Security Test Environment for VoIP Research

Florian Fankhauser, Maximilian Ronniger, Christian Schanes, Thomas Grechenig
Vienna University of Technology
Industrial Software (INSO)
Austria

Abstract

Voice over IP (VoIP) is in wide use today, replacing phone lines in many scenarios. However, often, security isn't considered well enough, even though many security attacks are already known. More research on VoIP security is needed to enhance the level of security of VoIP systems and to show the implications of failing to take appropriate security measures. This paper presents a short introduction in testing VoIP components, proposes an architecture and implementation of a robust, flexible and efficient VoIP test environment for security related tests. Experiences using the implemented environment for different VoIP security tests are shown to demonstrate the suitability of the proposed test environment for research and teaching purposes.

1. Introduction

Voice over IP (VoIP) is in wide use in homes, educational institutions and businesses. It has gained a foothold as a supplement to Public Switched Telephone Network (PSTN) based phone lines or has replaced them completely in some scenarios. VoIP provides a way of sending phone calls over Internet Protocol (IP) based networks. This allows the use of one type of wiring for both computers and phones in office buildings, making management simpler and changes in the setup faster. However, if no additional hardening steps are taken to secure the VoIP traffic and the network infrastructure in those uniform networks, sufficient security of phone calls is usually not given. Additionally, security errors can also occur in VoIP phones themselves. Therefore, a way to apply security tests on VoIP equipment and infrastructures to assess possible vulnerabilities is needed to harden such systems and enhance the trust in VoIP calls

Various partly overlapping technologies have been introduced to cover different aspects of VoIP calls. Several signaling technologies (that take care of the setup of a voice channel) are in use today, e.g., H.323 or Session Initiation Protocol (SIP) as well as proprietary solutions, e.g., InterAsterisk eXchange (IAX) protocol. At the moment, SIP is probably the most widely adopted protocol [24] and, therefore, in focus of our proposed security test environment as the signalling technology for VoIP.

To secure a VoIP system, every component has to be taken into consideration, e.g., weaknesses in implementations of components that seem safe can cause problems in the whole infrastructure if multiple weaknesses are combined. Another example is shown by Zhang et al. [23]. In this example, a crafted SIP message in combination with a spoofed Domain Name System (DNS) response can cause a Man in the Middle (MitM) attack, thus wiretapping phone calls from a remote point of attack. As pointed out by Sicker [18] security has to be considered as early as in the design of a VoIP system, applying security after a system is installed is difficult and expensive to implement [6] and in some cases might be impossible.

This paper is based on our previous work [15] and presents in addition to requirements and a possible architecture of a flexible and robust VoIP test infrastructure which allows to run VoIP security tests for different VoIP test targets some advanced aspects like efficiently creating new test setups and using test results for further research. The evaluation of the test infrastructure was done by executing different security tests for various VoIP test targets like VoIP servers and user agents (UA).

Depending on the executed security tests, different requirements for the test environment have to be fulfilled. For example, to test a VoIP software for unknown buffer overflows, fuzzing test tools can be used. However, the use of such tools might lead to a crash of the tested software. As the attack vectors for fuzzing might contain automatically randomly manipulated and generated data, thorough logging mechanisms are required in order to automatically detect and reproduce the problem. Another security test might consist of a Denial of Service (DoS) (e.g., User Datagram Protocol (UDP) flood) attack of some kind, which would

require a setup with physical machines for maximum efficiency. Requirements such as these need to be considered when designing an environment for security tests.

The proposed test environment is built to be flexible enough to cover both diverse test cases while providing reliable tools (e.g., network traffic recording capabilities, revision control,...) to reproduce, record and analyze the robustness of VoIP components during attacks started by researchers. It is also designed in a scalable and flexible way so it can grow with demand while still allowing to setup and run new security tests easily.

The remainder of this paper is structured as follows. An overview of related work is given in Section 2. In Section 3 an introduction to testing VoIP components is given. Section 4 lists general requirements for a VoIP test environment, covering most aspects that need to be considered in the planning phase. Section 5 covers the architecture that is derived from the requirements and outlines the implemented setup used to run the tests, including a listing of components used. Moreover, some efficiency considerations are explained. Section 6 shows some benefits of using the proposed test environment and presents test results gained while using the test environment for VoIP security tests. The paper finishes with a conclusion in Section 7 and ideas for future work in Section 8.

2. Related Work

Several approaches to setup and operate a test environment for (security) tests exist (e.g., [4,21,22]), which mostly concentrate on functional aspects. In our approach we focus on security test specific aspects, e.g., robustness of the environment.

Cao et al. [4] describe the setup of a multifunctional network laboratory. They provided ideas for a structure of a lab network and also supported the idea of using virtual servers for running server software needed for the test environment. They focus on teaching hands on skills on network layers and infrastructure services, e.g., Dynamic Host Configuration Protocol (DHCP), but do not cover security aspects needed for security research.

Willems et al. [21] describe the implementation of a network lab with security focus, which offers remote access to students. While this paper focuses on providing a teaching environment for students, research requirements such as listed in Section 4 are missing.

Many attacks against VoIP infrastructures are known and described by different authors [2,3,12,23]. Butcher et al. [3] discuss various methods of securing VoIP infrastructures against attacks. A large number of attacks that are common today are listed. The attacks are categorized into groups according to which of the security aspects availability, confidentiality and/or integrity are compromised.

Blake [2] provides a good high level overview of the types of threats and attacks that VoIP Systems face today. The information on how to defend against certain types of attacks was used to specify security test cases for the test environment.

Zhang et al. [23] give a good example on how different weaknesses/attacks can be combined to achieve an overall goal, e.g., intercepting and/or modifying a VoIP conversation from a remote point of attack, by restarting a VoIP Analog Telephone Adapter (ATA) using a malformed SIP message and brute forcing DNS responses. This way, a SIP registration can effectively be rerouted to any server on the Internet. As a consequence, all services which are needed for the successful execution of a VoIP call have to be examined in security tests in a test environment. Even minor flaws can, in combination, cause a security issue and can allow to compromise the whole VoIP infrastructure. A security test infrastructure has to enable and support such testing scenarios.

Iossifov et al. [12] analyzed security aspects of the VoIP infrastructure of the University of Applied Sciences in Berlin. Their work shows methods of how to run penetration tests against a VoIP infrastructure and thus presents some test procedures in general. Some tests of Iossifov et al. were not executed due to missing requirements in the test infrastructure, e.g., a modification of VoIP hardphones via the web interface (while disconnected from the productive system) was not attempted because the hardphones required Power over Ethernet (PoE) to operate and a PoE switch was not available. This shows the need to have a flexible test environment to support test execution of different VoIP systems as they are used in real world scenarios.

While Herzog et al. [11] mainly focus on security tests on a client's premises, one part covers the conduction of security tests that can be applied to a VoIP infrastructure lab, e.g., router and DoS testing.

3. Testing VoIP Components

As seen in Section 2 many different security threats exist. For some examples of security attacks, see Section 4.1. As the usage of VoIP increases, the threat level of VoIP applications increases analogously. Therefore, security testing of VoIP infrastructures is needed.

A VoIP infrastructure consists of many different components. These include network equipment like routers or switches, VoIP servers and VoIP clients (UAs). Many implementations of such VoIP servers and UAs exist. There are softphones as well as hardphones. For many Operating System (OS)s softphones, software programs that are used for communicating over VoIP, are available. Hardphones are dedicated hardware phones with their own OS, which sometimes is Linux.

Softphones as well as hardphones consist of a user agent client (UAC) and UAS. This implies that both types of phones always need to have an open port, i.e., provide a server interface, that can have security errors and be exploited remotely. Therefore, a flexible security test environment has to provide mechanisms to support both types of UAs.

Sometimes, security tests have a high complexity and take a long time to complete. Parallelization should be used, therefore, to speed up test execution. This way the duration of a security test can be improved.

4. Requirements for a VoIP Test Environment

Requirements for test environments for different applications are discussed by different authors [1, 7, 14]. For environments to test voice applications some profound requirements are mentioned by Butcher et al. in [3] and Blake in [2]. The main focus of the proposed test environment is the use for security tests, where each test can bring its own set of requirements. The test environment will be used to test VoIP applications using simulated VoIP attacks. Attacks common today are listed in Section 4.1. At a minimum, these attacks have to be executable on their own as well as in combination in the test environment to simulate known attacks and test novel attacks found in the process of researching.

Section 4.2 lists general (security) test requirements. Some requirements are already known form software testing [1, 7, 14] which are integrated to support a wide range of various test cases and adapted to be flexible enough to be used for security tests. For the described infrastructure they are brought into the context of VoIP. Like in any test environment, the results of the security tests need to be reproducible with little effort as mentioned by Fewster [7]. Moreover, the test environment has to be flexible enough to easily add new hardware and software components and replace existing equipment.

4.1. Common Attack Vectors to VoIP Infrastructures

As shown by different authors (e.g., [2,3,5,9,13]), many attacks against VoIP are readily available and conducted in deployed VoIP infrastructures. Since one of the requirements for our VoIP test environment was to allow to simulate known attacks against VoIP, these attacks and possibilities to simulate them have to be analyzed.

VoIP related attacks for every network layer exist. The test environment design needs to reflect this fact by allowing security testers to execute attacks on every layer and analyze the behaviour of the system under test. This also includes VoIP application layer attacks such as attacks against the SIP stack or the used codecs.

Common threats against VoIP are brute forcing logins (possibly leading to toll fraud), Media Access Control (MAC)-Address spoofing or Address Resolution Protocol (ARP) poisoning (permitting to eavesdrop a VoIP call). As a consequence, this requires the test environment to support brute forcing tools and dialplan testers to check voice menus for possible weaknesses. Additionally, network technology is required that does not integrate counter measures against attacks on network level in order to research counter measures of VoIP components for those attacks.

Some of the less obvious attacks such as Virtual Local Area Network (VLAN) hopping or Trivial File Transfer Protocol (TFTP) spoofing can be used to simulate the circumvention of some security mechanisms. Since most devices in real world scenarios are capable of fetching configurations remotely (e.g., using TFTP or Hyper Text Transfer Protocol (HTTP)), the test environment should support security tests with manipulated configurations.

4.2. (Security) Test Requirements

When setting up a VoIP test environment for security tests, requirements from functional tests are known, e.g., reproducibility. Additionally, security test related requirements have to be considered like robustness of the environment during execution of attacks.

The core test environment has to be a solid and robust base, i.e., a crashed component should not affect other components in the test environment. This is especially important when automated tests are conducted. Additionally the test environment has to be separated from the productive network to avoid any side effects of tests [1].

The test environment has to be easily extensible in a modular way, i.e., the environment is required to provide an easy way to add new devices or replace existing ones to fulfill changing dependencies. This ensures that new test modules can be defined and added quickly, e.g., to replace a budget switch with one that is able to support separate VLANs and provides port level authentication as required in Butcher et al. [3].

In order to simulate a more complex VoIP environment a way to simulate multiple office locations (with links between them) has to be provided. These links may have less bandwidth (e.g., T1 or E1 link) when compared to the common 100Mbps office connection.

Tools capable of crafting invalid/modified data packets that can be sent to the System under Test (SUT) are required. Moreover, a well documented set of valid SIP/codec conversations is useful. This way, the valid traffic can be modified to create specific attacks, which is often easier than generating both invalid traffic and valid traffic for an

attack (see Section 4.1).

The tests have to be reproducible later to make sure no glitch caused false findings. To achieve reproducibility each test run has to be documented accurately, the test environment has to support this process by providing the necessary tools to capture and archive them. As mentioned by Black [1] documentation of every change in the test environment is required, e.g., newly installed software.

The test environment should allow access to logged information during execution (e.g., used configurations, log files or network dumps) for further analysis. Logging capabilities should be automated as much as possible – preserving the used configuration files, network capture(s) and relevant log files of any test run. Events in the log files have to be easily matched, to allow for simple analysis when reviewed later. This is important to be able to analyze and possibly debug it later as required in [7]. For detailed analysis of security tests captured network traffic is desirable. The test environment should be flexible to allow different locations to capture the network traffic.

The infrastructure has to offer a basic framework of services (as described in Section 5.2) and should support security tests of hardphones and softphones.

Moreover, it should be possible to conduct different test scenarios. These include, e.g., testing a single VoIP server, a single VoIP client as well as testing complete VoIP infrastructures, e.g., the behavior of a VoIP infrastructure when malicious code (e.g., a worm) is executed within the infrastructure.

5. Architecture for a Robust VoIP Test Environment

The architecture introduced in this section is crucial in fulfilling the elicited requirements from Section 4. In Section 5.1 an overview of the architecture used is given, while Section 5.2 describes the components used to implement the test environment.

5.1. Architecture Details

The architecture for the security test environment considers classical telecommunication issues. One of the first is the dimension of the environment which directly affects the amount of resources needed (e.g., IP addresses or phone numbers) for assignment. A dialplan is a mapping between a number and a "service" that is offered at that number. Service can refer to an extension - a physical line or registered VoIP phone, a Voice User Interface (VUI), a Voice Mail System (VMS) or just a playback of a recording (e.g., Time Service). In real world applications, numbers of any length (usually ranging from 1 to 7 digits) are used. For practical reasons all valid numbers should have the same length. For the test environment a length of 4 was chosen because it is short enough to remember, long enough to incorporate space for future expansion and doesn't collide with emergency numbers (length of 3 digits). When simulating multiple locations or branch offices, a portion of the total number pool has to be assigned to each location, while still keeping blocks of free numbers for later assignment.

Similar to the phone numbers, the network layer has to offer enough address space to accommodate the current and future network devices. The allocated IP addresses were chosen in a way that would permit to add more addresses later on without reassigning the IP addresses and adding new gateways to the new IP addresses for the virtual machines.

A Private Branch Exchange (PBX) software was integrated to implement more complex features, commonly used in small to big offices (e.g., hold call or call forwarding) and to provide many features needed by telecoms. Asterisk was chosen because it is one of the most widespread open source solutions. It allows the implementation of most of the features known from traditional telephone systems in software. It can handle all scenarios from a simple call between two endpoints to more complex scenarios like a group call, a simple voice menu (VUI), an echo test, group ringing and a conference call.

To make the test environment more flexible, most services (PBX, DHCP, etc.) are run on virtual machines, so their state can be saved at any time and restored easily later in case the service crashes due to a security test. This also allows to save the current version of the test environment for reproducibility issues. As each service runs on its own (virtual) server, a crash of one service does not compulsorily affect other services directly. In order to increase availability even more, high availability mechanisms provided by different OS could be used in future versions.

The first and foremost network consideration for a security test environment is to make sure the test environment is safely separated from the production/office network and the Internet. That way any simulated attack safely stays within the test environment without any effects on a 3rd party, even if a test tool was configured incorrectly and the attack would target a machine on the Internet. To achieve this, the network has no direct access to the outside network. Only access from the outside into the test network (for management access) is permitted.

To make the test environment scalable and manageable a centralized configuration repository, which is also used in larger operational environments, was implemented. Most hardphones support this by allowing to fetch the configuration data from a remote location (TFTP or HTTP). For the evaluation one configuration file per hardphone and test scenario was used. For reproducibility the files were added to the integrated revision control system.

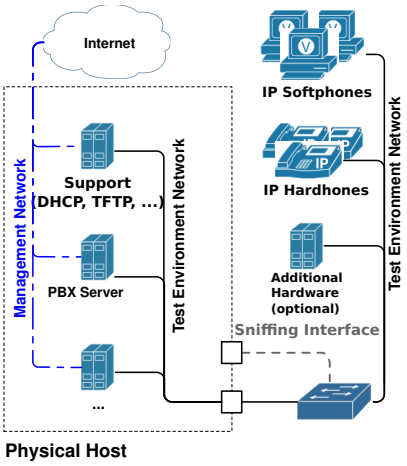

Figure 1. Test environment setup

To enable an easy analysis of security incidents, log files produced by different software need to be correlated. A central time source in the described environment provides common time for all used components. This allows to correlate log information from different hosts and log sources for the analysis.

5.2. Implementation of the Test Environment

The proposed test environment was implemented by using the following hardware components: A server machine with an Intel compatible CPU and 4 GB of RAM, 1 managed Gbit switch and 3 different state-of-the-art hardphones. An overview of this setup is given in Figure 1. For the proof of concept environment and the executed tests this hardware provides enough resources and the environment allows to be easily extended with additional resources when required.

The server machine had 3 Ethernet Network Interface Card (NIC)s which provided the following functionality: first, a connection to the test environment switch where the hardphones and optionally available additional hardware were connected, the second NIC for recording the traffic, connected to a port of the switch that was set up to mirror all traffic and the third interface was connected to a management network for remote management access.

The switch is connected to the virtual environments by a 100Mbps NIC which was sufficient for up to 500 concurrent calls in G711 a-law codec (assuming 100Kbps per call and direction, when the IP and SIP signaling overhead is taken into account) to take place. If more calls need to be simulated, the use of a 1Gbps NIC or multiple NICs utilizing Link aggregation is suggested. In contrast, if a link with less bandwidth should be simulated, in a test environment, this can be done easily by applying traffic shaping measures. The whole communication can be intercepted by connecting a computer with packet capture software to the mirror port of the managed switch.

The test environment provides the researcher with documentation of the setup as well as some basic configuration sets to use as a base for tests. All the templates, log files and network dumps are kept in a version control system so changes can be tracked, and previous configurations can be reactivated. Changes in the version control system are activated on a reboot (a startup script was setup to update the configuration files relevant for each machine).

The host machine provided remote management forwarding to the virtual machines as well as a temporary transparent HTTP-proxy (set up to only allow specific domains) to allow quick software installation/updates on the virtual machines while blocking all undesired outgoing traffic to the office network/Internet.

The virtual machines (as utilised by other test environments, e.g., [4]) provided multiple infrastructure services (Figure 1):

For example, a support machine was used to provide DHCP for dynamic client configuration, TFTP and HTTP to serve configuration files to the hardphones, and Network Time Protocol (NTP) to provide one synchronous time source to all the clients.

The Asterisk machine provided a small PBX. The hardphones and softphones registered at the PBX using SIP.

Two configuration flavors of Asterisk were setup. For the first one, used to test Asterisk itself, no hardening steps were taken, as this reflects many common installations. For the second configuration basic hardening was applied by removing unnecessary modules (asterisk.conf) and codecs (sip.conf). This instance was used when the hardphones were tested.

An extension with a sound file playback was integrated in the dialplan to make debugging easier, simulate server load and to check if the testing environment was working correctly. It also offered a way to subjectively test the Quality of Service (QoS) (e.g., sound quality during DoS attacks).

5.3. Efficiency Considerations of a VoIP Test Environment

Some of the test cases, especially the ones including fuzzing/brute force techniques, often take a long time to complete. This is caused by the high number of test cases. In these cases running the tests in parallel on multiple test instances is an efficient way to improve the overall runtime of a specific test set.

In order to parallelize the test, multiple instances of the same software (setup) are used. Using cloned images with the same base configuration can easily provide these in-

```
host server-23 {
  hardware ethernet 00:B0:CF:8B:49:23;
  fixed-address 192.0.0.23;
  [...]
}
```

Figure 2. dhcpd.conf extract

stances. In order to use the parallelization test cases have to be split up into subsets which then can be run against an instance of the SUT.

For faster creation of these multiple instances a cloning process was used. A base image was created by cloning a vanilla (i.e., unmodified) OS image using the available virtualization tools, installing the needed additional software and applying a configuration independent on the specific instance of the image to the software (e.g., don't bind the software to a specific IP address). Afterwards, this image was cloned n times, using the virtualization management tools to modify the images to make them unique (e.g., MAC address,...). The available tools of the virtualization software used allowed to customize a small number of hardware characteristics only for each new instance created. This functionality was, therefore, mainly used to change the MAC address of the specific image.

When an instance is started as much configuration information as possible is supplied using DHCP. If, for some reason, DHCP cannot be used to modify a configuration parameter, direct file modification will be applied.

When storage space for the instance images is limited, the instances can share the base disk image amongst all of them (using the immutable feature). Saving $(n-1) * imagesize$ storage for a minor memory penalty. In this case the differences to the base file system will be kept in RAM. However, the configuration of each instance has to happen at every instance start using this method. In order to manage the configurations and log files efficiently, a revision control system was used for permanent storage of the relevant files. A version for each instance will be checked out of the revision control system during the boot procedure.

DHCP was used to supply configuration data dynamically to each used instance. To ensure identical network configurations between multiple test runs (which is, e.g., important for retests) the options supplied to the instances were bound to the MAC address of an instance, which is automatically changed during the creation process of a new image and is unique in the test environment. An excerpt of an example of a client specific DHCP configuration file is given in Figure 2.

When the needed configuration options cannot be passed by DHCP options to the specific image, e.g., SIP user authentication/registration information, direct modification of configuration files in combination with the revision control system was used. The disk image of each instance was directly accessed from the host machine and the files altered to reflect the instance's requirements. After this initial setup, the relevant configuration files were checked into the revision control system, from where they were restored at each boot of the instance.

6. Using the Test Environment

For evaluation of the implemented proof of concept test environment, multiple security tests were accomplished, the test environment proved to be flexible and reliable enough to conduct different kinds of security attacks.

At first, we started to capture network traffic on the sniffing interface (Figure 1) and booted several hardphones. We found that most of the hardphones try to access multiple NTP servers via Internet and the manufacturer's websites on their first boot, i.e., with their default configuration settings. This information could be used to attack the hardphones, e.g., by sending wrong NTP responses. Moreover, if DNS records get tampered, an attacker could try to find out how many hardphones get started and at which times. If known vulnerabilities for different hardphones exist, this information might further be used to exploit hardphones with their default configuration.

Another opportunity for which the sniffing interface is used, is finding out about configuration or software updates of hardphones. By sniffing the packets the hardphones are sending while booting, TFTP access can be found out. Access to TFTP servers could be attacked afterwards to upload, e.g., wrong configuration files or faked software updates to hardphones. By using the provided TFTP server within the test environment as described in Section 5.2, various attacks could easily be brought to completion.

Endler and Collier showed many security attacks against VoIP [5]. In the test environment it was easily possible to attack various hard- and softphones as well as PBX servers with attacks and tools shown by Endler and Collier, e.g., a UDP flood attack. The various test cases got executed and the automatic network dump was utilized in order to be able to analyze the security attacks later. The test environment provided enough resources so that the VoIP component received the whole flood of packets and none where throttled by bottlenecks in the testing environment.

Fuzzing is a widely used dynamic test technique to detect security vulnerabilities of different software [8, 10]. We conducted fuzzing tests in the test environment [20] and found different security vulnerabilities of various softphones (e.g., [16, 17]). The test environment allowed the testers to focus on the setup of the fuzzing software because the needed infrastructure for the tests was in place and only little infrastructure setup was required. The network sniff-

ing was easily enabled and helped to analyze different reactions of the softphones to various attack strings.

The test environment was also flexible and robust enough so we could conduct DoS attacks and load tests against various PBX servers. A physical machine was added to the lab environment to start different DoS attacks (e.g., UDP Flood, INVITE Flood, REGISTER Flood,...). The load of the PBX servers was measured with various metrics at the PBX servers themselves as well as via the network dump at the sniffer interface [19].

6.1. Additional Benefits of the Security Test Environment for VoIP

Using the proposed VoIP test environment has several benefits.

First, the VoIP test environment can be used by researchers for analyzing security errors in different components of a VoIP infrastructure (see Section 3).

Second, as all the traffic in the VoIP test environment is stored for later analysis, it can be further used as a basis for additional security test cases, i.e., for creating more security attacks (see Section 4.2).

Moreover, the saved network dumps and configurations can be used in teaching students VoIP security. The behavior of different attack vectors can easily be shown and different exercises can be assigned.

6.2. Analyzing Test Results

Analyzing test results is one of the most important and often most time-consuming steps when conducting automated security tests. Multiple methods to improve the analysis of test results are available within the proposed VoIP test environment that can be used in addition to methods that are available from the security test tools used.

Basic analyzer scripts have been written with `tshark` which is a command line application that can be used for automatically extracting information out of network dumps. With these scripts it is, e.g., possible to find timing problems or check if a SIP communication was executed correctly. For different use cases these scripts have to be adapted.

7. Conclusion

VoIP is in wide use today in various productive use cases. Common to most of these use cases is that the security of the VoIP systems is business critical. Although many security attacks are already known, VoIP systems are still vulnerable to many different attacks as was shown in this paper. Therefore, more research has to be performed to further raise the level of security of VoIP telephony.

The test environment proposed in this paper supports the research of various attacks against VoIP systems. These attacks include various known attacks, e.g., DoS or spoofing attacks. Several executed test cases showed that the test environment is flexible and robust enough to conduct different kinds of known and unknown security attacks and thereby enhancing the level of security of VoIP systems. It was shown how the test environment can be set up efficiently so, e.g., parallelization of security tests is supported.

Some benefits of using the proposed test environment were shown. These include collecting data for enhancing further attacks and using the collected data for teaching.

8. Future Work

Further work is needed to simulate more complex setups, e.g., trunked VoIP connections between remote PBX systems or failures of, respectively, lossy connections. The analysis of test case results should get even more automated. Continued security tests and accompanying extensions to the test environment are planned. The security test environment should be extended so multiple, different security test cases can be executed in parallel without interference of the different tests.

9. References

[1] R. Black. *Managing the Testing Process: Practical Tools and Techniques for Managing Hardware and Software Testing*. John Wiley & Sons, Inc., New York, NY, USA, 2002.

[2] E. A. Blake. Netwok security: VoIP security on data network – a guide. In *InfoSecCD '07: Proceedings of the 4th annual conference on Information security curriculum development*, pages 1–7, New York, NY, USA, 2007. ACM.

[3] D. Butcher, L. Xiangyang, and G. Jinhua. Security challenge and defense in VoIP infrastructures. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 37(6):1152–1162, Nov. 2007.

[4] X. Cao, Y. Wang, A. Caciula, and Y. Wang. Developing a multifunctional network laboratory for teaching and research. In *SIGITE '09: Proceedings of the 10th ACM conference on SIG-information technology education*, pages 155–160, New York, NY, USA, 2009. ACM.

[5] D. Endler and M. Collier. *Hacking Exposed VoIP: Voice Over IP Security Secrets & Solutions*. McGraw-Hill, Inc., New York, NY, USA, 2007.

[6] S. R. Faulk. Software requirements: A tutorial. Technical report, Center for High Assurance Computer Systems, US Navy, 1995.

[7] M. Fewster and D. Graham. *Software test automation: effective use of test execution tools*. ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 1999.

[8] V. Ganesh, T. Leek, and M. Rinard. Taint-based directed whitebox fuzzing. pages 474–484, May 2009.

[9] M. Garuba, L. Jiang, and Y. Zhenqiang. Security in the new era of telecommunication: Threats, risks and controls of VoIP. In *Information Technology: New Generations, 2008. ITNG 2008. Fifth International Conference on*, pages 587–591, April 2008.

[10] P. Godefroid, A. Kiezun, and M. Y. Levin. Grammar-based whitebox fuzzing. In *PLDI '08: Proceedings of the 2008 ACM SIGPLAN conference on Programming language design and implementation*, pages 206–215, New York, NY, USA, 2008. ACM.

[11] P. Herzog, R. Lee, R. Tucker, N. Hedges, C. Clark, A. Barisani, M. Ivaldi, R. Chiesa, K. Supporters, D. Lavigne, F. Schallock, A. Chuvakin, E. Torres, L. Vera, R. M. Azorin, R. Feist, R. J. Meijer, J. Pascuzzi, and C. Griffin. OSSTMM 2.6.- the open source security testing methodology manual, 2006.

[12] V. Iossifov, T. Totev, and A. Tochatschek. Experiences in VoIP telephone network security policy at the University of Applied Sciences (FHTW) Berlin. In *CompSysTech '07: Proceedings of the 2007 international conference on Computer systems and technologies*, pages 1–18, New York, NY, USA, 2007. ACM.

[13] G. Me and D. Verdone. An overview of some techniques to exploit VoIP over WLAN. In *Digital Telecommunications, , 2006. ICDT '06. International Conference on*, pages 67–67, Aug. 2006.

[14] G. J. Myers and C. Sandler. *The Art of Software Testing*. John Wiley & Sons, 2004.

[15] M. Ronniger, F. Fankhauser, C. Schanes, and T. Grechenig. A robust and flexible test environment for voip security tests. In *Internet Technology and Secured Transactions (ICITST), 2010 International Conference for*, pages 1–6, Nov. 2010.

[16] C. Schanes. ESSE - security advisory 201004-0001, http://security.inso. tuwien.ac.at/advisories-html/20100406-SIP-Communicator-DoS-Port-Limit. html, Apr. 2010.

[17] C. Schanes. ESSE - security advisory 201004-0002. http://security.inso. tuwien.ac.at/advisories-html/ 20100406-SIP-Communicator-XSS.html, Apr. 2010.

[18] D. C. Sicker and T. Lookabaugh. VoIP security: Not an afterthought. *Queue*, 2(6):56–64, 2004.

[19] P. Steinbacher, F. Fankhauser, C. Schanes, and T. Grechenig. Work in progress: Black-Box approach for testing quality of service in case of security incidents on the example of a SIP-based VoIP service. In *Principles, Systems and Applications of IP Telecommunications (IPTComm'10)*, pages 107–116, Munich, Germany, Aug. 2010. Technische Universität München.

[20] S. Taber, C. Schanes, C. Hlauschek, F. Fankhauser, and T. Grechenig. Automated security test approach for SIP-based VoIP softphones. In *The Second International Conference on Advances in System Testing and Validation Lifecycle, August 2010, Nice, France*. IEEE Computer Society Press, Aug. 2010.

[21] C. Willems, W. Dawoud, T. Klingbeil, and C. Meinel. Security in tele-lab – protecting an online virtual lab for security training. In *Internet Technology and Secured Transactions, 2009. ICITST 2009. International Conference for*, pages 1 –7, 9-12 2009.

[22] D. Yuan and J. Zhong. An instructional design of open source networking laboratory and curriculum. In *SIGITE '09: Proceedings of the 10th ACM conference on SIG-information technology education*, pages 37–42, New York, NY, USA, 2009. ACM.

[23] R. Zhang, X. Wang, R. Farley, X. Yang, and X. Jiang. On the feasibility of launching the man-in-the-middle attacks on VoIP from remote attackers. In *ASIACCS '09: Proceedings of the 4th International Symposium on Information, Computer, and Communications Security*, pages 61–69, New York, NY, USA, 2009. ACM.

[24] T. Zourzouvillys and E. Rescorla. An introduction to standards-based VoIP: SIP, RTP, and friends. *Internet Computing, IEEE*, 14(2):69 –73, 2010.

