Fast Network-Based Brute-Force Detection

Robert Koch, Gabi Dreo Rodosek
Universität der Bundeswehr
Institut für Technische Informatik
Neubiberg, Germany
[Robert.Koch, Gabi.Dreo]@UniBw.de

***Abstract*—Today, the Internet is a crucial business factor for most companies. Different traditional business divisions like distance selling or money transfers enhanced or even switched to the Internet, others emerged directly from it and a billion dollar business evolved over the past years. Therefore, the high fiscal values are alluring criminals. Attacks with the aid of the Internet can be executed from a safe distance, different (or even missing) IT laws in different countries are hampering the transboundary criminal execution. For example, brute-force attacks to gain access to systems and servers are still a popular and successful attack type. After gaining access, sensitive data can be copied, spyware can be installed, etc. Current protection mechanisms require extensive administration or can reduce network performance. Therefore, we propose a new architecture for network-based brute-force detection in encrypted environments. The system evaluates the similarity of the network packet payload-sizes of different connections. No information about the encryption in use or the functionality of the authorization process is required. Based on the high similarity of rejected connections, an identification of brute-force attacks is realized.**

***Keywords-brute force*; *intrusion detection*; *network-based*; *similarity*; *inherent knowledge*.**

I. INTRODUCTION

In 1994, the first websites for online banking and the online ordering of pizza were published. Since then, endless services have been developed and also traditional services like distance selling moved to the Internet. The collapse of the dotcom bubble in 2000 did not end this trend, but reduced the superelevated expected profits to more realistic levels again. For example, in Germany, the World Wide Web became the revenue driver with 53.3 percent of the entire mail order business in 2010.

The financial power and the possibility to conduct attacks over the Internet from long distances (therefore, hampering a criminal prosecution even in the case of a detection and localization of an attacker) attracts numerous criminals. By that, a complex and surprisingly specialized cyber crime market has been evolved over the past years. Today, professional attack tools are dealt in the digital underground which are able to perform effective attacks.

Brute-force attacks are a popular technique to gain system access [1]. They are easy to accomplish and will have a great impact, if the attacker breaks into an account. Therefore, attacks like *SSH* brute-force are still one of the most frequent attacks [2], [3]. For example, the statistics of ATLAS provide information about the most frequent attacks in the last 24 hours [4]. Repeatetly, brute-force attacks (*SSH brute-force login attempts*) had been in the Top 5 of the most frequent attacks per subnet. Numerous tools provide an easy and automated attack conduction with detailed configuration options (e.g., number of parallel connections, number of connection tries per second, ciphering, dictionary-based or systematic tries). `SSHatter` [5] and `brutessh` [6] are two examples of corresponding attack tools.

The popularity of this kind of attack is based on the bad usage of passwords, too. Often, passwords are very simple, used for multiple systems and logins or they are created based on simple formulation rules (e.g., the current month and a number) when systems force regular changes of passwords. Different studies (e.g., see [7], [8], [9]) demonstrate that this situation does not improve even with intense information and warnings to the users.

Today, numerous services offer remote access or they are used as substructure for further services, e.g., the safeguarding of remote desktop access by tunneling unencrypted or vulnerable protocols like *RDP* over secure connections, for example by the use of *SSH*. Also, numerous web services are based on an authentication by an username-password combination which can be attacked by brute-force dictionary attacks. Therefore, the user remains a crucial weakness and opens up important points of attack. Current systems for the detection and prevention of brute-force attacks often have to be installed and administrated for every system with corresponding services. Network-based solutions often reduce network performance, too.

We present a new network-based architecture for the detection of brute-force attacks. The system evaluates statistical data of the network-connections and calculates the similarity of authorization requests to detect attacks. Neither the location or the type of services nor any information about the encryption in use is needed: The system can be placed into the network and runs out-of-the-box, resource-saving and without any configuration.

The remainder of the paper is organized as followed: In Section II, the mode of operation of remote sessions is presented and detection possibilities of brute-force attacks are identified. The architecture and the new detection principle

 ISBN: 978-1-61208-199-1

Table I
PAYLOAD SIZES OF SERVER AND CLIENT PACKETS DURING AN AUTHORIZATION PROCESS. ON THE LEFT SIDE THE AUTHENTICATION FAILED, WHILE THE PACKET SERIES ON THE RIGHT WAS SUCCESSFUL: THE PROCESS FINISHED AND A SESSION WAS ESTABLISHED. n IS THE NUMBER OF THE OBSERVED PACKET, ZERO-SIZED PACKETS AND ADMINISTRATIVE NETWORK PACKETS (E.G., AN *Acknowledge*) ARE LEFT OUT FOR BETTER READABILITY.

n	Size	Source	n	Size	Source
5	39	S	5	39	S
7	39	C	7	39	C
9	792	C	9	792	C
10	784	S	10	784	S
13	24	C	13	24	C
14	152	S	15	152	S
16	144	C	17	144	C
17	720	S	18	720	S
18	16	C	19	16	C
20	48	C	21	48	C
22	48	S	23	48	S
23	64	C	24	64	C
25	64	S	26	64	S
27	144	C	28	144	C
29	64	S	30	32	S
31	144	C	32	128	C
33	64	S	34	48	S
35	144	C	35	448	C
37	64	S	37	112	S
			38	368	S
			40	80	S
			42	48	C
			43	176	S
			44	64	S
			47	32	C

is presented comprehensively in Section III, while the results of the prototypical implementation are given in Section IV. A brief overview of related work is given in Section V, while Section VI concludes the paper.

II. REMOTE SESSION BASICS

Todays remote session services are typically encrypted or transported over encrypted channels, for example *RDP* through an encrypted *SSH*-tunnel. Therefore, only limited information about the transported datagram can be evaluated and used for the detection of attacks, namely the packet sizes of the encrypted payload and the points in time when the different network packets of a session pass a specific observation point, e.g., a border gateway in the subnet. In particular, the often used content of the payload is not available. Therefore, a deep packet inspection (DPI) is not possible; a detection must be independent from the availability of the payload.

After the client initiated the connection to the server, the authentication phase is started, e.g., by the request of an username and password. After a successful authentication, the user will be able to proceed with the remote session, otherwise the login information will be requested again. Typically, the login information can be entered a limited number of times (e.g., three times), after that the server will close the session and disconnect the client. Therefore, an attacker can try to infiltrate the system by testing various username-password combinations, e.g., based on dictionaries.

The observable parameters of a *SSH* remote session are shown in Table I. In the example, the encryption algorithm `AES128-CBC` is used, therefore generating the characteristic payload sizes as shown.

As one can see, there are typical similar packet series in the case of the rejected authentication tries and different characteristic packet series in the case of a success. Therefore, these packet series can be used to provide a detection of brute-force attacks, based on the recurrent observability of packet series corresponding to rejected authentications. Anyway, because the exact sizes can differ keenly, e.g., based on the used encipher algorithm or used padding, a detection based on a comparison of the strict packet sizes would need numerous patterns in a database and one can not be sure to catch all possible cases.

III. A NEW CONCEPT FOR BRUTE-FORCE ATTACK DETECTION

To enable a network-based fast brute-force detection which is independent from the used remote session protocol as well as the used cryptographic algorithm, we present a new detection concept and corresponding detection system which overcomes the shortcomings of current systems.

The detection of attacks is realized by monitoring the encrypted datastreams and evaluating selected packet sequences by the calculation of their similarity: If the packet sequences of failed login attempts are examined by a similarity measure, e.g., a cross-correlation, repeated negative authentications will result in high similarity values. On the other side, if a connection is authenticated successfully, the packets sent for the authentication process and the subsequent network packets will be quite different, therefore ending in small similarity values. By that, the similarity between data streams of one or different connections can provide information about an ongoing attack. Two basic types for the comparison of data streams and calculating their similarity can occur, by ports (different connections of one or more IP addresses) and by lines (one or more authentication tries within one connection). Analyzing a remote session in more detail, multiple cases of parallel and serial data streams can occure (see Figure 1) and must be taken into consideration for the calculation, namely:

- Initiation of a connection and successful authentication: After the preamble, a successful authentication is fulfilled and a data connection is open up and used for data transfer.
- Initiation of a connection and repeated rejected authentication attempts. Termination of the connection after a maximum number of false tries, initiation of a new connection on a new port.
- Initiation of parallel connections with one or more authentication attempt(s), termination by the client after

 ISBN: 978-1-61208-199-1

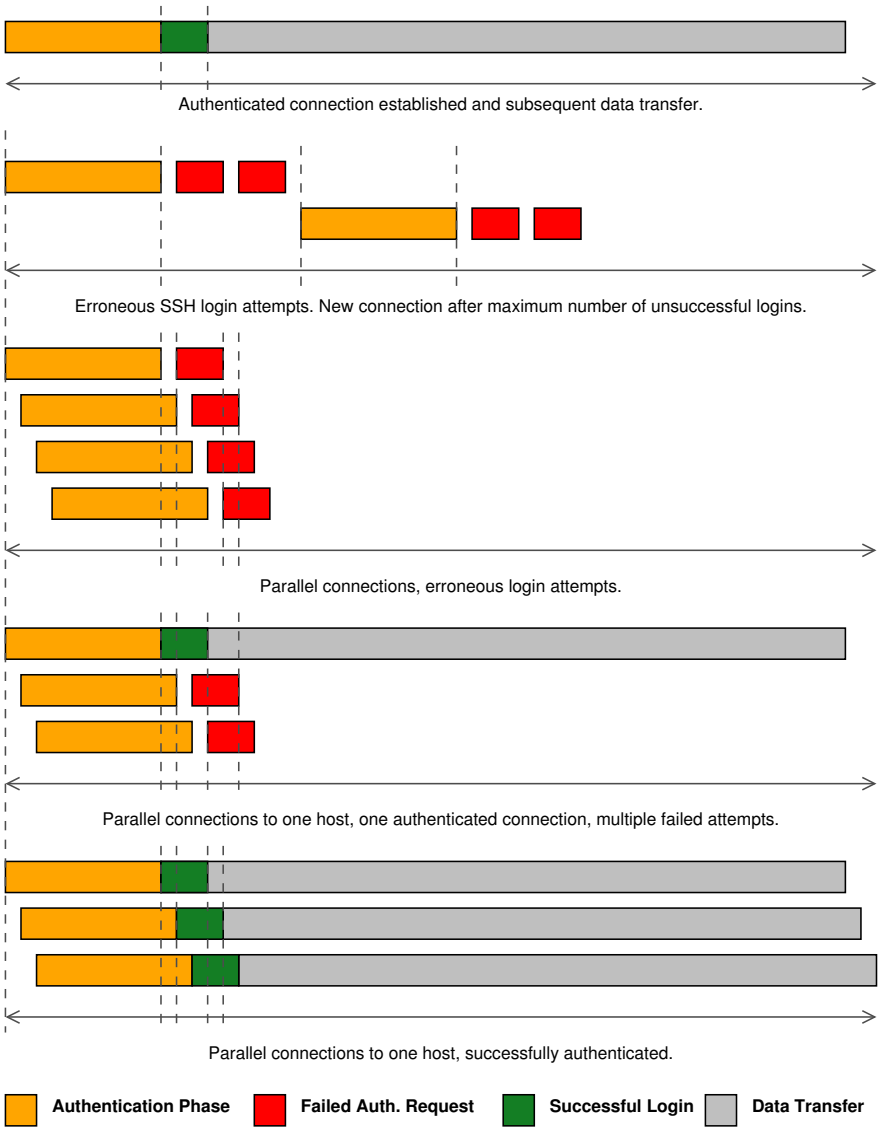

Figure 1. Different cases during the authorization phase of a remote session. After the preamble, the user has to be authenticated, e.g., by a username-password-combination (authentication phase). Based on that, the access is granted (successful login) and data can be transmitted (data transfer) or it is rejected (failed auth. request) and the user has to re-enter the information or the connection is closed.

an unsuccessful try.

- Initiation and successful authentication of a connection, concurrently rejected authentication attempts on parallel connections (e.g., when using NAT).
- Initiation and execution of parallel sessions with subsequent data transfers.

Further cases, e.g., a connection to a port with one unsuccessful authentication try and a subsequent successful one, can be subsumed to the similarity calculations of the given cases.

As indicated in Figure 1, only the sequence of network packets which represent a successful respectively failed authorization attempt is needed for the calculation. Anyway, it is not necessary to know the exact number and position of the involved network packets, because of the subjacent similarity measurement. Therefore, even if some packets are missing or are not belonging to the authentication itself, the similarity values will stay within an evaluable border. By using this approach neither the payload sizes nore the exact number of packets must be known.

The system is based on a transparent network bridge. All the network packets are copied by the use of the `pcap`-library and sent to the detection engine. First, hash-values are built for the identification and management of each connection. Based on the address and port and the payload sizes of the transmitted packets, the used type of connection

 ISBN: 978-1-61208-199-1

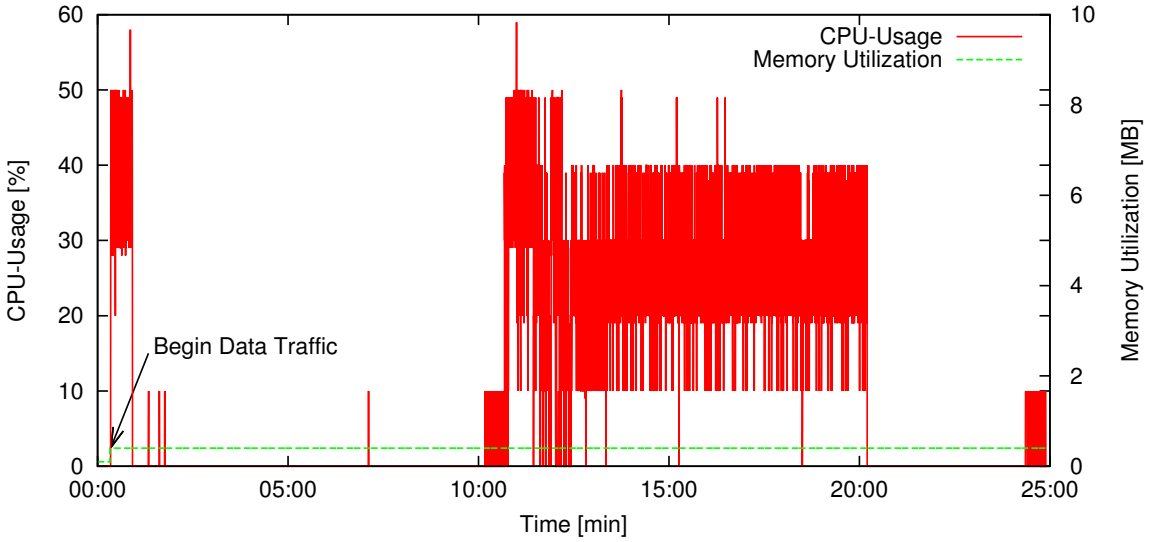

Figure 2. Resource usage of the detection system under load of synthetic traffic.

and correlation is identified, afterwards this information and the packet series are given to the correlation function which calculates their similarity. For the calculation of the similarity, the following formula is used:

$$s_d = \frac{\sum_{i=1}^{n} [(f[i] - m_f) \cdot (g[i-d] - m_g)]}{\sqrt{\sum_{i=1}^{n} (f[i] - m_f)^2} \sqrt{\sum_{i=1}^{n} (g[i-d] - m_g)^2}}$$

m_f and m_g are the arithmetical means of the value series $f[x]$ respectively $g[x]$, d is the considered shifting of $g[x]$. If the values of the two series are similar for a specific d, the standardized cross-correlation will result in a value of one. If the correlation produces zero for all shiftings of d, the series will be uncorrelated. Therefore, the algorithm uses the series of network packets transmitted during the authentication process, searching for areas of high similarity based on repeated rejected authentications. Again, note that it is not necessary to know the exact borders of packets belonging to the authentification process because of the used shifting.

If high similarities are detected a defined number of times, a blocking rule for the `netfilter` firewall is generated and installed on the bridge via a system call of `iptables`. Therefore, the IP address which is the originator of the attack is no langer able to reach the server.

IV. Proof of Concept

For the verification of the efficiency of our approach, a prototype was implemented and tested. The system is realized as a lightweight network-based detector integrated into a transparent bridge. Therefore, it can be placed everywhere into the network, typically close to the border router (therefore being able to analyze all incoming traffic).

Two test methodologies had been used: Firstly, a simulated environment was installed, where both the benign and the malicious traffic was synthetic. Therefore, the behavior and correctness of the system could be decided exactly. Secondly, evaluations in a productive network had been fulfilled, therefore being able to analyze the system under real-world conditions. There, the real traffic of the network presents the background traffic and attacks onto targets in the network had been conducted. By that, the detection system has to operate in the real-word environment and it is possible to evaluate wether all conducted attacks are recognized. Beyond that, it is possible that real attacks appear in the network during the evaluation phase. Because these attacks are not known, the system should detect them, but one can not say if all unknown attacks are detected correctly. To reduce the probability of a wrong evaluation, additional Intrusion Detection Systems were used to analyze the connections on the server. For the execution of attacks, `SSHatter` and `brutessh` had been used as well as manual connections initiated with `ssh`. Different encryption algorithms had been used to verify the robustness of the system. For example, the algorithms `AES128-CBC`, `BLOWFISH-CBC` or `AES128-CTR` can be used by the `Ciphers`-option of *SSH*:

```
ssh -o Ciphers=blowfish-cbc IP-Addr.
```

First, the resource usage during operation on loaded network links was analyzed. The detection system was installed on an Intel P4 computer (3 GHz, 498 MB main memory) which was used for the synthetic as well as the real-world evaluation on a 100 Mbps link.

Figure 2 shows the CPU-usage and memory utilization

 ISBN: 978-1-61208-199-1

over a period of 25 minutes of synthetic traffic.

Easy to see, the memory usage rises in the beginning to prepare the initial hashtables but remains below one MB which is constant during the evaluation period. During the evaluation multiple blocks of intense traffic were generated, resulting in higher CPU-load. As one can see, even with the outdated hardware the CPU-usage remains between 20 and 40 percent on average.

The result of the performance evaluation in the real-world environment is shown in Figure 3. As like as in the synthetic case, the CPU-usage remains on 44 percent in average when evaluating a fully-loaded 100 Mbps-link. Also the memory usage for the needed hashtables is quite low and remains on an average of 0.3 MB. Therefore, the system is efficient and able to monitor broadband links with minor hardware resources.

Next, the detection capabilities had been evaluated. An example of the evaluation in the synthetic scenario is shown in Figure 4.

The total number of connections at a time (representing the different remote sessions respectively their authorization phase) are differing based on the simulation an go up to approx. 50 sessions in parallel. Meanwhile, tool-based and manual attacks had been conducted. As soon as an attacking IP was identified, it was blocked and not able to execute attacks any longer for at least the default blocking-time of five minutes. By the evaluation of the detection- and false alarms, a correct classification of about 98.7 percent of all connections was possible. In detail, 92 percent of all conducted attacks had been identified and blocked, while only 0.8 percent of the benign connections had been misclassified as attacks. The false alarm rate of the system is about 0.8 percent while the false alarm ratio (the portion of the alarms which was falsely generated when examine benign traffic) is about 4 percent.

Next, the system performance in a productive network was evaluated. Figure 5 gives an example of a test run over one and a half hour, containing multiple brute-force attacks. It is important to know that in the shown example *all* network traffic passing the transparent bridge was evaluated, not only the traffic of remote sessions. The number of active IPs and used ports are shown as well as the number of concurrently blocked IPs. Because of the readability of the graph, only the beginning of a blocking phase is drawn, not the whole phase or its ending.

In average, about 99.5 percent of all connections had been classified correctly, with a detection probability of about 98.4 percent and a false alarm rate of about 0.5 percent. On the other side, the false alarm ratio of the tests in the productive network is about 51.8 percent. Therefore, half of the generated alarms are based on benign traffic which is quite high. Anyway, this only happens when the *complete* network traffic is examined by the system for the appearance of brute-force attacks, not only the traffic of remote sessions or other login mechanisms. If only the relevant connections are selected and analyzed, the false alarm ratio drops down to about 1.6 percent. This can be achieved by the integration of a protocol analyzer into the system (e.g., see [10], [11], [12]).

V. Related Work

Despite the wide dissemination and broad usage for attacks, only little work is done in the area of brute-force detection. Today's detection mechanisms are typically based on the number of login attempts within a limited time window and originated by a unique IP address. Multiple host-based tools are available which monitor this behavior and block IP addresses, e.g., [13]. There, the evaluation is done by monitoring the logfiles or counting connections made from a single IP address to a configured remote service like *SSH*.

Firewalls can be configured to slow down brute force attacks and block identified IP addresses. For example, the netfilter-firewall can be used to detect brute-force attacks and block IPs by the use of two rules: `iptables -I INPUT -i eth0 -p tcp -m tcp --dport 22 -m state --state NEW -m recent --set --name SSH_BRUTE --rsource` and the second rule `iptables -I INPUT -i eth0 -p tcp -m tcp --dport 22 -m state --state NEW -m recent --update --seconds 300 --hitcount 5 --name SSH --rsource -j DROP`. The first rule dynamically creates a list of IP addresses which is matched as followed: First, the scource address will be added to the list when using port 22. If the address already exists, the entry is updated. Therefore, the already seen addresses can be stored. The second rule monitors the number of connections from this address; if the number of *NEW* connections from the same address and to port 22 exceeds four, the IP address will be blocked for 300 seconds.

The current approaches are easy to use, but have several disadvantages, e.g., only specified ports can be monitored and network performance can be reduced keenly. For example, often the standard port 22 is monitored but *because* of the popular brute-force attacks, many administrators moved the port to non-standard ones. Anyway, attackers often scan their targets, therefore being able to attack non-standard ports, too. Another shortcoming of these approaches is the lack of detecting distributed attacks.

Blacklists are another popular method for avoiding brute-force attacks originated from well-known IP addresses (e.g., [14]). Anyway, because of the intense use of Botnets and therefore an extensive pool of end-user IP addresses, this traditional mechanisms are not adequate any longer. By the distribution of the authentication tries to numerous different IP addresses, a detection based on the number of parallel or consecutive connection tries can be avoided. Also, the use

 ISBN: 978-1-61208-199-1

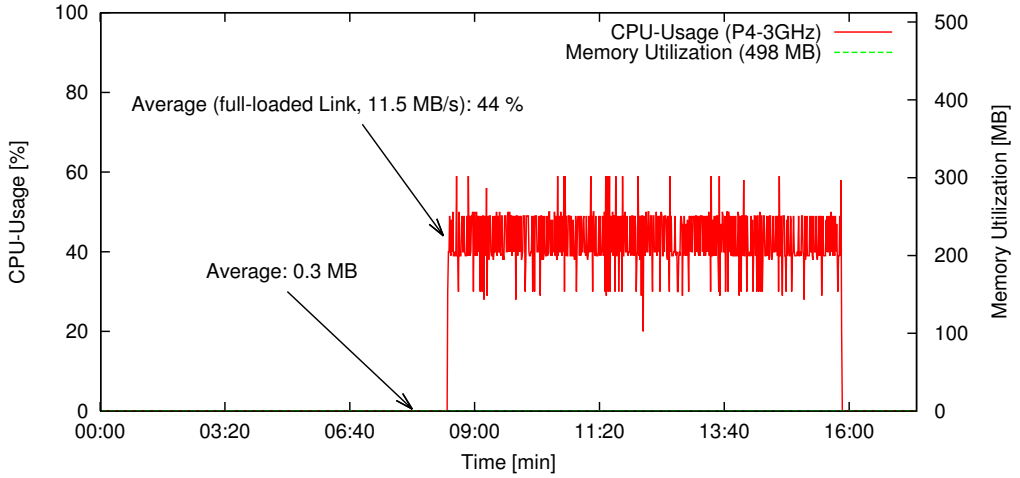

Figure 3. Resource usage of the detection system integrated into a productive network.

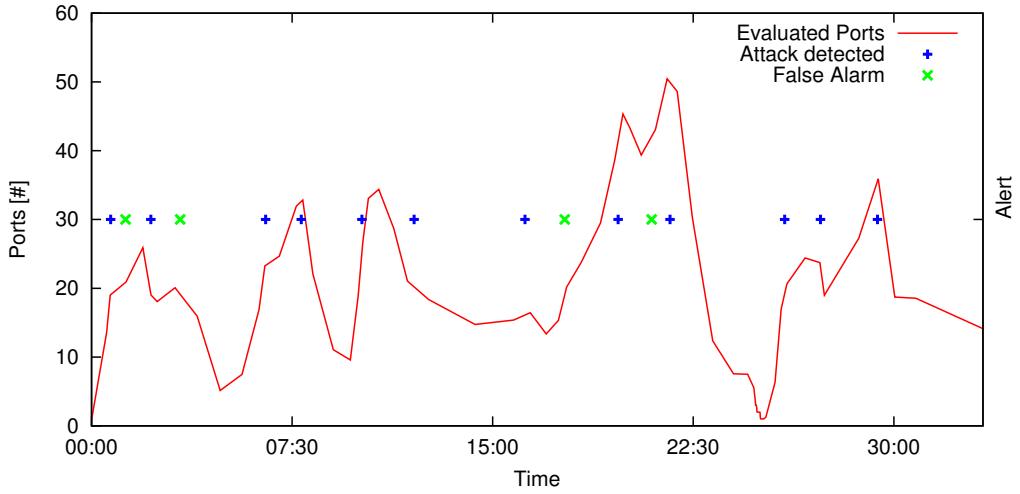

Figure 4. Attack detection and false alarms of the security system in a synthetic environment.

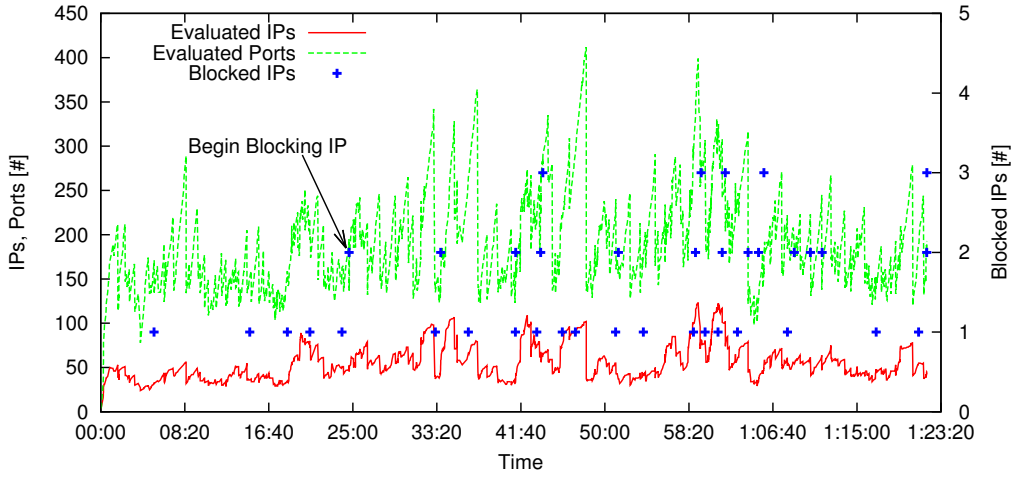

Figure 5. Attack detection of the security system in a productive network.

 ISBN: 978-1-61208-199-1

Table II
DETECTION PROBABILITIES OF THE PROPOSED ARCHITECTURE. VALUES IN BRACKETS: RATIO AFTER FILTERING.

	Classification	Detection Prop.	False Alarm Ratio	False Alarm Rate
Synthetic	98.68	92.0	4.0	0.84
Productive Network	99.53	98.41	51.79 (1.59)	0.48
Combined	99.11	95.21	27.90 (2.80)	0.66

of different idle times, multiple encryption algorithms, etc. can avoid the detection of attacks, too.

VI. CONCLUSION AND FURTHER WORK

Brute-force attacks are a popular and common used attack technique which can greatly endanger systems and networks. Anyway, the current security mechanisms are not able to provide an adequate protection against these attacks.

Therefore, we proposed a new architecture, which is able to provide a fast network-based brute-force detection. Only little system resources are required for the operation of the detection engine. In contrast to existing approaches, the system design enables an attack detection independent of the specific communication between the server and the client, used ports and IPs, the encryption in use, the configuration of the server and the speed and kind of executed brute-force attacks. Therefore, the system can be used in general and without any configuration. All required information is gathered in real-time from the live connections by the use of similarity measurements. It can secure all hosts in a network, disabling the necessity to provide security mechanisms for every single system.

To further improve the detection results and limit the false alarm rates, a pre-filtering of the traffic can be done. At the moment, the complete network traffic is analyzed, but a service detection can be used to analyze only services which are prone to brute-force attacks. By that, the false alarms can be reduced considerably, enabling the system for an application in productive networks. In this connection, we are also going to evaluate the preformance of our system on 1 and 10 Gbps network links.

Also, we are going to adapt the working scheme for a high-efficient Distributed Denial of Service (DDoS) Attack Detection, which is another widespread attack type. By the use of Botnets, DDoS-attacks easily can be used to take down servers and also whole infrastructures of a provider, for example to press money. The presented concepts are the base for an efficient network-based detection and prevention system.

ACKNOWLEDGMENT

This work is done at the Chair for Communication Systems and Internet Services led by Prof. Dr. Dreo Rodosek, part of the Munich Network Management (MNM) Team.

REFERENCES

[1] A. Sperotto, *Flow-Based Intrusion Detection*, PhD thesis, University of Twente, The Netherlands, 2010.

[2] C. Seifert, *Analyzing Malicious SSH Login Attempts*, website, http://www.symantec.com/connect/articles/analyzing-malicious-ssh-login-attempts, last seen on January 19th, 2012.

[3] A. Sperotto, R. Sadre, F. van Vliet and A. Pras, *A Labeled Dataset for Flow-Based Intrusion Detection*, LNCS 5843, pp. 39-50, IPOM 2009, Springer-Verlag Berlin-Heidelberg, 2009.

[4] ATLAS Arbor Networks, *Active Threat Level Analysis System*, website, http://atlas.arbor.net/, last seen on January 19th, 2012.

[5] SSHatter, *SSHatter - Freecode*, website, http://freecode.com/projects/sshatter, last seen on January 22th, 2012.

[6] BruteSSH, *BruteSSH, SSH password brute forcer*, website, http://www.edge-security.com/brutessh.php, last seen on January 22th, 2012.

[7] B. Schneier, *Real-World Passwords*, website, http://www.schneier.com/blog/archives/2006/12/realworld_passw.html, December 2006, last seen January 19th, 2012.

[8] Imperva Application Defense Center, *Consumer Password Worst Practices*, whitepaper, http://www.imperva.com/docs/WP_Consumer_Password_Worst_Practices.pdf, 2011.

[9] C. Herley, *So long, and no thanks for the externalities: the rational rejection of security advice by users*, Proceedings of the 2009 workshop on New security paradigms workshop, pp. 133-144, NSPW 09, ACM, 2009.

[10] L. Bernaille, R. Teixeira, I. Akodkenon, A. Soule and K. Salamatian, *Traffic Classification on the fly*, SIGCOMM Comput. Commun. Rev., 36(2):23-26, April 2006.

[11] A. Moore and K. Papagiannaki, *Toward the Accurate Identification of Network Applications*, In Passive and Active Network Measurement, LNCS 3431, pp. 41-54, Springer-Verlag Berlin-Heidelberg, 2005.

[12] C. Wright, F. Monrose and G. Masson, *On Inferring Application Protocol Behaviors in Encrypted Network Traffic*, J. Mach. Learn. Res., 7:2745-2769, December 2006.

[13] R-FX Networks, website, http://www.rfxn.com/projects/, last seen on January 19th, 2012.

[14] Blacklisting and Abuse Reporting, website, http://www.openbl.org/, last seen on January 19th, 2012.

 ISBN: 978-1-61208-199-1