Using Application-Aware Flow Monitoring for SIP Fraud Detection

Tomas Cejka, Vaclav Bartos, Lukas Truxa, Hana Kubatova

Using Application-Aware Flow Monitoring for SIP Fraud Detection

Tomas Cejka[1], Vaclav Bartos[2], Lukas Truxa[1], Hana Kubatova[3]

[1] CESNET, a.l.e.
Zikova 4, 160 00 Prague 6, Czech Republic
cejkat@cesnet.cz
[2] Faculty of Information Technology, Brno University of Technology
Bozetechova 2, Brno, Czech Republic
ibartosv@fit.vutbr.cz
[3] CTU in Prague, FIT
Thakurova 9, 160 00 Prague 6, Czech Republic
kubatova@fit.cvut.cz

Abstract. Flow monitoring helps to discover many network security threats targeted to various applications or network protocols. In this paper, we show usage of the flow data for analysis of a Voice over IP (VoIP) traffic and a threat detection. A traditionally used flow record is insufficient for this purpose and therefore it was extended by application-layer information. In particular, we focus on the Session Initiation Protocol (SIP) and the type of a toll-fraud in which an attacker tries to exploit poor configuration of a private branch exchange (PBX). The attacker's motivation is to make unauthorized calls to PSTN numbers that are usually charged at high rates and owned by the attacker. As a result, a successful attack can cause a significant financial loss to the owner of PBX. We propose a method for stream-wise and near real-time analysis of the SIP traffic and detection of the described threat. The method was implemented as a module of the Nemea system and deployed on a backbone network. It was evaluated using simulated as well as real attacks.

1 Introduction

Computer networks are a multifunctional communication channel used by various different applications. The example of such an application that is studied in this paper is the telephone service – the Voice over IP (VoIP) technology. This, as well as many other applications, is often considered to be critically important for users. It is therefore important to have effective ways for quick detection of any problems, including security threats. This often means a necessity for monitoring and analysis of the traffic. A common approach allowing situational awareness even in high speed networks is the usage of flow monitoring.

Traditional flow monitoring provides data extracted from packet headers up to the transport layer. Therefore, it provides information about IP addresses,

TCP/UDP ports, TCP flags or ICMP message types in form of flow records. The flow records also contain statistics such as number of packets, number of transferred bytes and information about observation time. However, there is no detailed information about application layer protocols in the flow records.

The traditional flow record is sufficient for many purposes, including detection of several types of malicious traffic. For example, port scanning and SYN flood attacks are easy to detect using only these basic flow data, since these attacks have clearly distinguishable characteristics on network and transport layers. Even some attacks on application layer, such as dictionary attacks on SSH, can be detected using basic flow data with proper algorithms [7]. However, this is not always possible or it may be very difficult and unreliable. For some kinds of malicious traffic, knowledge of additional information from the application layer is necessary for reliable detection.

Fortunately, application awareness has been implemented into some flow exporters in the last years, usually in the form of plugins [9]. Such exporters inspect packet payload, extract information from headers of application layer protocols and add this information into flow records. The *extended flow records* can contain e.g. URLs, response codes from HTTP, or domain names from DNS requests along with common features of the flow record. The flow records are then transferred to a collector using the IPFIX protocol [3].

The extension of flow records by application layer information was added mainly to allow more detailed statistics about traffic or to support application performance monitoring. However, it can be used for detection of security threats as well. In this paper, we show an example of such a usage.

We focus on monitoring of the VoIP traffic, in particular the Session Initiation Protocol (SIP). Our goal is to detect one of the most common VoIP frauds – the one in which an attacker tries to misuse a poorly secured gateway to the public switched telephone network (PSTN) to make unauthorized calls. Such calls are often made to premium-rate numbers (operated by the attackers) causing significant financial losses to operators of the misused gateway.

According to [4], worldwide losses due to VoIP hacking and calling to premium rate services go to billions of dollars per year. It is one of the most costly fraud types in the telecommunication industry. Therefore, even though successful attacks are not very usual, it is highly important to detect them as soon as possible, before a significant damage is caused.

The rest of this paper is organized as follows. The next section describes the related work. Sec. 3 describes details about the attack on which we focus. Sec. 4 describes the detection method and our implementation of it, followed by Sec. 5 where it is evaluated. Sec. 6 concludes the work.

2 Related Work

Attacks and security threats are an ordinary part of network traffic. There are several different types of attacks that are targeted to VoIP infrastructure. Some possible attacks and vulnerability exploits are shown in [5] by El-Moussa et

al. The paper describes denial of service attacks, which are very common in computer networks, or SPAM over internet telephony. Furthermore, the authors mention brute-force attacks against authentication mechanism, which are somewhat similar to the prefix-guessing attacks described in this paper. They however do not present any countermeasures or a way of detection of such attacks.

Another work enumerating possible attacks against VoIP technology is a survey by A. Keromytis [12]. Although it summarizes hundreds of papers from the area, there is no mention about the kind of attack we deal with, nor any work using flow measurement to detect security threats in VoIP traffic.

To our best knowledge, the only paper providing a way of detection of toll fraud attempts is [8] by Hoffstadt et al. It is focused on monitoring of VoIP threats using honeypots. The authors describe principle of toll fraud based on hijacking of a SIP account. An attacker that gains a user's identity is able to establish a phone call that can be charged. Discovery of a fraud is based on an off-line analysis of honeypot logs. Even though the authors stated that they observed manual attempts of toll fraud, we are observing automatic brute-force guesses of dialing prefixes. Also, our flow-based approach allows to monitor traffic going to real gateways, not only honeypots, therefore we are able to detect real attacks and possibly raise alerts on the successful ones.

Besides common hardware and software VoIP phones, there are several tools that allow users to communicate over SIP in unusual ways. For example, they allow to craft a request with any values of headers or to perform brute-force attacks automatically. Examples of the tools are [6, 11, 13, 15].

A detailed description of the flow monitoring technology can be found in the article [9] by Hofstede et al. It also contains an overview of available software. The paper [14] by Velan and Celeda introduces the concept of application-aware flow monitoring.

3 Principle of the Phone Call Fraud

The following subsection provides a brief introduction to VoIP telephony and SIP. We focus only on aspects needed to understand the type of fraud discussed in this paper and the proposed detection method; we knowingly leave out many otherwise important details for brevity.

3.1 Short Introduction to SIP

Voice over IP is a technology for transferring voice and multimedial data over computer networks. Session Initiation Protocol (SIP) is the well-known protocol used for initiation, control and termination of VoIP sessions (or calls). The multimedial data are transferred in a separate channel using Real-time Transport Protocol (RTP).

SIP is a protocol based on a request-response transaction model. Every device can act both as a client or as a server. The client creates and sends requests, the

server receives and processes them, and generates one or more replies. The high-level architecture consists mostly of end devices (hardware or software phones) and SIP proxy servers. The proxies receive requests for calls, localize called parties and route the requests to them (possibly via other proxies). They also route replies on their way back to the callers. The proxies also provide authentication services and several other tasks, which are out of scope of this short introduction.

There are several types of requests in SIP. The one which is crucial for this work is `INVITE`. It is used to initiate a new call. As any request in SIP, it carries several headers describing parameters of the request. The most important headers of `INVITE` request are:

- *Request-URI:* Used for addressing the called party. It is usually in the form `sip:user@host`, although more complex forms are possible. The *user* part usually consists of user name or phone number and the *host* part is the destination server where the request should be sent to (domain name or IP address).
- *To:* Called party identification.
- *From:* Identification of the caller.
- *Call-ID:* Unique identifier of a call, usually a long random string.

When `INVITE` is sent by a client to a proxy server, the request is propagated to the destination, possibly via other proxy servers. Responses such as `TRYING` and `RINGING` are returned and when the called party eventually indicates that it is ready to establish the call, the `OK` response is sent to the caller and the multimedia transfer begins. When the connection can not be established for some reason, a reply with corresponding error code is returned.

If a SIP proxy server is deployed in some private organization to serve as a central hub for internal phone communication and as a proxy for communication with outer world, it is usually called a Private Branch Exchange (PBX). These PBXs usually operate with both VoIP as well as classic telephone networks (PSTN), acting as gateways between those two technologies. The following text focuses on misuse of such gateways.

3.2 Principle of the Fraud

Depending on configuration a PBX may allow users to make VoIP calls to PSTN numbers by setting the destination user ID in an `INVITE` request to the called number prepended by a special prefix. For example, to call a PSTN number 555-555-0123 a SIP call to `995555550123@example.com` needs to be performed, assuming that the gateway is located at `example.com` and it has "99" configured as the prefix for PSTN calls.

The attack is based on finding poorly secured PBXs and using them to make fraudulent calls to PSTN. Motivations to make such calls may differ, but the common one is to gain money by calling to paid services. This is outlined in Fig. 1. The attacker first loans a premium-rate phone number[4], usually in a

[4] *I. e.* a number which is charged at a high rate in favour of the line operator.

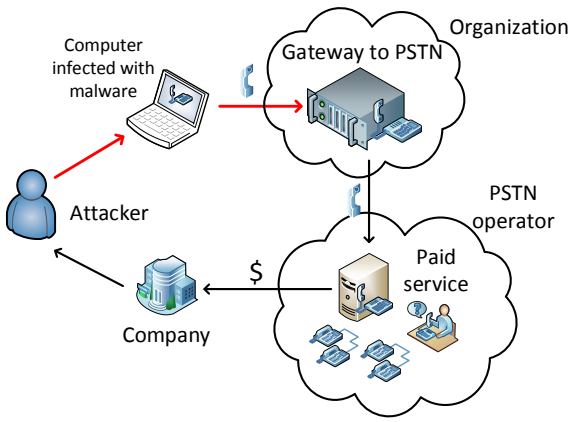


Fig. 1: Principle of the SIP fraud.

foreign country and via some intermediary company. Any calls to that number generate revenue for the attacker. Computers controlled by the attacker are then instructed to find open PBXs and make fake calls via them. When a call is successful, the PSTN operator charges the organization operating the PBX. The money goes to the company operating the premium line and to the attacker.

In order to make calls to PSTN via a PBX, the attacker needs to know the prefix which must be prepended in front of the number. Since this prefix depends on a particular PBX and its configuration, attackers usually do not know it. They must therefore guess it by trying various possibilities until the correct prefix is found and the call is successful or until all possibilities from a dictionary are used and the attacker moves on to another victim.

Such guessing can be recognized as a large number of `INVITE` requests from the same source, all trying to call the same number but with different prefixes. A typical sequence of URIs in the `INVITE` requests is shown in Fig. 2.

Such a sequence typically contains tens of `INVITE` requests with different prefixes. All the prefixes may be tried within a few minutes, but attackers often try to evade detection by putting long intervals between individual trials, so it may take up to several days. Such slow attacks are harder to notice in logs and generally harder to detect by any means.

In some cases, PBXs are configured insecurely and allow to make such calls without a proper authentication. More secured PBXs require an authentication header in the `INVITE` request. In such a case, attackers can perform a dictionary attack first, in order to find login credentials of some user. If some login and password is successfully found, the attacker may impersonate the user and the prefix guessing can be run in the same way as described. This paper focuses solely on the prefix guessing part of attacks. The detection method described in the next section makes no difference between authenticated and non-authenticated users.

00972592577956@A.B.C.D	**9999990**972592577956@A.B.C.D
000972592577956@A.B.C.D	**99999990**972592577956@A.B.C.D
900972592577956@A.B.C.D	**999999900**972592577956@A.B.C.D
+972592577956@A.B.C.D	**9999999900**972592577956@A.B.C.D
972592577956@A.B.C.D	**99999999900**972592577956@A.B.C.D
100972592577956@A.B.C.D	**9000**972592577956@A.B.C.D
800972592577956@A.B.C.D	**0**972592577956@A.B.C.D
600972592577956@A.B.C.D	**0000**972592577956@A.B.C.D
700972592577956@A.B.C.D	**0000000**972592577956@A.B.C.D
400972592577956@A.B.C.D	**00000000**972592577956@A.B.C.D
300972592577956@A.B.C.D	**000000000**972592577956@A.B.C.D
200972592577956@A.B.C.D	**0000000000**972592577956@A.B.C.D
500972592577956@A.B.C.D	**9100**972592577956@A.B.C.D
9990972592577956@A.B.C.D	**990**972592577956@A.B.C.D
99990972592577956@A.B.C.D	**9100**972592577956@A.B.C.D
999900972592577956@A.B.C.D	. . .

Fig. 2: An example of URIs called during a typical prefix guessing attack (IP address anonymized).

4 Detection Method

The detection method is designed to work without any prior knowledge of VoIP infrastructure and dialing plans on the network. The assumed deployment is on an ISP level or in a network of a large organization, where an operator of the detection system has no direct control over VoIP equipment but still wants to know about any issues related to it.

The detection is based on an analysis of SIP INVITE requests trying to make calls to PSTN numbers. The goal is to find IP addresses that generate large number of such requests varying only in a prefix of the called number. The detection method works even if prefixes are tried in a very low rate (e.g. one attempt per day). Also, it was needed to design the method to be efficient since we are targeting large networks with high volumes of traffic.

The input data comes from flow monitoring probes. Basic flow records are not sufficient for the detection of the SIP fraud, it is needed to extract additional information from SIP headers. In particular, we extended the flow records by *request/response code*, *Request-URI* and *To*, *From*, *Call-ID* and *User-Agent* headers from SIP messages. We achieved this by using a plugin for FlowMon probes [10]. It is the probe able to monitor high speed networks and parse information from application layer protocols. The whole monitoring infrastructure is shown in Fig. 3. Data from the monitoring probes are passed to a collector in the IPFIX format [3] and then into the Nemea system [1, 2] – a modular framework for network traffic analysis and anomaly detection. The detection method described in the following paragraphs was implemented as a software module for the Nemea system. It receives and analyses extended flow records of SIP traffic and reports detected attacks.

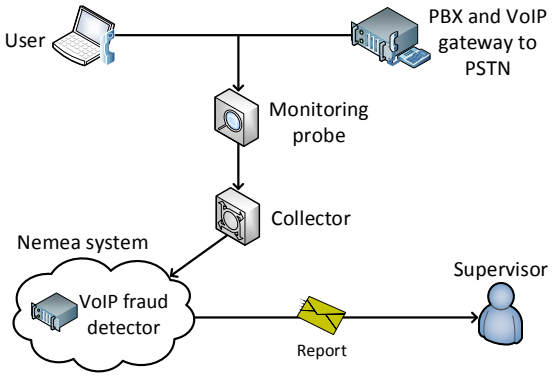


Fig. 3: Infrastructure of the monitoring system with the detection module.

The detection algorithm works as follows. For each incoming flow record carrying information about an INVITE request a called party identification is taken from *Request-URI* (or *To* header, depending on configuration; however, both are usually the same). If its *user* part, i.e. the part before the `@`, contains only digits or some of the allowed special symbols (`+`, `*`, `#`, `-`, `:`) it is further processed. Otherwise, the message is ignored since it is not a call to a phone number.

Responses to the INVITE messages are also processed and are used for determination whether the call was successfully established. In particular, when an OK response is observed after a previous INVITE request and their *Call-ID* headers match, the call is considered to be successful.

A set of URIs observed in INVITE requests is stored for each source IP address. In order to allow efficient storage and analysis of such sets, the URIs are stored into a specially designed data structure based on the suffix tree. Figure 4 shows an example of a set of URIs stored in such a tree. In the suffix tree, the common suffix of two or more URIs is represented by a parent node while the children nodes (or subtrees) represent their different prefixes. There is a rule that none of nodes can have a common part with its sibling. Therefore, in case a newly inserted URI contains an unknown prefix which has a common part with some existing prefix, it can cause a split of an existing node.

Each node represents an URI given by its value concatenated with values of all its ascendants. Each node also contains a number of call attempts to that URI, number of successfully established calls and other information, mostly for optimization of the detection algorithm.

Such a tree is constructed for each source IP address and is continuously updated as new INVITE requests from that address are observed. The trees are periodically analyzed in order to detect prefix guessing attacks. As shown in Sec. 3.2, during such attack a large number of URIs is observed with the same phone number and destination host but many different prefixes. That results in

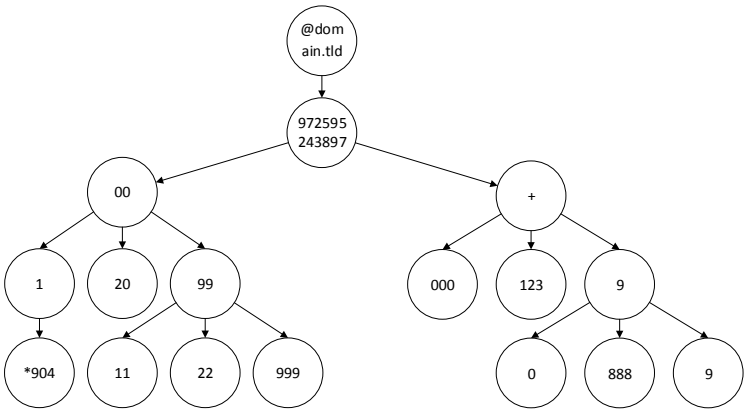


Fig. 4: Suffix tree for analysis of phone number prefixes.

a tree in which there is a single node which contains the phone number and have a large number of descendants.

The algorithm for detection of such a node works with two parameters – maximal prefix length (l_{max}) and a threshold on number of tested unique prefixes (T). At first, the tree is traversed from the bottom to the top (i.e. from leaves to the root). For each leaf node, the algorithm goes up through its ascendants until the total length of numbers stored in the visited nodes exceeds l_{max}. The final node potentially represents the called number. Then, the number of its descendants satisfying the following two conditions is counted: 1) the prefix represented by a node must be shorter than l_{max} and 2) there must be an unsuccessful call attempt made to the node's URI. If the number of such descendants is the same or higher than the threshold T, an attack is reported. Otherwise, the algorithm continues traversing the tree from another leaf node.

After the attack is reported, all related nodes are removed from the tree, so the same attack is not reported in the next run of the algorithm. Basic information about the attack is however kept. Therefore, if the attack continues by trying another prefixes and their number again exceeds the threshold, so it is detected as an attack, it is recognized that it is only a continuation of the attack reported earlier. The new detection is thus reported only as an update of the previous one.

Besides the suffix tree, some other information is stored per IP address, mostly for the purpose of reporting. This information includes the time of the last seen SIP message, the time of the last detected attack or the value of the *User-Agent* header.

The detector is designed for continuous processing of potentially infinite stream of data from the network. Some of the incoming data are stored in memory, but because the capacity of available memory is always limited, old data must be periodically removed. Most of the data removals are based on a simple timeout. If no SIP communication from an IP address has been detected for a

given time period, all information about that address is removed. The default timeout in our implementation is 14 days. Also, if a suffix tree of some address grows into a huge size (we use a threshold of 100,000 nodes), the whole tree is removed. Because such a big tree is often the result of a large attack, the detection algorithm is applied to the tree before removal. Finally, nodes representing prefixes in an attack are removed after the attack is reported, as was described earlier.

5 Evaluation

In order to evaluate the detection method, we prepared a SIP server simulating a PBX with a gateway to PSTN. The server was configured to not require authentication and to allow calls to PSTN using a three-digit prefix. A modified SIPVicious tool [6] (`svwar.py` script) was used to generate attacks to the server from several sources. The simulated attackers tried to call to a number with randomly changing prefixes until they guessed the correct one. The traffic between the attacking machines and the server was monitored by the detector. Both parameters of the detection algorithm, that is the maximal length of a prefix (l_{max}) and the minimal number of call attempts that is considered as a guessing (T), were set to 10.

At first, the tests were performed in a virtual environment with no other traffic than the generated attacks. As expected, all attacks were successfully detected and reported, except a few cases in which the correct prefix was guessed in less than 10 tries (such attacks could be detected as well by decreasing the threshold T, but too low threshold might cause false alerts when deployed on real network).

We continued by tests in the real environment – in the CESNET2 network. CESNET2 is the academic network of the Czech Republic, connecting Czech universities and many other organizations to the Internet (around 1 million IP addresses in total). Its perimeter – 10 peering links, all with wire speed of 10 Gbps – is monitored using FlowMon probes. The total traffic on these links ranges from 5 Gb/s at night to 25 Gb/s during the day (50k to 150k flow records per second). The average amount of SIP traffic is around 50 flows per second, with occasional peaks up to several hundreds.

The probes were running the plugin for extending flow records with values of SIP headers. The detector was deployed into an operational instance of the Nemea system which receives and analyzes flow records from all the probes. The SIP server and the machine simulating attacks were placed so that the traffic between them is observed by one of the monitoring probes. Configuration of the server, attacks and the detector was the same as before.

Despite the generated attacks were hidden in a lot of real traffic now, all the attacks consisting of at least 10 attempts were successfully detected again, no matter how slow or fast they were. A lot of real attacks were detected, too.

In a measurement period of two weeks, the detector received and analyzed 10.5 million flow records corresponding to SIP INVITE messages. There were

Tab. 1: Top-20 prefixes observed in the backbone network traffic.

Prefix	Count	Succ. calls	Prefix	Count	Succ. calls
00	3800	22	9	1946	2
000	3412	9	810	1706	6
900	3273	12	9000	1608	8
+	3072	13	9900	1599	4
(none)	2498	8	9011	1582	7
0000	2464	14	99900	1462	10
011	2286	3	9009	1330	2
800	2248	5	9810	1323	9
0011	2092	4	005	1303	2
009	1982	5	001	1297	8

15,992 prefix guessing attacks reported consisting of 201,438 INVITE messages. That means that on average 12.6 prefixes are tried by a single attacker at a single gateway. Around 1.9 % of all INVITE messages were marked as part of this kind of attacks (although we expect that many of the others are malicious as well, since authentication attacks generate a lot of INVITE messages, too). Approximately 1.1 % of attacks seemed to be successful[5], i.e. a call was successfully established.

During its operation, the detection module consumed about 200 MB of memory and took only about 5 % of CPU on average (Intel(R) Xeon(R) CPU E5-2630 @ 2.30 GHz).

During our testing of the detection module, we gathered a lot of information and statistics about the SIP attacks as well as the SIP traffic in general. The interesting results are summarized in the rest of this section.

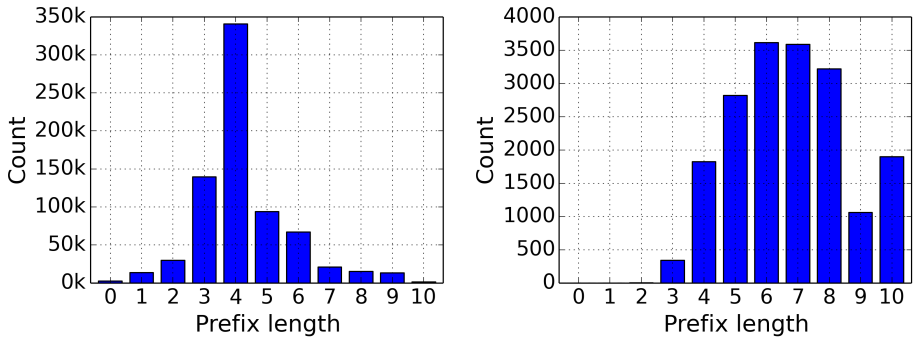
Table 1 shows a list of the 20 most often observed prefixes that were tested by attackers. The data for the table was taken from a two week period. The "Count" column represents the number of times the prefix was used and the "Succ. calls" represents the number of successful calls that were observed with the prefix.

Figure 5a shows the histogram of lengths of all prefixes that were used in attacks reported within a two week period. Figure 5b shows the histogram of the longest prefixes that were used in the reported attacks. That means it shows the maximal length of prefixes used in individual attacks. It can be observed that while most prefixes have 3 or 4 digits, attacks almost always contain some longer prefixes as well.

Table 2 shows the most frequent values of the *User-Agent* header that were observed on the backbone network. The data for the table was taken from a one week interval. There were 384 distinct values of the header. As can be seen, the most often used user agents (*sipcli*, *friendly-scanner*[6]) are scripts that allow users/attackers to craft SIP messages with any values of headers. They can be

[5] Not every successful attempt necessarily means a breach of a real gateway. Some of the gateways may in fact be honeypots.

[6] *friendly-scanner* is the default *User-Agent* value used by SIPVicious tool.



(a) (b)

Fig. 5: Histogram of the lengths of a) all prefixes tried in attacks, b) the longest prefixes tried in attacks.

Tab. 2: The most frequent values of the *User-Agent* header.

User-Agent	**Count**
sipcli/v1.8	643,312
friendly-scanner	424,178
Cisco-SIPGateway/IOS-12.x	6,304
FPBX-2.10.1(1.8.7.1	1,153
Asterisk PBX 11.11.0	570
None or empty	2,440
Other values	7,220

used to generate e.g. phone numbers of the callee in the case of prefix guessing. Of course, the *User-Agent* header cannot be a reliable source of information since a client can fill in any string. However, a legitimate client usually does not have any reason to present itself by the name of another tool and malicious clients apparently do not do that often.

6 Conclusion

This paper presented a possible usage of the emerging technology of application-aware flow monitoring in the area of security threat detection. In particular, a method for detection of VoIP-based toll fraud – a network attack that can lead to a significant financial losses – has been proposed. The detection is enabled by special flow monitoring probes which are able to extend flow records by information from application-layer protocols.

Using exported headers of Session Initiation Protocol (SIP), the proposed detection module is able to analyze SIP transactions and detect attempts to guess a prefix configured on a PBX to allow calls to PSTN. It is also able to

detect whether any of the attempts was successful. A successful call after a previous guessing indicates that the attacker found a way to make unauthorized calls via the PBX. In such a case it is necessary to alert an operator of the PBX immediately.

An implementation of the method was deployed and evaluated on a real backbone network. Some interesting results of the SIP analysis in real network traffic were presented in Sec. 5. The information from the extended flow records allows us to observe statistics about *User-Agent* headers and called phone numbers, for example. The detection capabilities of the proposed method are very good according to our experiments – all simulated attacks were successfully detected, as well as many real attacks. In fact, any attack consisting of at least 10 `INVITE` requests in a time window of 14 days is detected in default configuration. Of course, these thresholds can be tuned to fit requirements of the network operators.

While these attacks may be detected by other methods as well, for example by analysing logs of SIP servers, the flow monitoring approach allows to monitor all SIP servers in the network from one place, without necessity of access to the servers (which are usually operated by other people than those responsible for security). Moreover, if the detector is deployed on backbone links, like in our test scenario, it allows to observe attacks from many different sources to many different destinations, which is impossible with other methods (log parsing or honeypots). It provides us with more complete view on attacks and attackers. For example, by deep analysis of attack characteristics and their sources, it may be possible to detect groups of IP addresses attacking collectively, which can lead to revealing botnets.

Acknowledgments This work was partially supported by the "CESNET Large Infrastructure" (LM2010005), CTU grant No. SGS15/122/OHK3/1T/18 funded by the Ministry of Education, Youth and Sports of the Czech Republic, and BUT grant FIT-S-14-2297. This work was also supported by the IT4Innovations Centre of Excellence project (CZ.1.05/1.1.00/02.0070), funded by the European Regional Development Fund and the national budget of the Czech Republic via the Research and Development for Innovations Operational Programme, as well as Czech Ministry of Education, Youth and Sports via the project Large Research, Development and Innovations Infrastructures (LM2011033).

References

1. Bartos, V., Zadnik, M., Cejka, T.: Nemea: Framework for stream-wise analysis of network traffic. Tech. rep., CESNET (2013)
2. CESNET: Nemea, `https://www.liberouter.org/nemea/`
3. Claise, B., *et al.*: Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information. RFC 7011 (Sep 2013)
4. Communications Fraud Control Association: 2013 CFCA Global Fraud Loss Survey. Press release (October 2013), `http://www.cfca.org/pdf/survey/CFCA2013GlobalFraudLossSurvey-pressrelease.pdf`

5. El-Moussa, F., Mudhar, P., Jones, A.: Overview of sip attacks and countermeasures. In: Information Security and Digital Forensics, pp. 82–91. Springer (2010)
6. Gauci, S.: SIPVicious. Tools for auditing sip based voip systems (2012), https://code.google.com/p/sipvicious/
7. Hellemons, L., Hendriks, L., Hofstede, R., Sperotto, A., Sadre, R., Pras, A.: SSHCure: A Flow-Based SSH Intrusion Detection System. In: Dependable Networks and Services, LNCS, vol. 7279, pp. 86–97. Springer (2012)
8. Hoffstadt, D., Marold, A., Rathgeb, E.: Analysis of SIP-Based Threats Using a VoIP Honeynet System. In: Proccedings of the 11th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom). pp. 541–548 (June 2012)
9. Hofstede, R., Celeda, P., Trammell, B., Drago, I., Sadre, R., Sperotto, A., Pras, A.: Flow monitoring explained: From packet capture to data analysis with netflow and ipfix. IEEE Communications Surveys Tutorials 16(4), 2037–2064 (2014)
10. INVEA-TECH a.s.: FlowMon Probe – High-performance NetFlow Probe up to 10 Gbps, http://www.invea-tech.com/products-and-services/flowmon/flowmon-probes
11. KaplanSoft: SipCLI, http://www.kaplansoft.com/sipcli/
12. Keromytis, A.D.: A comprehensive survey of voice over ip security research. IEEE Communications Surveys & Tutorials 14(2), 514–537 (2012)
13. Ohlmeier, N.: SIP Swiss Army Knife (Sipsak), http://sourceforge.net/projects/sipsak.berlios/
14. Velan, P., Celeda, P.: Next generation application-aware flow monitoring. In: Monitoring and Securing Virtualized Networks and Services, LNCS, vol. 8508, pp. 173–178. Springer (2014)
15. VoP Security: SiVuS (SiP Vulnerability Scanner) – User Guide v1.07, http://www.voip-security.net/pdfs/SiVuS-User-Doc1.7.pdf