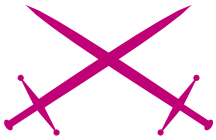

Storming SIP Security

Sandro Gauci

Difficulty

VoIP is a hot and steadily gaining market share in the phone business. As people constantly seek to make long distance calls cheaper, they are moving away from relying on the traditional telephone companies and heading more towards Voice over IP (VoIP). Phone calls between two VoIP users are usually free and do not carry any additional costs, other than that of the Internet connection and possibly the bandwidth are sed.

Connecting to *Public Switched Telephone Network* (PSTN) phone numbers from VoIP will usually carry a fee however that typically gets paid by the VoIP user. In fact, a lot of Internet Service Providers around the globe are now advertising VoIP as part of their services on offer. Apart from this, corporations are moving away from the traditional PBX systems to an IP based phone system. While there are a number of proprietary and non-proprietary protocols in existence, SIP looks like the one that is emerging as the standard. Many of the VoIP phones currently deployed now support the SIP protocol even if they might not fully implement it.

As SIP starts to make a difference in the ways we communicate, it will become yet another target for malicious attackers looking to make a quick buck, or maybe just have some fun at the expense of others. As security professionals and system administrators who might deploy a VoIP system relient on SIP, we have the responsibility of understanding what security challenges exist, in order to be able to fix or avoid these issues.

In this article, we shall be describing attacks that can be used to compromise VoIP systems which use the SIP protocol, and protocols that rely on it. Although we do not present any new attacks, all of the described methods can be very effective offensive tools for a malicious user, and make use of freely available software. We will be describing attacks that target:

- Information gathering: identifying SIP devices on the network and extensions on a Private Branch Exchange (PBX)
- Availability issues: denial of service on the phone system
- Toll fraud and identity theft: stolen accounts

What you will learn...

- Why IP phone systems are the new target
- In depth examples of attacks on IP phone systems
- How to mitigate security issues related to SIP-based phones

What you should know...

- Basics of security terms such as denial of service and availability
- Basics of networking as the UDP and VPN

Important Note:
All code listings for this article can be found on the CD attached to this magazine.

The attack coverage is not extensive but we will attempt to describe the attacks in detail, rather than just give a brief introduction to each attack. In this article, we will not be going into confidentiality or integrity issues to do with SIP. This means that we will not be talking about phone taps or man in the middle attacks, which are already thoroughly discussed in other articles, books and the popular media. We will, instead, be covering attacks that can be launched remotely over the Internet, without having access to the *Local Area Network*. Throughout this article, we will be making use of traces of SIP packets to easily illustrate how the protocol works, and how SIP network entities behave. These packet dumps can be easily reproduced by making use of Wireshark and tcpdump, both of which are network protocol analyzers. More importantly, we will give multiple layers of security solutions to counter these security concerns.

The reader is expected to be familiar with basic security concepts such as denial of service and availability, as well as technologies such as the *Virtual Private Network* (VPN) or challenge response mechanisms. On the other hand, the reader does not need to have experience with the *Session Initiation Protocol* (SIP), since we will start by introducing SIP and describe how it makes up a Voice over IP system.

What You Need to Know About SIP

As with many other protocols, the *Session Initiation Protocol* is defined in a *Request for Comments* (RFC) document, developed and designed within the *Internet Engineering Task Force* (IETF). The main RFC that defines SIP is RFC 3261, which is 269 pages long and takes a lot of variables into consideration. In this section, we shall be giving a basic introduction to SIP, so that if you are not familiar with the protocol, then you will have enough knowledge to follow the rest of the article. If you are already familiar with the protocol, you might wish to skip the next section and go straight to the attacks section.

The *Session Initiation Protocol* (SIP) is an application layer protocol that takes care of connecting two or more participants via a session. SIP also takes care of any modifications to this session and session termination. Since it is independent of the transport layer, SIP can make use of UDP and TCP (usually on port 5060), as well as TLS over TCP (typically on port 5061). SIP is not limited to just telephone calls, but can also be used for multimedia distribution, multimedia conferences, instant messaging and online games. While SIP is often taken as synonymous with VoIP, the protocol itself does not handle everything that has to do with VoIP. For the delivery of voice, SIP relies on other protocols such as the *Real-time Transport Protocol* (RTP), and the *Session Description Protocol* (SDP) for initializing the RTP stream. The job of SIP is to act as an intermediary protocol to help two

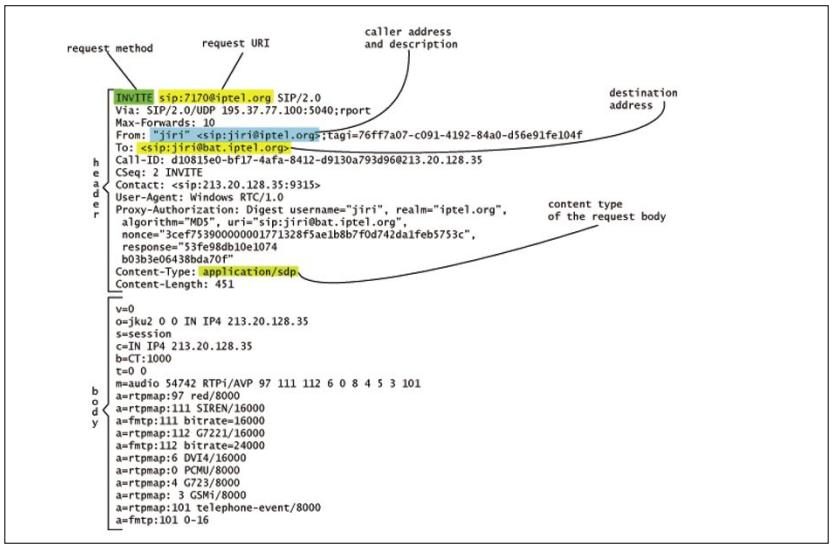

Figure 1. *An INVITE message*

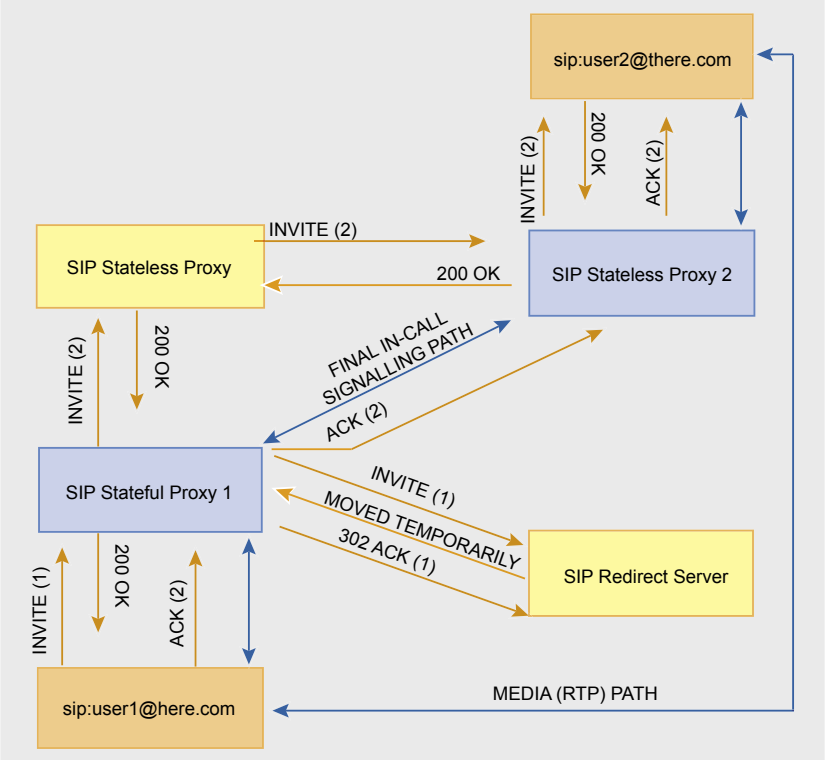

Figure 2. *A high level view of various SIP network elements*

devices to find a way to contact each other and establish a session. While the traditional PSTN (*Public Switched Telephone Network*) relied heavily on the network, SIP depends more and more on the end device, which does its own state and logic book keeping. This means that SIP shifts the intelligence from the network to the end devices (telephones).

A SIP network element is called a User Agent or UA. User agents can be clients or servers; SIP Phones are typically User Agent Clients, while a Proxy or Registrar is conventionally called a User Agent Server (UAS). Three types of UAS are SIP Proxies, *Registrars* and *Redirect Servers*. Proxies can be Stateful and Stateless. The difference between a Stateful proxy and a Stateless one is that the Stateless just forwards SIP messages, while the Stateful Proxy creates a state and keeps it until the session finishes. A Registrar is a UAS that can handle and process a REGISTER request. Such a Server keeps a database of addresses mapped to a specific User Agent Client or Clients. It is important to note that the difference between different SIP servers is logical and not physical, which means that a server can be both a proxy and a registrar. For example, Asterisk PBX will act as both a Registrar and a Stateful Proxy Server.

What Does SIP Look Like?

SIP resembles HTTP. Similar to HTTP, it has a header and a body, consists of printable characters (not a binary protocol) and supports various methods. While in HTTP we are used to GET and `POST` requests, SIP supports a number of methods such as `INVITE`, `REGISTER`, `OPTIONS`, `BYE` and `CANCEL`. If you are familiar with protocols related to email, you will also at least notice that SIP borrows the *To* and *From* headers from SMTP and Message format.

Figure 1 shows an INVITE request which is typically used to establish a session. The request is coming from *Jiri sip:jiri@iptel.org* and destined to *sip:jiri@bat.iptel.org*, and contains an SDP body. On a higher level, a phone call via SIP will probably look like the Figure 2.

There are various SIP network elements in the diagram. User1 is a SIP phone trying to call user2 and both phones are behind a *Stateful Proxy*. The RTP media (which carries the voice data) in the diagram is established directly between `user1` and `user2`. In reality, if user1 and `user2` are behind a NAT (*Network Address Translation*), then both user-agents will not be able to stream RTP directly. For that, many SIP phones nowadays support a protocol called Simple Traversal of UDP over NAT or STUN. This protocol allows both SIP phones to punch holes in a NAT to allow both devices to contact each other directly.

A Typical Phone Call

When a SIP phone calls another phone, it starts by sending an INVITE message to the proxy which usually contains an SDP body. The proxy will then send back a *100 Trying* response, which means that the proxy has received the message and is trying to contact the destination. Once the proxy has managed to route the `INVITE` request to the destination, it sends the user agent a *180 Ringing* or *183 Session Progress* response. The phone at the other end starts ringing as soon as it receives the original INVITE. If the receiver of the call picks up the phone, it sends a *200 OK* response to the proxy, which is then relayed to the originator of the call. This response typically contains an SDP body which allows the phones to negotiate the codec used for RTP and other variables. The SIP phone that originated the call then confirms the receipt of the OK with an ACK request. At this point both phones start the voice stream and communicate via RTP. When one of the parties decides to hangup, the phone sends a BYE request which should be responded with a *200 OK* message. Figure 3 illustrates this typical situation.

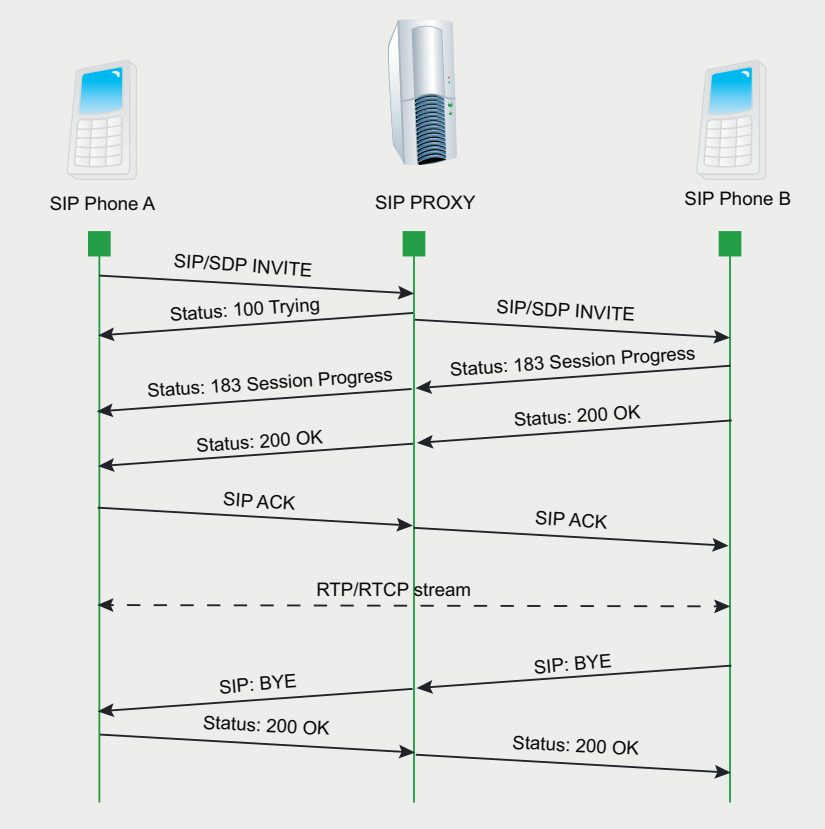

Figure 3. *Phone A rings Phone B through a SIP proxy*

Authentication For SIP Phones

To receive phone calls, a SIP phone needs to tell a SIP User Agent Server that it is ready to receive the

phone calls that are destined to a given extension. This is achieved by sending a REGISTER request to a registrar server. Although not required, REGISTER messages are usually authenticated. As presented in Figure 4, the first REGISTER message sent by the user agent to the registrar does not contain any credentials. As a response, if authentication is required, the registrar then sends back a *401 WWW-Authenticate* message, which contains an authentication challenge. The SIP phone computes the challenge response and sends a second REGISTER request which contains the authorization header and the challenge response. If the challenge response is the same as the one expected by the registrar, then the registrar sends a *200 OK* response indicating that the user agent has been authenticated. From now on, any calls destined to the registered SIP address will be routed to the authenticated User Agent. For authentication, SIP typically relies on digest authentication, which makes use of md5 hashing algorithm.

Other SIP Messages

Other methods of interest are:

- OPTIONS is used to query the user agents for their capabilities
- CANCEL is used to cancel a previous request issued by the client
- BYE is used to terminate a dialog/media session initiated by an INVITE
- ACK is used to acknowledge final responses to INVITE requests

Attacking SIP Devices

Identifying Valid Extensions on a PBX. An attacker targeting a phone system will first need to identify the IP address and port of the PBX. There are various tools which can do this, including typical port scanners such as nmap. One may also make use of tools dedicated to SIP such as smap – which is described as a mixture of nmap and sipsak – and svmap, which is part of the SIPVicious tool suite (written by yours truly). Once the PBX server has been identified on the network, an attacker can attempt to find out which extensions can be registered on the PBX. Knowledge of these valid extensions will be useful for an attacker attempting to strategically exploit the phone system further. Traditionally, phone *Phreaks* (phone system hackers) made use of war dialing, which is the act of calling each possible number or extension on a phone system in an attempt to identify interesting devices behind that number. With SIP, this is not required since most of the times there are more efficient methods which allow you to achieve the same results.

To identify an extension, the attacker needs to differentiate between an existing one and a non-existent extension. By existing extension we mean an extension that can be registered. Following some research, we found that the best method to identify existing extensions is to record the response for a request to a non-existent extension, and then look out for requests that produce a different SIP response code.

This method was implemented in a security tool called SIPScan (part of the *Hacking VoIP* book) and later also by svwar, which is part of the SIPVicious tool suite. Let us look at how svwar works. Initially we shall be targeting an Asterisk box on a preconFigured test VM (*Virtual Machine*) running Trixbox, which is an easy to use Linux distribution that comes with Asterisk installed. We made use of the web interface on Trixbox to conFigure Asterisk with four SIP extensions (can be seen in Figure 5). Thus, for this test we created a few extensions which will show up later on during the scan. Since we are running a default scan, svwar will be scanning a range of extensions between 100 and 999 and making use of the REGISTER SIP method. As you can see in Listing 1, svwar identified the four extensions on the PBX. These extensions are conFigured to allow registration of SIP phones when supplied with the right credentials.

In the background, svwar first sent a request to a non-existing extension on the PBX and recorded the SIP message response. Based on this test, svwar learns that when an unknown extension is getting registered, Asterisk will reply with a *SIP/2.0 404 Not found* message, as seen in Listing 2. This same response is seen when

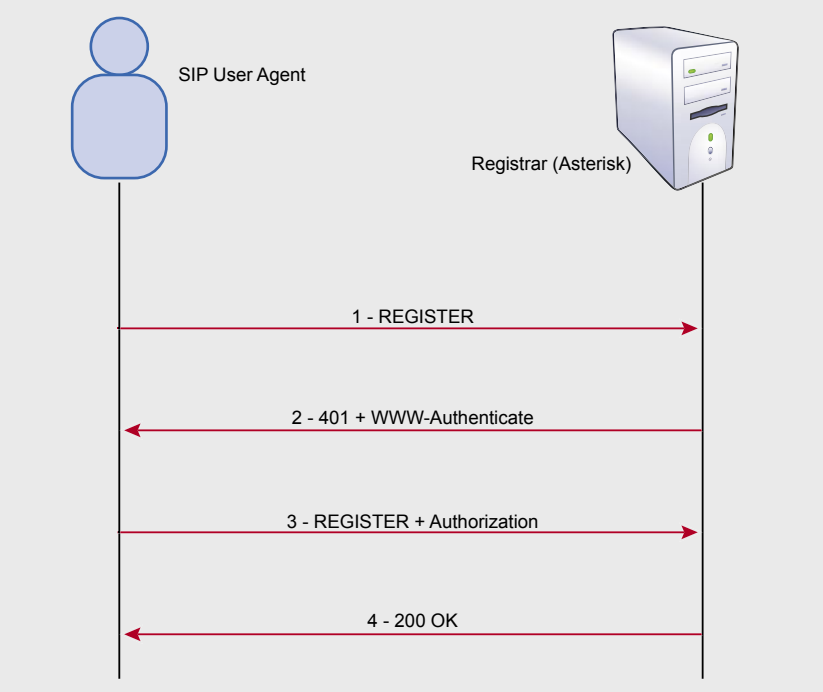

Figure 4. *Authentication with a SIP registrar*

svwar then starts to send REGISTER requests for the extension range. Listing 3 shows a SIP message response which is different from the responses generated previously. For svwar this indicates a valid working extension. While this works for Asterisk, many PBX servers out there exhibit different behaviors. Let us look at Java based PBX named Brekeke. Listing 4 shows a register request sent to a non-existent extension, while Listing 5 shows one that was sent to an existing extension. As you will notice, the responses for both requests emit the same kind of response, which is a *403 Forbidden* message. Svwar version 0.2.1 will detect this and will inform the end user about the problem (see Listing 6).

However, there are other methods that can be used to detect existing extensions on Brekeke PBX and other servers. By making use of the OPTIONS request (Listing 9) instead of a REGISTER, we are able to emit a different response. In the case of an OPTIONS SIP request, Brekeke acts as a proxy and sends the OPTIONS request to the SIP phone, which happens to be an X-lite client. This means that the extension is currently being used by a softphone. If we were to try the same request on an Asterisk, we would notice a very different behavior. Asterisk always replies with a *200 OK* and unlike Brekeke, does not forward these requests to the registered SIP Phone. When neither the REGISTER method nor the OPTIONS method work, one can try other valid methods such as INVITE. It is also possible to make use of invalid request methods (see Listing 10), which might give out some interesting results.

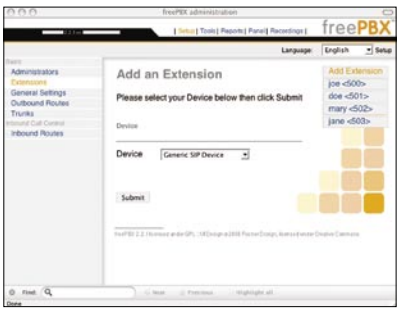

Figure 5. *Trixbox configuration*

Making Direct Phone Calls and Causing Havoc

Various network elements are involved in a phone call which uses the SIP protocol. A SIP phone will typically be registered with a VoIP provider or PBX, and to call another SIP phone it will need to find out the location of the destination by asking a SIP proxy. This system has the benefit of allowing the systems administrators to centrally manage the voice infrastructure, enabling the possibility of preventing abuse. For example, VoIP spam (better known as SPIT – Spam over Internet Telephony) can be controlled better by disallowing unauthenticated calls coming from the Internet.

Let us illustrate this behavior by looking at a softphone which is making a call through a fictitious VoIP provider (called Sip Provider) that the user does not have access to. Listing 11 shows a soft phone trying to call *sip:4717081@sipprovider.com* by sending the INVITE request directly to the VoIP provider. As can be seen from the response, the phone call is not successful unless the caller has access (valid credentials) to the network. However, as we shall see, such restrictions can be bypassed easily by making use of freely available tools.

The truth is that (by design) most SIP phones will ring upon receiving an INVITE request. This behavior applies to both soft phones and hard phones. Therefore, if the caller knows the IP and port of the destination SIP phone, he or she can initiate a phone call to the SIP phone without having contact with the SIP provider. In order to find out the IP and port of the SIP phone, we should make use of svmap. It usually helps to know either the port or the IP. Most of the times, SIP phones will listen on the default port 5060. In Listing 12 we scan for SIP devices by making use of svmap and identify an SJphone softphone.

If we send an INVITE request to the softphone at 192.168.1.137:5060 running SJphone, the soft phone starts ringing (see Figure 6). To make a phone call like this, one would typically make use of a soft phone that supports direct calling. In this case, we make use of X-lite and conFigure it to bypass the proxy and contact the target domain directly. The configuration is illustrated in Figure 7. Once X-lite is set up, all one has to do is dial the SIP address, which in this example is sip:1234@192.168.1.137:5060. Depending on the SIP phone, the SIP user part of the address (which is 1234 in this case) may be omitted, or it may even be anything. Therefore, in the case of SJPhone and various other phones, it is simply a matter of finding out the IP address where the SIP service is listening in order to make a phone call without passing through the VoIP infrastructure.

However, in the case of some phones, the address may need to have the exact user with which the SIP phone is registered to the registrar server (PBX or VoIP provider); otherwise the phone would not ring.

A softphone called WengoPhone also shows similar behavior. To identify the valid user on WengoPhone, we

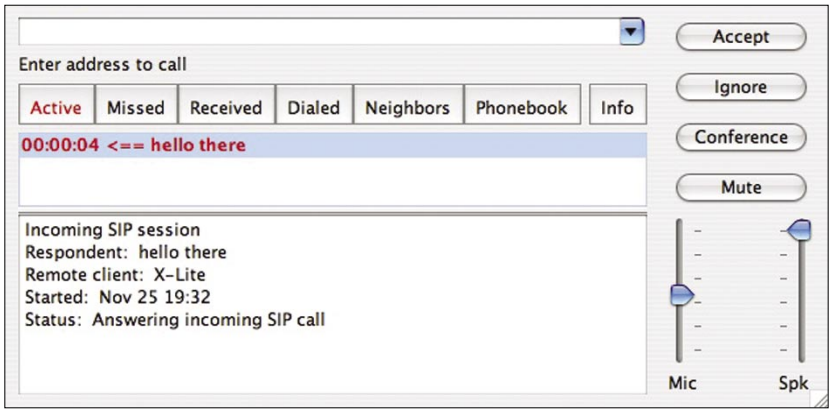

Figure 6. *SJphone rings when it receives an INVITE message*

can make use of svwar. The process is very similar to identifying existing extensions on a PBX, with the difference being that we are targeting the user agent client rather than the server this time. But first, let us find out the port (which is not the default) on which WengoPhone is serving SIP. In Listing 13, we run the command to send its probes to the port, range between 1024 and `65535`. As soon as we identify the port we can stop the scan by pressing `Control^C`. Once we know the port on which the phone is listening, we then make use of svwar to identify the softphone's user. Since SIP phones do not process the `REGISTER` message (which is the default scan method in svwar), we need to make use of the `INVITE` method to identify the correct user (see Listing 14). One should be aware that this method is not exactly stealthy, and will get WengoPhone to ring when a valid user is found. Once a valid extension is found, one can now make a direct call by making use of the extension as the user in the SIP address. In this case, the sip address to call would be *sip:100@192.168.1.112:1169.*

The same concept can be applied to cause a Denial of Service attack which I like to call *ghost phone call*. The objective is to cause a large number of phones on a network to ring at the same time. How is this possible? In most cases, an INVITE scan using svmap on an internal network will cause all phones to start ringing. These phones may keep on ringing until someone manually goes ahead and hangs up the phones. A malicious attacker could very well launch a script which sends these `INVITE` requests until someone stops the script. The simplicity and practicality of this attack is impressive. Such an attack can have a disastrous effect for companies (such as call centers) that rely on IP phones. Listing 15 shows how svmap can be used to send an `INVITE` to a subnet.

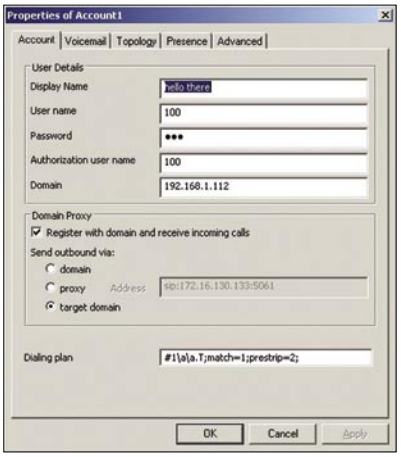

Figure 7. *Configuration of X-Lite for direct calls*

Why would anyone run such attacks? Here are some reasons:

- A disgruntled employee might want to launch a denial of service on the most basic communication service – the phone system
- Malicious users can pull off social engineering attacks with less paper trail by directly calling the IP phone rather than passing through the PBX. Outsiders as well as employees from different departments are known to make use of social engineering techniques.
- Some people might want to pull off the *ghost phone call* as a practical joke. However, in many environments this prank can lead to disruption of service.

Toll Fraud and Password Cracking

Voice over IP service providers typically offer a paid service which allows its clients to make worldwide phone calls at cheaper rates than the traditional phone system. Many VoIP service providers will also give you a phone number reachable from PSTN, thus allowing you to receive calls from the normal telephone network. These services are what makes IP telephony attractive for most customers, and also attackers looking for a cheap way to make long distance calls. Traditionally, phreaks or phone hackers have targeted PBX, access codes, and made use of hardware such as the bluebox (or a simple whistle) to make long distance calls for free. The widespread use of VoIP creates yet another venue for toll fraud. To be able to make fraudulent calls, an attacker typically needs to assume the identity of a valid user on the system by obtaining someone else's username and password. It does not only give access to long distance calls, but also allows the attacker to receive phone calls destined to the victim.

In 2006, the police arrested Edwin Andres Pena, who made over a million dollars through their company named Fortes Telecom Inc. selling VoIP access to smaller service providers. The catch was that instead of buying minutes from larger carriers, Edwin (together with Robert Moore) devised a scheme to route phone calls through illegally obtained user accounts on various VoIP providers. As the story goes, Robert Moore made this possible by scanning for H.323 (another VoIP protocol) devices and trying default or easily guessable passwords on these devices.

Although the criminal duo made use of H.323, similar attacks also apply to SIP. Whatever the protocol, one of the most straightforward ways to obtain the identity of a victim is to guess the password. If the attacker has access to the victim's or provider's traffic, then he or she can make use of tools such as Cain and Abel and sipcrack to launch an offline password cracking attack. This is especially true if no encryption is used. Since SIP makes use of digest authentication which relies on md5, offline password cracking can be very fast, and with tools such as Cain – also very easy (see Figure 8). However, many times the attacker does not have access to the traffic and therefore an offline password attack is not feasible.

In that case, it is still possible to perform a password attack. Instead of making use of an offline password attack, one can use svcrack (which is part of SIPVicious tool suite) to launch an online password attack. Svcrack currently allows up to 80 password guesses per second against a SIP based PBX. This amounts to 6,912,000 passwords a day. It is able to do this by continually sending authentication requests to a SIP registrar server (a PBX such as Asterisk) until

a *200 OK* message is received. If we were to look back at Figure 4, we see how a normal registration works when the SIP user agent client (IP Phone) has the correct credentials. In the case of svcrack, the number of *401 WWW-Authenticate* messages usually indicates the number of passwords guesses until the correct password is supplied.

Let us look at how SIP password cracking typically works. In Listing 16, we have a normal `REGISTER` authentication session. The highlighted parts are the sections that we are interested in. Digest authentication makes use of a nonce, which is a unique string generated each time a 401 response is made. This value is used as part of a challenge to compute a challenge response which is unique. The way that svcrack works is very simple. Instead of making just one `REGISTER` request, it makes various requests which generate different nonce values. For each of these nonce values, it then sends a `REGISTER` request with a challenge response computed using the nonce and a possible password.

Since the digest authentication RFC does not enforce single usage of the nonce in a challenge response, many SIP registrars are known to allow attackers to reuse the same nonce to generate different challenge responses. This is actually an optimization implemented in svcrack and can double the speed at which passwords are checked on certain registrar servers. Listing 17 shows the authorization header when making use of the same nonce and a different password, generated by running svcrack with the optimization enabled (see Listing 18).

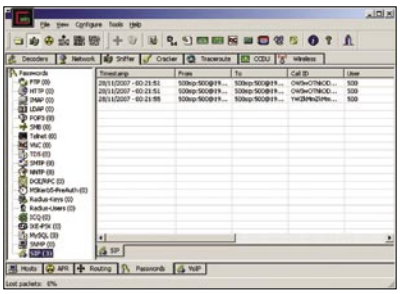

Figure 8. *Cain and Abel SIP cracking*

Detection and Protection

The Basics. When implementing a VoIP infrastructure or any kind of network technology, it is best to try reduce the exposure to attack. The fact that the VoIP infrastructure is typically sitting next to other network entities makes the SIP network elements reachable and possibly vulnerable to an attack coming from the other network servers. The number of VoIP phones and PBXs on the Internet is constantly growing, and if the infrastructure does not require exposure to the Internet, then avoid it. To help you separate the VoIP network from the rest, various network switch vendors allow you to set up a VLAN specifically for VoIP. However, be aware that VLANs are not a panacea, and tools like VoIPhopper make it easy to demonstrate the fact that VLAN is not enough. Cisco published a white paper called *VLAN Security*, where they describe how to protect against a number of attacks aimed at VLAN technology. Segregating the VoIP network can also be done through the use of firewalls or physical separation. VPN tunneling has also been previously suggested because it provides both encryption and can also be used to separate the VoIP traffic from the normal traffic.

However, these solutions might not always be feasible – especially since one major advantage of VoIP is that it integrates with other network elements on the Internet. In fact, various VoIP vendors market the fact that you can use your existing network infrastructure without having to lay new cables. Whether or not this is a good idea depends on a large number of factors. When designing a VoIP infrastructure, it is therefore important to understand the requirements and mitigate depending on the case. For example, a hotel VoIP network will have different requirements than a corporate IP phone network, and therefore a systems designer can apply different security precautions during the planning stage. Some other suggestions and observations:

- It is of course good to make use of encryption mechanisms such as TLS and SRTP. Unfortunately, the encryption for SIP and RTP is not yet widely supported. Zfone by the creator of PGP is particularly interesting. We shall not be going through this subject in depth since it is not within the scope of the attacks described within this article, but it definitely deserves a mention.
- The importance of good passwords for IP Phones should not be underestimated. If the system does not require that end users set their own password, then do not allow this functionality. Instead, make use of some kind of password management and set their password to one that is unique and hard to guess. Applications such as KeePass, which is open-source and free, allow you to generate strong random passwords for you, as well as manage such passwords in a relatively secure manner.
- OpenSER, which is an open-source SIP server, has a module named *pike*. This module is able to block requests that exceed a given limit. This can allow for blocking of both extension guessing and password cracking. However one has to be cautious with such solutions. Attackers can make use of IP spoofing to intentionally block legitimate traffic. It might also unintentionally block legitimate traffic if its not properly conFigured.
- SIP allows extension lines which do not require authentication. If there is no justification for unauthenticated extensions, then make sure NOT to use this feature.
- Hardphones will get security fixes in the form of a firmware update, while softphones will get a new software release. Keeping up to date with the latest versions can be a pain, but it is certainly one way of making sure your system does not fall victim to attackers exploiting a security vulnerability in your SIP phone.

Knowing That You Are Under Attack

Detection is a very important step in a security solution. A network IDS such as Snort, when placed at the right location, can be of great help when trying to detect that an attack is underway. Snocer, which describes themselves as providing *Low Cost Tools for Secure and Highly Available VoIP Communication Services*, has previously published some Snort rules for public consumption. These rules are also available in the latest Snort community rules. In this section we will describe some of them and explain how they can be effective in catching the attacks mentioned previously. We will also provide some new Snort rules which can also detect activity described in this article and not caught by the current Snort community rules.

The Snort rules by Snocer are quite easy to understand, and are able to provide generic detection. Each of the rules looks out for an excessive number of SIP messages coming from a single IP address over a short period of time. The different SIP messages are `INVITE` and `REGISTER` requests, and *401 Unauthorized* messages. The `INVITE` and `REGISTER` flood attacks catch svwar and svcrack being run with default options against a SIP proxy. To be able to catch a default svmap scan, we need to be looking out for SIP messages with an OPTIONS request, spanned over different hosts in a short time. Listing 20 shows one such rule that triggers an alert if the rule is infringed 30 times in 3 seconds. One should probably adjust this rule depending on the address space being watched by Snort. If Snort is watching a /29 mask, i.e. only 6 hosts, then one should change the count to 6 and number of seconds to 1 or less. On larger networks, increase the count number to decrease the chance of a false positive.

The rule on *excessive number of SIP 4xx Responses* attempts to catch the majority of attacks outlined in this article. What it effectively does is match responses which contain a client error. This may be a 404 not found response like the one given by an Asterisk box when running svwar to identify SIP extensions or users. It will also match a password cracking attempt on an Axon PBX, or an extension enumerating attack on a Brekeke PBX when using svwar with the `OPTIONS` method. Of course, it will not catch a network scan for SIP devices on one which does not have a lot of devices, simply because the number of responses would be low.

The *ghost phone call* can also be easily detected since it generates a large number of ringing messages. Of course a payload of this attack is audible, and therefore the benefits of adding this rule might not be immediately apparent since it makes itself so obvious. However, a Snort rule at this stage might be very useful during incident response, when trying to determine things such as the source of the attack. The rule should be modified depending on the network. For example, it does not make sense to deploy this Snort rule on a calling center that takes 50 calls every minute.

Snort is not the only tool to monitor your VoIP infrastructure for attacks. In fact, Snort would very likely NOT detect any attacks passing through encrypted traffic. On the other hand, monitoring the logs on your IP PBX might be a good way of detecting some attacks destined to the SIP gateway. J. Oquendo posted a BASH script called astrap which monitors the Asterisk log entries for excessive number of failed authentication attempts. This small tool will list the offender's IP address, the number of password failures, and the extensions that were targeted on the Asterisk.

A host intrusion detection system such as OSSEC can be equally useful in detecting and automatically mitigating attacks. At the time of writing, OSSEC does not come preconFigured to support Asterisk log files, but this functionality can be easily added. Listing 21 includes a sample rule file for OSSEC to show how it can be conFigured to detect username enumeration and password attacks on an Asterisk system such as Trixbox. Listing 22 shows the changes required to enable this new Asterisk rule. We include a decoder entry so that OSSEC will be able to extract the attacker's IP address and then use that to automatically block the attack by adding the appropriate firewall rule. ●

About the Author

Sandro Gauci has over 7 years experience in the security industry and is focused on analysis of security challenges and providing solutions to such threats. His passion is vulnerability research (a.k.a. breaking things) and has previously worked together with various vendors such as Microsoft and Sun to fix security holes. The latest interest is VoIP and he is currently working on a suite of free security tools to audit SIP network entities called SIPVicious. The tools can be downloaded from *http://sipvicious.org/* Sandro can be reached at *sandro@enablesecurity.com*

On the 'Net

- *http://www.ietf.org/rfc/rfc3261.txt* – RFC 3261
- *http://www.iptel.org/sip/intro/purpose* – Purpose of SIP
- *http://www.wormulon.net/* – smap
- *http://sipvicious.org/* – SIPVicious tool suite
- *http://tinyurl.com/rtjl8* – SIP peers external authentication in Asterisk/OpenPBX
- *http://www.hackingvoip.com/* – SIPSCAN
- *http://www.oxid.it* – Cain and Abel
- *http://tinyurl.com/yph6jy* – Interview with Robert Moore
- *http://tinyurl.com/56bwd* – VLAN Security White Paper
- *http://www.snocer.org/Paper/sip-rules.zip* – Snocer, snort rules
- *http://www.infiltrated.net/scripts/astrap* – astrap
- *http://www.ossec.org/* – OSSEC
- *http://www.trixbox.org/* – Trixbox