Labeled VoIP Data-set for Intrusion Detection Evaluation

Mohamed Nassar, Radu State, and Olivier Festor

INRIA Research Center, Nancy - Grand Est
615, rue du jardin botanique, 54602
Villers-Lès-Nancy, France

Abstract. VoIP has become a major application of multimedia communications over IP. Many initiatives around the world focus on the detection of attacks against VoIP services and infrastructures. Because of the lack of a common labeled data-set similarly to what is available in TCP/IP network-based intrusion detection, their results can not be compared. VoIP providers are not able to contribute their data because of user privacy agreements. In this paper, we propose a framework for customizing and generating VoIP traffic within controlled environments. We provide a labeled data-set generated in two types of SIP networks. Our data-set is composed of signaling and other protocol traces, call detail records and server logs. By this contribution we aim to enable the works on VoIP anomaly and intrusion detection to become comparable through its application to common datasets.

1 Introduction

Voice over IP (VoIP) has become a major paradigm for providing flexible telecommunication services while reducing operational and maintenance costs. The large-scale deployment of VoIP has been leveraged by the high-speed broadband access to the Internet and the standardization of dedicated protocols. The term is often extended to cover other IP multimedia communications in general and convergent networks. VoIP services are much more open if compared to PSTN networks. A typical VoIP service is composed by three main parts: the user premises, the VoIP infrastructure for signaling and media transfer, and a number of supporting services (e.g. DNS, TFTP). From a technical point of view, a SIP-based VoIP service is similar to an email service more than a conventional telecommunication service. Hence, VoIP suffers from the same threats of the TCP/IP networks and services. In fact, VoIP faces multiple security issues including vulnerabilities inherited from the IP layer and specific application threats. The attacks against VoIP can typically be classified into four main categories [1]: (a) service disruption and annoyance such as flooding and SPIT (Spam over Internet Telephony), (b) eavesdropping and traffic analysis, (c) masquerading and impersonation such as spoofing, (d) unauthorized access and fraud such as Vishing (Voice over IP phishing). These attacks can have significant consequences on the telephony service, such as the impossibility for a client of making an urgent phone call.

The research community started to investigate the best ways of protection, detection and response for the VoIP services [2]. Researchers argue that intrusion detection is necessary to struggle against VoIP fraudsters. The main difficulty remains to obtain real-world traces of attack incidents or even normal traffic. Labeled data-sets are necessary to evaluate the accuracy results of the proposed techniques which are often based on learning. It is also hard to obtain these data from VoIP providers which are constrained by user privacy agreements.

In this context, we propose a framework to annotate and customize VoIP traffic. The normal traffic is generated by profiled emulated users based on a social model. The attack traffic is based on currently available VoIP assessment tools. We deploy and configure two illustrative VoIP networks in a local test-bed and we generate a labeled data-set that can be used to compare different detection algorithms. The data-set is composed of signaling and other protocol traces comprising call detail records and server logs. Thus, correlation approaches can also be applied to the data.

The rest of the paper is composed as follows: In Section 2 we expose the related works on VoIP intrusion detection. In section 3 we describe the data types. Section 4 presents the traffic generation model and the generation of attacks. Section 5 describes the test-bed and the labeled data-set. Finally Section 6 concludes the paper and discusses the future work.

2 VoIP Intrusion Detection

An Intrusion Detection System (IDS) is a second line of defense behind protection systems like authentication, access control and data encryption. Intrusion detection refers to the process of monitoring computer networks and systems to track violations of the applied security policies. VoIP intrusion detection consists on automating the analysis of specific VoIP data sources. Many works have focused on the Session Initiation Protocol (SIP [3]) as the de-facto signaling protocol for the Internet and the Next Generation Networks (NGN).

Basically, SIP allows two communicating parties to set up, modify and terminate a call. Text-based with heritage from HTTP and SMTP, SIP is a request-response transaction-based protocol. A SIP Dialog is composed of one or more transactions. The SIP addressing scheme is based on URIs (Uniform Resource Identifier) in the form of `sip:user@domain`.

Niccolini et. al. [4] propose a network-based IDS to be deployed at the entry point of the VoIP network. This system is composed of two stages of detection: a rule-based stage and a behavioral-based stage. It implements a preprocessing logic of SIP into the SNORT IDS. SCIDIVE [5] is a knowledge-based IDS based on a rule matching engine. This system performs two kinds of detection: stateful and cross-protocol. The stateful detection focuses on multiple incoming packets belonging to the same protocol. The cross-protocol detection scheme spans packets from several protocols, e.g. a pattern in a SIP packet is followed by another pattern in a RTP packet. SCIDIVE has to be deployed at multiple points of the VoIP network (clients, servers, proxies) in order to correlate events at a global level. Sengar et. al. [6] propose specification-based detection on an extended finite state machine representing the interaction between the different protocols. By tracking deviations from the interacting protocol state machine, a series of SIP, RTP and cross-protocol attacks are detected. The VoIP Defender [7] is a highly-scalable architecture building a generic framework for supporting the detection algorithms.

To struggle against flooding, Chen [8] modifies the original state machines of SIP transactions and applies detection thresholds on the error statistics generated by those modified machines. Similarly in [9], the authors apply thresholds at three different scopes: transaction level, sender level and global level based on a modified User-Agent Server (UAS) state machine. SIP DNS flooding (flooding by SIP requests containing irresolvable domain names) are considered by Zhang et. al. [10] who propose practical recommendations in order to mitigate their effects. Sengar et. al. [11] propose a VoIP Flood Detection System (vFDS). The vFDS measures the difference between two probability distributions (one at training and one at testing) using the Hellinger Distance. In [12], the authors propose a general methodology for profiling SIP traffic at several levels: server host, functional entity (registrar, proxy) and individual user. They propose an entropy-based algorithm detecting SPIT and flooding attacks. As noticed, the learning plays a major role in most of the cited approaches.

Many of the cited contributions have similar approaches although they can not be compared because they are evaluated on different and often unavailable data-sets. A common data-set for evaluating VoIP intrusion detection is missing. We aim by this contribution to provide a totally repeatable test-bed for generating and customizing VoIP traffic as well as a labeled data-set composed of normal, different kind of attacks and mixed normal/attack traces.

3 Data Types

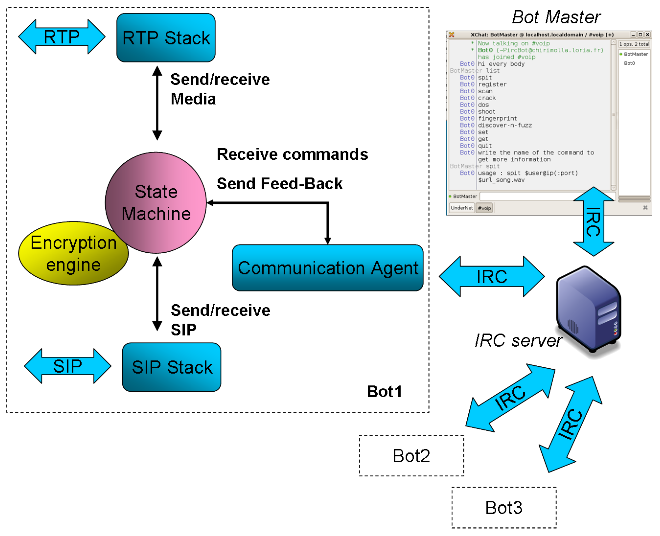
Monitoring distributed applications such as VoIP is challenging because in many cases monitoring only one source of data is not sufficient to reveal the complete picture of some malicious activities. Therefore we have decided to provide three types of data sources: the network traffic, the call detail records and the server logs. Each of these types is important for a group of detection approaches. Moreover, correlation of patterns spanning multiple data sources may reveal more evidence about certain anomalies. Next, we highlight the importance of each type:

– The network traffic -especially the protocols that are essential to the normal operation of VoIP (e.g. SIP, DNS)- is the typical data source for network-based anomaly detection. In [13], we defined 38 statistics (or probes) on the SIP traffic and showed their efficiency for detecting a series of flooding and SPIT attacks.
– The Call Detail Records (CDR) are particularly important for fraud detection and SPIT mitigation. The call history helps building individual user profiles and peer groups in order to detect fraudulent calls.

– The log and statistics of VoIP servers are the typical data source for host-based intrusion detection. For example, the Opensips SIP proxy[1] provides six groups of probes: core, memory, stateless statistics, transaction statistics, user location and registration. These probes can be monitored online in order to reveal abnormal processing and memory loads.

4 Traffic generation model

Several investigations use tools like Sipp[2] for SIP traffic generation. Sipp generates traffic by repeating call flow scenarios described in custom XML, hence it is convenient for benchmarking and assessing stress conditions at SIP servers. However Sipp -in its current state- is not able to emulate real SIP traffic. In fact, Sipp doesn't represent a SIP user-agent state machine and is more oriented towards the transaction layer than the call layer. Otherwise, our unit of traffic generation is the VoIPBot (the source code is available at [14]) which we have initially designed as an attack tool [15]. The new version of the VoIPBot supports emulation of a normal user profile. Its design is based on four main components: the protocols stack, the communication agent, the encryption engine and the SIP state machine as shown in Figure 1.



```
        * Now talking on #voip
        * Bot0 (~PircBot@chirimolla.loria.fr)
          has joined #voip
     Bot0 hi every body
BotMaster list
     Bot0 spit
     Bot0 register
     Bot0 scan
     Bot0 crack
     Bot0 dos
     Bot0 shoot
     Bot0 fingerprint
     Bot0 discover-n-fuzz
     Bot0 set
     Bot0 get
     Bot0 quit
     Bot0 write the name of the command to
          get more information
BotMaster spit
     Bot0 usage : spit $user@ip(:port)
          $url_song.wav
```

Fig. 1. Using VoIP Bots for simulating real traffic

The protocols stack provides the bot with an application interface to use the different signaling and media transfer protocols. We use the Jain-SIP stack

[1] `http://www.opensips.org`
[2] `http://sipp.sourceforge.net`

for sending and receiving, manufacturing and parsing SIP messages. We use the Java Media Framework (JMF) stack for coding and decoding, compressing and expanding, encapsulation and demultiplexing of RTP media flows. The communication agent is based on the Internet Relay Chat (IRC) in order to exchange information and commands with the bot manager. In fact the bot connects to a pre-defined room on the manager IRC server at bootstrap. This scheme allows us to manage several groups of bots in an easy and efficient way. The encryption engine enables the bot to create digest authentication from credentials when authentication is required within the registration process. The SIP state machine drives the operations of the bot with respect to the commands issued by the manager. It defines the bot behavior upon occurrence of SIP events (i.e. receiving a SIP request `RequestEvent` or a SIP response `ResponseEvent`) and `TimeOut` events. The transition from a state to another is constrained by predicates on a set of global and local variables. The bot is configured to generate normal traffic based on the following parameters:

– The SIP port and IP: Currently, we are launching several bots from the same machine/IP. Each bot has a SIP port based on its identifier (e.g. Port 5072 is reserved to Bot12). Therefore, any SIP user-agent in the traces must be identified by the IP/port(SIP) couple rather that only the IP. The virtualization techniques can be further used in order to provision an IP for each bot.
– The SIP registrar: The bot registers to one or more SIP registrars using a user-name and a password in order to initiate and receive calls. For convenience, both the username and the password of a bot are set to its identifier.
– The probabilistic distribution of emitting calls: Traditionally in PSTN models the call arrival follows a Poisson distribution. The VoIP calls follow the same distribution and the aggregated VoIP traffic has fractal characteristics [16]. We setup the bots to make calls with respect to a Poisson distribution by peaking one destination form a phonebook. The mean number of calls per unit of time (λ) has to be defined. The bots have different seeds (based on their identifiers) for their pseudo-random generators. Each bot sleeps for a random boot-time before starting the emission of calls.
– The probabilistic law of call holding time: Currently the bot sets the duration of a call with respect to an exponential distribution such as in PSTN models. The mean of the call duration ($1/\lambda$) has to be defined. If the callee doesn't hang-off during the chosen call duration, the bot sends a BYE and closes the session. Authors of [16] argue that in VoIP the call holding times follow rather heavy-tailed distributions and suggest the generalized Pareto distribution for modeling them. We intend to adopt this distribution in the future.
– The social model: The bot belongs to one or more social cliques. The bot is socially connected with another bot if both belong to the same clique. P_{social} (e.g. 0.8) represents the probability that a bot calls a destination which is socially connected with. $1 - P_{social}$ (e.g. 0.2) is the probability that a bot calls a destination which it is not socially connected with. We provide a script (within the VoIPBot bundle) that assigns random social cliques to

a group of bots based on a given size interval (e.g. each clique has between 3 and 7 bots).
– The behavior upon receiving a call: Using statistics from traditional telephony, about 70% of calls are answered in roughly 8.5 seconds while unanswered calls ring for 38 seconds [17]. For instance, the bot takes the call and responds with BUSY only if it is already in-call. The time of ringing follows uniform distributions over pre-defined intervals (e.g. uniformly distributed from 5 to 10 seconds).

These parameters are configured through the IRC interface or at boot-time.

The generation of attacks is based on available VoIP assessment tools. The VoIPBot can also emulate a "malicious" behavior by committing attack actions commanded by the manager. The VoIPBot supports scan, spam a wav file as given by a URL, crack a password and register with, download and send exploits, fingerprinting VoIP devices, shoot crafted SIP messages and flooding.

Inviteflood, Sipscan and Spitter/asterisk[3] are used to generate flooding, scan and SPIT attacks. Inviteflood floods the target with INVITE messages. It doesn't acknowledge the target's response causing many retransmissions and memory overload. It allows to set the SIP URI of the destination and the rate of the attack, hence making possible different scenarios (e.g. invalid SIP user-name, invalid domain, etc.). Sipscan supports three types of scanning using REGISTER, OPTIONS and INVITE messages. It doesn't support the control of the scan intensity which makes its effect similar to a flooding. Spitter uses the Asterisk IP PBX as a platform to launch SPIT calls based on an Asterisk call file. The Asterisk call file format allows configuring calls towards different destinations with different parameters. It is possible to configure a different sender for each SPIT call which makes detection hard if based only on the caller's SIP URI.

5 Test-bed and labeled data-set

Physically, the test-bed is composed of four machines (Intel Pentium 4 CPU 3.40GHz and 2G RAM memory running Ubuntu 9.10) connected through a hub (10Mb/s). The machines are assigned static private IP addresses. A router plays the role of DNS server and Internet gateway for the test-bed. We set-up two illustrative VoIP networks based on available open source servers. The first network uses Asterisk: an open source VoIP PBX[4]. The second one is based on a SIP proxy called Opensips (former SIP Express Router).

5.1 Asterisk Network

Asterisk is growing fast typically for Small Office/Home Office environments (SOHO). Regarding the SIP terminology, Asterisk can be considered as a registrar server, location server, and back-to-back user agent. Our illustrative network

[3] `http://www.hackingvoip.com/sec_tools.html`
[4] `http://www.asterisk.org`

based on Asterisk is shown in Figure 2. We have two groups of VoIPBots emulating SIP users: the first group registers to Asterisk while the second group represents external users. Routing calls to and from external users is fixed at the Asterisk dial-plan. The first group connects to the #asterisk channel at the central IRC server and the second group connects to the #exterior channel. The manager issues commands to one group of bots by writing to the corresponding channel. It is possible to command each bot aside by opening a private channel with it.

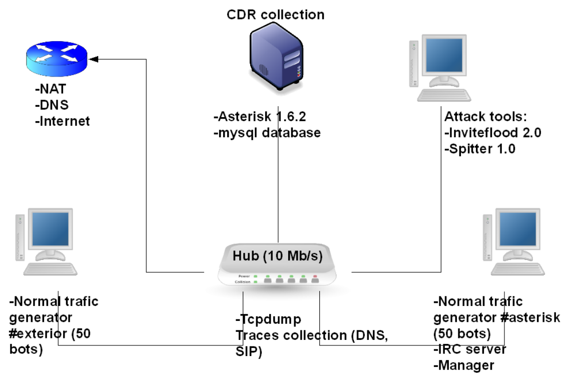


Fig. 2. Asterisk illustrative network

5.2 Opensips+MediaProxy+Radius Network

Opensips is a high performance SIP proxy since it is able to manage a high number of registered users and thousands of calls per second without the need of a highly sophisticated hardware. Opensips supports the proxy, registrar and redirect SIP functionalities. We configure a MySQL back-end to Opensips for consistent storage. MediaProxy[5] provides NAT traversal capabilities for media streams. It is composed of one media-dispatcher and one or more media-relays. The media-dispatcher runs on the same host as Opensips and communicates with its mediaproxy module through a Unix domain socket. The media-relay connects to the dispatcher using TLS. FreeRadius[6] works in tandem with Opensips and Mediaproxy to manage the call accounting.

We setup this triplet to serve two groups of bots: the first group is directly registered to the Opensips server and the second group represents the external

[5] `http://mediaproxy.ag-projects.com`
[6] `http://freeradius.org`

extensions. Routing calls to and from external users is fixed at the Opensips routing logic. The CDRs are collected using the CDRTool[7] which is an open source web application that provides accounting and tracing calls for Opensips. The CDRTool provides the SIP trace for each call based on the siptrace module of Opensips. It provides media information of each call based on the MediaProxy accounting.

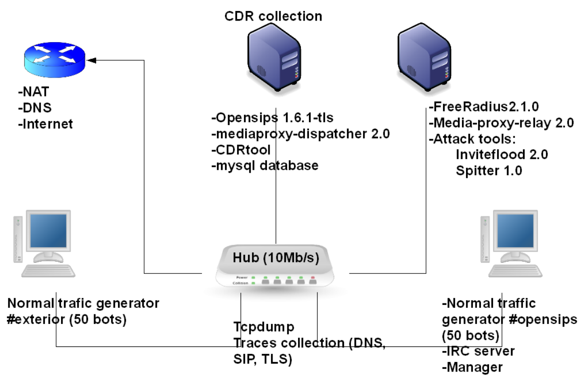


Fig. 3. Opensips illustrative network

5.3 Labeled Traces

We configure a set of normal and attack scenarios over the two aforementioned networks. The configuration choices are constrained by ensuring a relatively short time for each experiment (between 1 and 30 minutes) and a supportable overload for machines. In particular, we deploy a relatively low number of users (100 bots) and compensate this by a high number of calls per user and small call durations. The resulting traces of all experiments are available at `http://webloria.loria.fr/~nassar/voip-dataset/`. Each experiment is represented by a folder containing at least 4 files: a file containing the network trace in the tcpdump format, a text file representing the server log, a CSV file containing the CDRs and a meta-description file. The meta-description contains information about the experiment scenario (the number of bots, their repartition, the social model, the configuration of each group of bots, the network topology, etc.) and information about the attacks (if any) such as type, source and intensity.

The attack data are self-labeled in the sense that they carry a particular URI representing the attack tool. For example, in all the SIP messages and calls

[7] `http://cdrtool.ag-projects.com`

generated by Inviteflood and Spitter, we set the From-URI to start by "iflood" and "spitter" respectively. Table 1 shows a subset of the traces. The flooding intensity is measured by Invite/s while the SPIT intensity is measured by concurrent calls. A SPIT with no-limit intensity means that all the destinations are called in the same time.

Table 1. Labeled traces

Platform	$\lambda_{Poisson}$ (bot*hour)	Attack tool	Attack intensity	Trace duration	Number of calls
Asterisk	10	None	None	6 min	110
	10	Inviteflood	100 I/s	2 min	118
	10	Spitter	10 C.C.	2 min	82
Opensips	100	None	None	10 min	905
	100	Inviteflood	1000 I/s	2 min	1050
	100	Spitter	no-limit	2 min	452

6 Conclusion

In this paper, we have presented a unique experience of customizing and generating a labeled VoIP data-set in a controlled environment. We aimed by this contribution to make the recent works on VoIP attacks detection comparable. We provide SIP traffic traces, call detail records and server logs for a set of normal and attack scenarios.

In the future, our intention is to expand the test-bed by using virtualization techniques. We will extend the generation models in order to support more call statistical distributions, more attack types and more complex scenarios (e.g. with NAT traversal, inter-domain routing). The social model has to be refined based on the results of the social networks research. We also aim to evaluate the different proposed intrusion detection methodologies based on the data-set.

References

1. VoIPSA: VoIP security and privacy threat taxonomy. Public Release 1.0 (Oct 2005) `http://www.voipsa.org/Activities/VOIPSA_Threat_Taxonomy_0.1.pdf`.
2. Reynolds, B., Ghosal, D.: Secure IP telephony using multi-layered protection. In: Proceedings of The 10th Annual Network and Distributed System Security Symposium, San Diego, CA, USA (feb 2003)
3. Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., Schooler, E.: RFC3261: SIP: Session initiation protocol (2002)
4. Niccolini, S., Garroppo, R., Giordano, S., Risi, G., Ventura, S.: SIP intrusion detection and prevention: recommendations and prototype implementation. In: VoIP Management and Security, 2006. 1st IEEE Workshop on. (April 2006) 47–52

5. Wu, Y., Bagchi, S., Garg, S., Singh, N., Tsai, T.K.: SCIDIVE: A stateful and cross protocol intrusion detection architecture for Voice-over-IP environments. In: International Conference on Dependable Systems and Networks (DSN 2004), IEEE Computer Society (Jun 2004) 433–442
6. Sengar, H., Wijesekera, D., Wang, H., Jajodia, S.: VoIP intrusion detection through interacting protocol state machines. In: Proceedings of the 38th IEEE International Conference on Dependable Systems and Networks (DSN'2006), IEEE Computer Society (2006)
7. Fiedler, J., t. Kupka, Ehlert, S., Magedanz, T., Sisalem, D.: VoIP defender: Highly scalable SIP-based security architecture. In: Proceedings of the 1st international conference on Principles, systems and applications of IP telecommunications (IPT-Comm '07), New York, NY, USA, ACM (2007)
8. Chen, E.Y.: Detecting DoS attacks on SIP systems. In: Proceedings of 1st IEEE Workshop on VoIP Management and Security, San Diego, CA, USA (apr 2006) 53–58
9. Ehlert, S., Wang, C., Magedanz, T., Sisalem, D.: Specification-based denial-of-service detection for SIP Voice-over-IP networks. In: The Third International Conference on Internet Monitoring and Protection (ICIMP), Los Alamitos, CA, USA, IEEE Computer Society (2008) 59–66
10. Zhang, G., Ehlert, S., Magedanz, T., Sisalem, D.: Denial of service attack and prevention on SIP VoIP infrastructures using DNS flooding. In: Proceedings of the 1st international conference on Principles, systems and applications of IP telecommunications (IPTComm '07), New York, NY, USA, ACM (2007) 57–66
11. Sengar, H., Wang, H., Wijesekera, D., Jajodia, S.: Detecting VoIP floods using the Hellinger distance. IEEE Trans. Parallel Distrib. Syst. **19**(6) (2008) 794–805
12. Kang, H., Zhang, Z., Ranjan, S., Nucci, A.: SIP-based VoIP traffic behavior profiling and its applications. In: Proceedings of the 3rd annual ACM workshop on Mining network data (MineNet '07), New York, NY, USA, ACM (2007) 39–44
13. Nassar, M., State, R., Festor, O.: Monitoring SIP traffic using support vector machines. In: Proceedings of the 11th International Symposium on Recent Advances in Intrusion Detection (RAID '08), London, UK, Springer-Verlag (2008) 311–330
14. Nassar, M., State, R., Festor, O.: The VoIP Bot project `http://gforge.inria.fr/projects/voipbot/`.
15. Nassar, M., State, R., Festor, O.: VoIP malware: Attack tool & attack scenarios. In: Proceedings of the IEEE International Conference on Communications, Communication and Information Systems Security Symposium (ICC'09 CISS), IEEE (2009)
16. Dang, T.D., Sonkoly, B., Molnar, S.: Fractal analysis and modeling of voip traffic. In: In Proceedings of Networks 2004. (2004) 217–222
17. Duffy, F., Mercer, R.: A study of network performance and customer behavior during-direct-distance-dialing call attempts in the USA. Bell System Technical Journal **57**(1) (1978) 1–33