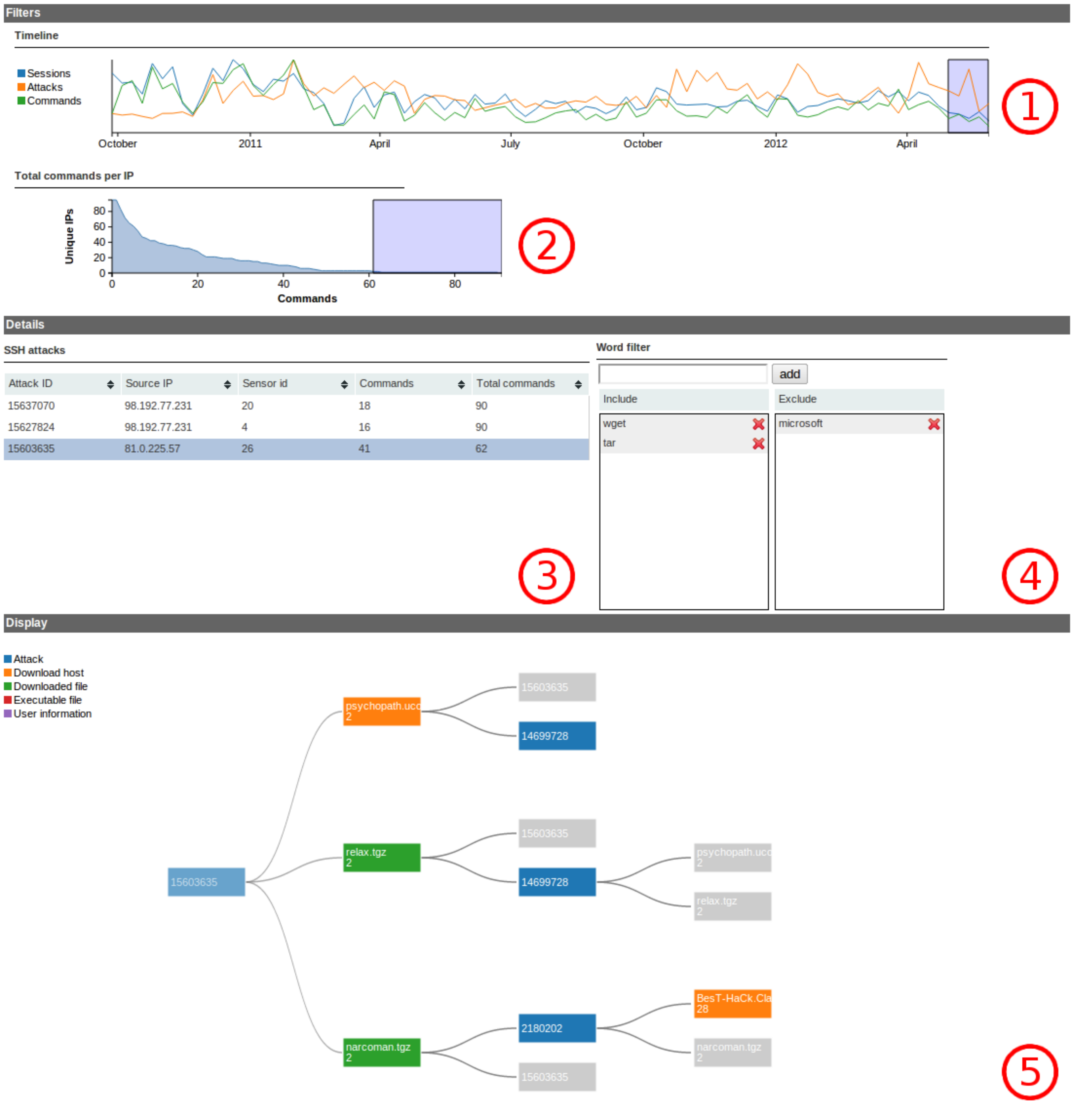
[20] J. Göbel. *Amun: A python honeypot.* Universität Mannheim/Institut für Informatik, 2009.

[21] Kippo: A ssh honeypot. `http://code.google.com/p/kippo/`.

[22] Kojoney: A ssh honeypot. `http://kojoney.sourceforge.net/`.

[23] M. Bostock, V. Ogievetsky, and J. Heer. D^3 data-driven documents. *Visualization and Computer Graphics, IEEE Transactions on*, 17(12):2301–2309, 2011.

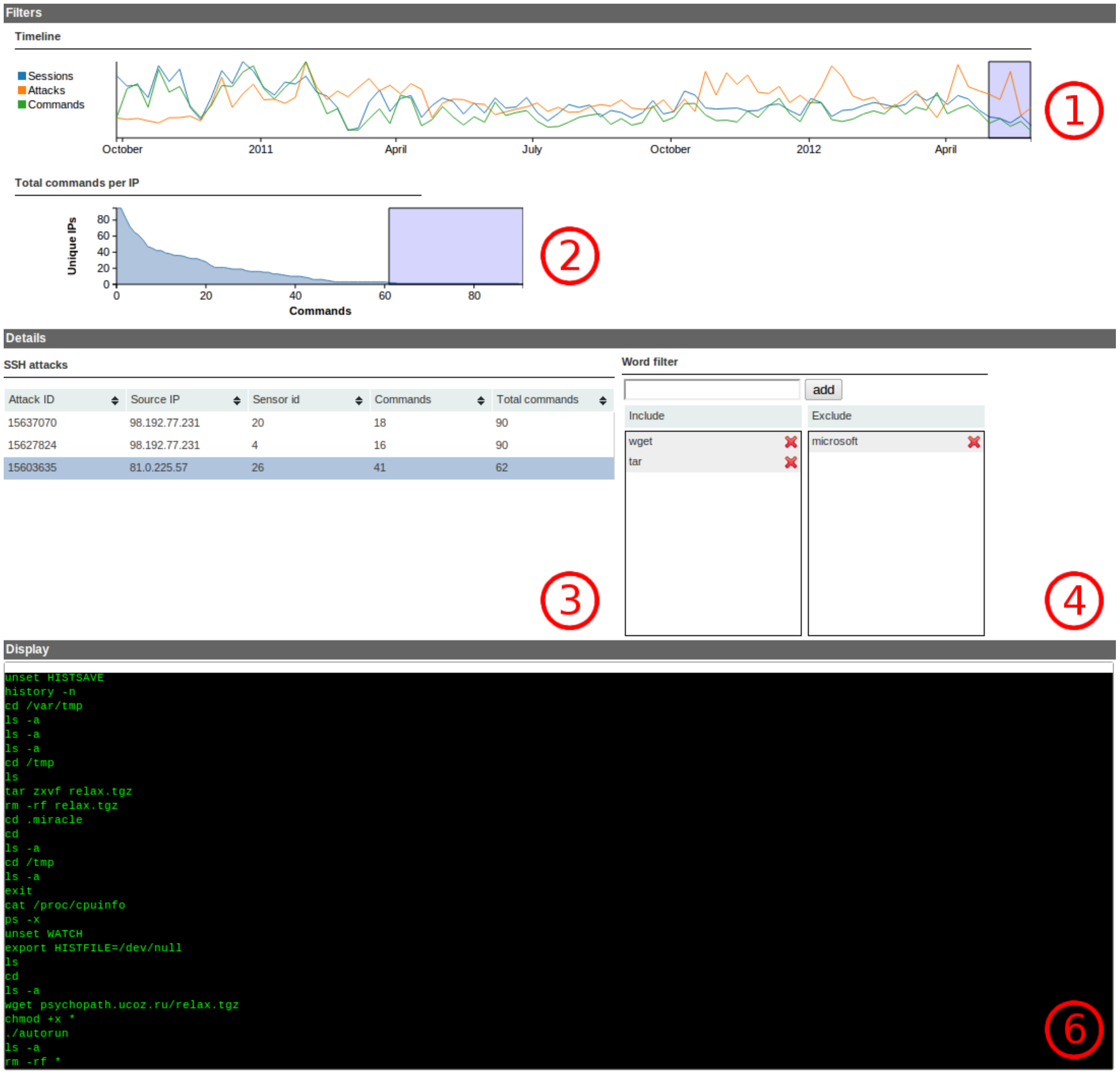
Appendix A: Full view of the dashboard



Attack ID	Source IP	Sensor id	Commands	Total commands
15637070	98.192.77.231	20	18	90
15627824	98.192.77.231	4	16	90
15603635	81.0.225.57	26	41	62

Include	Exclude
wget	microsoft
tar	

FIG. 1: Full view of the dashboard, while exploring the data.



SSH attacks

Attack ID	Source IP	Sensor id	Commands	Total commands
15637070	98.192.77.231	20	18	90
15627824	98.192.77.231	4	16	90
15603635	81.0.225.57	26	41	62

Word filter

add

Include	Exclude
wget	microsoft
tar	

```
unset HISTSAVE
history -n
cd /var/tmp
ls -a
ls -a
ls -a
cd /tmp
ls
tar zxvf relax.tgz
rm -rf relax.tgz
cd .miracle
cd
ls -a
cd /tmp
ls -a
exit
cat /proc/cpuinfo
ps -x
unset WATCH
export HISTFILE=/dev/null
ls
cd
ls -a
wget psychopath.ucoz.ru/relax.tgz
chmod +x *
./autorun
ls -a
rm -rf *
```

FIG. 2: Full view of the dashboard, while looking into a session.