Side effects of identity management in SIP VoIP environment

Ge Zhang\textsuperscript{a,*}, Yacine Rebahi\textsuperscript{b}
\textsuperscript{a} Karlstad University, Karlstad, Sweden
\textsuperscript{b} Fraunhofer FOKUS, Berlin, Germany

\textbf{Abstract}

In this article, we summarize the security threats targeting SIP proxy servers or other infrastructures in NGN by misusing a specific signaling authentication mechanism, which has been proposed in RFC 4474 (Peterson and Jennings, 2006). This mechanism is designed to authenticate inter-domain SIP requests based on domain certificates to prevent identity theft. Nevertheless, despite its contribution, this protection raises some "side effects", that actually lead to new vulnerabilities in both the availability and confidentiality of SIP services. We provide an overview of different attack possibilities and explain them in more detail, including attacks utilizing algorithm complexity, certificates storage, and certificates distribution. We also suggest some alternative design to prevent or reduce the attacks. SIP, VoIP, NGN, Authentication, Denial of Service, Timing attack.

© 2011 Elsevier Ltd. All rights reserved.

\textbf{1. Introduction}

The Session Initiation Protocol (SIP) (Rosenberg et al., 2002) is the de facto standard for Voice over IP (VoIP) signaling function in Next Generation Networks (NGN). Users exchange SIP messages to establish, terminate, or modify their VoIP conversations. SIP services are usually deployed over packet-switched networks in an untrusted environment, i.e. the Internet. Thus malicious users in the network can easily access SIP infrastructures to mount attacks. Moreover, they can employ fraud SIP identities and spoof their IP addresses to make themselves untraceable. To mitigate such a problem, a proper authentication mechanism is necessary. RFC 3261 (Rosenberg et al., 2002) proposes an authentication method by using shared secret between users and their service providers (e.g., a caller shares a password with his/her SIP service provider). This method works well in a single domain scenario, in which all users are served by one service provider. However, it is not the case for multiple domain scenario. The reason is that it is operational difficult to spread a user’s secret over all other SIP domains. As a result, the secret-based authentication scheme does not prevent identity theft in a multiple domain scenario. To solve this problem, a certificate-based identity management mechanism is proposed in RFC 4474 (Peterson and Jennings, 2006): For a SIP message across multiple domains, the home domain of the message originator signs the message using its domain private key. When another domain receives this message, it can download the certificate with public key to verify the message. In this way, it is rather difficult for attackers to misuse identities of other users unless they can craft a valid domain signature.

However, this identity management mechanism introduces several “side effects” to SIP services. For one thing, certificate-based verification algorithms (e.g., RSA (Rivest et al., 1978)) contain computation steps with large numbers and thus might overload SIP proxy servers. This can be potentially exploited by attackers to mount Denial of Service (DoS) attacks on a victim proxy server. Moreover, RFC 4474 recommends that a proxy server should cache downloaded certificates for reuse. Although the cache scheme saves CPU and bandwidth resources of the proxy server, it partly reveals the calling records between SIP domains. Attackers might
profile this information by timing attacks. In this paper, we discuss these “side effects” as well as their countermeasures. Furthermore, attackers can take advantage of certificate distribution. A sophisticated attacker can tempt a proxy server to connect to a selected host in the network. This vulnerability can result in two problems: (1) If connection between the proxy server and the host is in a bad condition, the proxy server has to spend more time on certificate downloading. Thus the proxy server might be blocked. (2) If there are a number of proxies servers in the network, the attacker can tempt them to connect to a host at the same time. In this way, the resources of the host (e.g., bandwidth, CPU) will be consumed considerably.

The remaining part of this paper is organized as follows. We introduce related background such as SIP and its identity management mechanism in Section 2. The side effects caused by the identity management mechanism as well as its impact are investigated in Section 3. Finally, we summarize our work and present the outlook in Section 4.

2. Background

In this section, we introduce some background information of SIP and the identity management mechanism proposed in RFC 4474.

2.1. The Session Initiation Protocol (SIP)

The Session Initiation Protocol (SIP) is a signaling protocol designed to establish, modify and terminate a media session between endpoints. Standardized by IETF working groups, it has been taken as a standard signaling protocol for VoIP in the Internet and NGN. SIP users are identified by means of Uniform Resource Identifier (URI) (Berners-Lee et al., 2005), a universal name with a pair of domain name and user name, e.g., (sip:ge.zhang@kau.se). Most current SIP applications in reality employ the client/server transaction model similar to HTTP. A SIP client generates SIP requests and a SIP server responds by generating responses. Both SIP requests and responses are following the message format with three elements: the first line, containing either a request method or a response code; headers, containing a list of message headers with values for SIP transaction; message payload, can be text-based content for different purpose (e.g., SDP payload). A basic SIP architecture consists of User-Agents (UAs) and several SIP servers, such as registrar servers, proxy servers (called proxy or proxies in the rest of this paper), etc. UAs are the users’ equipments which generate, receive and response SIP messages. Registrar servers enable users to login to corresponding domains. Proxies forward SIP messages within SIP networks. Other operations (e.g., authentication) can also be enforced on proxies.

The general SIP-based telephony calling setup is shown in Fig. 1. There are two SIP users located in different domains in this scenario: Alice, a caller, is located in the domain kau.se and Bob, a callee, is located in iptel.org. Initially, Alice sends an INVITE request to the local proxy. This INVITE request indicates that she wants to talk to Bob at iptel.org. Then, the local proxy forwards this INVITE request to the remote proxy at iptel.org domain. The request is finally delivered to the UA of Bob. If Bob accepts the call, his UA will reply with a 200 OK response back through the proxies. After Alice confirms the request with an ACK request, the signaling handshake is accomplished. Thus, Alice and Bob establish a peer-to-peer media session in which they can exchange voice packets. When Alice wants to tear down the conversation, her UA will send a BYE request to Bob, and Bob’s UA will reply with a 200 OK response. Then the call is terminated.

In the rest of this paper, we define a SIP request message forwarded between different SIP domains as an inter-domain request. For a given inter-domain request, the proxy which forwards the request is defined as an originator proxy, and its domain is named the originator domain. The proxy that receives the request is called a recipient proxy, and the terminating domain is called the recipient domain.

2.2. Identity management based on RFC 4474

Malicious users can mount attacks with fraud SIP identities and spoofed IP addresses in order to be untraceable. To prevent this, SIP proxies should authenticate the source of requests to assure that the originator is exactly the one he/she is claiming to be. One of the most widely used authentication methods is based on shared secrets (e.g., user name and password) (Rosenberg et al., 2002). However, this method is not scalable for a SIP network with multiple SIP domains. In particular, it is difficult to handle the trust issues if the secret information are widely distributed. As a result, identity fraud cannot be totally prevented in SIP networks by simply employing password scheme.

Nevertheless, RFC 4474 (Peterson and Jennings, 2006) proposes an identity management mechanism to solve the problem. It aims at helping the recipient proxy to authenticate an inter-domain request. For example, a SIP request issued, apparently, from kau.se could not be sent by a user outside kau.se. RFC 4474 assumes that there is a trusted Certificate Authority (CA) that is able to issue certificates, which could be in particular acquired by the recipient proxies in the network. The mechanism described in RFC 4474 is depicted in Fig. 2 and works as follows,

1. The originator proxy owns a private/public key pair with a corresponding certificate which is issued by a trusted CA (e.g., verisign). The originator proxy also performs...
password-based authentication to verify the requests from its own users.

2. When an originator proxy processes an inter-domain request from a user, the originator proxy computes a hash digest on some header fields as well as the body of the SIP request, signs it with its private key and inserts the result as a new header field Identity in the same SIP message. The SIP proxy also creates a second new header field Identity-info and inserts the Uniform Resource Locator (URL) where its domain certificate can be downloaded, as well as corresponding verification parameters (e.g., which algorithm will be used for verification). Finally, the updated SIP message is forwarded to the recipient proxy.

3. When the recipient proxy receives the SIP request, it downloads the certificate of the originator domain according to the URL within the Identity-info header field. The recipient proxy first verifies the certificate. If the certificate is signed by a trusted CA, the recipient proxy uses the public key retrieved from this certificate to verify the signature in the Identity header field. The request will be further processed only if the signature can be successfully verified. In addition, the verified certificate will be cached in the recipient proxy for a period of time. Thus, the proxy does not need to repeatedly download the same certificate for the messages came from the same originator domain.

2.3. The state of the art

The main objective of Identity Management services is to authenticate users and to support flexible access control to the services and resources, based on user identity properties (also called attributes). If we take SIP as an example, within this context, some standard solutions were suggested (RFC 3621). SIP supports Digest, hop-by-hop security using Transport Layer Security (Dierks and Rescorla, 2008) and end-to-end security using Secure MIME (S/MIME) (Ramsdell, 2004). It is clear that end-to-end security provides a higher level of security than hop-by-hop. However, some of the existing solutions are based on hop-by-hop security which makes them inadequate. RFC 4474 is one of the few mechanisms that provide end-to-end security based on public key cryptography. Unfortunately, this mechanism, if it is implemented in its current state, can harm more than help especially that it brings some vulnerabilities that might lead to some cyber attacks in addition to a questionable performance as it will be discussed later on.

3. Side effects introduced by RFC 4474

3.1. Side effect 1: performance penalty for signing/verifying

With the identity management mechanism, the originator proxy generates a signature for each out going request. On the other hand, the recipient proxy should verify the digest of each incoming request. Thus, both of them need to negotiate to employ the same cryptographic algorithm. This first introduced side effect of this mechanism is performance overhead caused by signing and verifying. In RFC 4474, the Rivest-Shamir-Adleman (RSA) (Rivest et al., 1978) algorithm is recommended to be implemented. Currently, the RSA algorithm is considered to be secure when sufficiently long keys are used (512 bits is insecure, 768 bits is moderately secure, 1024 bits is secure and 2048 bits should remain secure for the foreseeable future).

The results of our previous experiments shows that employing RSA algorithm introduces heavy overload on SIP proxies (Rebahi et al., 2008). We first built a testbed which has the capability to handle 920 calls/s without the identity management mechanism implemented. Thus, we take the 920 calls/s as a performance benchmark. We later implemented the mechanism on the testbed with different RSA key sizes (1024, 2048, 3072 bits) and restarted the experiments. We found that the testbed only can handle 123, 28 and 12 calls/s respectively. Considering that VoIP is a high realtime application, the performance penalty might leads to Denial of Service attacks. For example, attackers can flood the victim proxy using requests with signatures to bring down the proxy.

Elliptic Curve Cryptography algorithm (ECC) (Miller, 1986) is considered performance-friendly since it can provide high level of security with relatively small key size. As recommended by the National Institute for Standards and Technology (NIST), RSA uses 1024 bits key while ECC provides equivalent security for a key size of about 160 bits. Further equivalent key sizes for ECC and RSA are presented in Table 1.

To investigate the performance impact by using ECC, we also implemented ECC algorithm in our testbed. We repeated
the experiments mentioned above with ECC keys of 160, 224 and 256 bits, which are considered equivalent to RSA keys of 1024, 2048 and 3072 respectively. The results are shown in Table 1. We found that ECC does provide a better performance than RSA in SIP identity management implementation. However, compared to performance of the testbed without the mechanism implemented (can handle 920 calls/s), employing ECC is still not a satisfying solution.

### 3.2 Side effect 2: peeking from a certificate cache

In RFC 4474, a certificate cache is recommended to avoid the performance overhead by repeatedly downloading the same certificates. In this way, after a recipient proxy successfully downloaded a certificate, the certificate is supposed to be cached on the recipient proxy for a period of time. Later the cached certificate can be reused. Obviously, the performance can be enhanced by employing certificate cache. However, as a two-edged sword, cache may leak information. For instance, an attacker can profile which domains sent requests to “kau.se” recently if he/she knows the content of the cache on the proxy of “kau.se”.

One way to profile a cache is timing attack. We define that the time interval between one request and its response as Round Trip Time (RTT). The principle behind the attack is to exploit the RTT. In most cases, the motivation of using cache is to save time of actions by avoiding repeated operations. Then, cache might cause RTT difference for requests, depending on their certificate has been cached or not. Therefore, an attacker can pretend to be a specific originator proxy and sends crafted requests to a victim proxy. The attacker then observes the RTT of these requests. As shown in Fig. 3, we assume that proxy A and web server B are responsible for domain A and domain B respectively. The attacker aims to find out whether domain B’s certificate has been cached in proxy A. Therefore, the attacker sends a craft request to proxy A using a request with Identity-info header field of web server B’s URL (We call this kind of request as probing request). If the certificate has been cached, proxy A does not need to download it. Otherwise, proxy A needs to spend additional time ($t_d$) on downloading. Please note that it does not matter whether the probing request can be successfully verified or not. Attackers just concern about the RTT of requests.

Zhang et al. (2009) proposed the attacking method in detail: An attacker can continuously send the same probing request twice. If the certificate has been cached before the attack, the victim proxy does not download certificate for both of the two requests. Therefore, the RTT for the two requests should have a difference around $t_d$. The result of our experiments indicates that the $t_d$ can be from 0.5 to 1.5 s depending on the downloading website and methods (by HTTP or HTTPS). Thus, the $t_d$ in reality is large enough for the attacks.

This timing attack reveals the calling records among domains. The attack is especially attractive against enterprises’ SIP domains. Generally, two enterprises frequently communicate when they have a business connection. Therefore, the calling history might reveal such potential business connections. Companies usually have an interest to keep their business relationship confidential. In particular, companies that secretly talk about merging would like to keep this confidential in regard to the outside world. Therefore, the disclosure of the calling history might cause direct or indirect economic losses for a company. In the traditional PSTN context, the calling history of an enterprise is kept secret by the telephony operator, a trusted third party in a closed context. It needs more efforts for an external attacker to gather this information. However, in the SIP VoIP context, the telephony service is integrated with other ICT-based infrastructures based on an open environment, which simplifies the attack.

If a request cannot be successfully verified by the recipient proxy, the proxy will issue a negative response indicating the reason of failure. As we assumed in this chapter that the attackers cannot produce valid signatures for their probing requests, thus all the probing requests only can result in negative responses. An intuitive defending method is that a proxy does not send negative responses. In this way, the attackers do not have enough knowledge to observe the RTT of their probing requests. Nevertheless, this method is not robust since legitimate originator proxies may would like to have negative responses to find out the reason of failure.

A better design is that the recipient proxy still sends negative responses. However, the proxy enforces a constant RTT (saying $T_{const}$) for all negative responses. We let the proxies to track the time from receiving a request to generating a response for it (saying $T_i$ for a request $i$). The proxy then delays each generated negative response for a time period of $T_{const} - T_i$ before sending it out. With equalized RTT for negative responses, the timing attack does not work. Nevertheless, responses delayed in the proxy may consume memory resources. We did further research in (Zhang et al., 2009). For example, we set $T_{const} = 10$ s and let a client to send a proxy with 20 probing requests per second. The result shows that the increased memory consumption is around 270 kB, which is acceptable in most current server machines.

### Table 1 – Equivalent key sizes between of ECC and RSA, and their performance comparison in our SIP testbed.

<table>
<thead>
<tr>
<th>RSA Key Size</th>
<th>ECC Key Size</th>
<th>Capability of SIP proxy (RSA)</th>
<th>Capability of SIP proxy (ECC)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1024 bits</td>
<td>160 bits</td>
<td>123 Calls/s</td>
<td>263 Calls/s</td>
</tr>
<tr>
<td>2048 bits</td>
<td>224 bits</td>
<td>28 Calls/s</td>
<td>128 Calls/s</td>
</tr>
<tr>
<td>3072 bits</td>
<td>256 bits</td>
<td>12 Calls/s</td>
<td>104 Calls/s</td>
</tr>
</tbody>
</table>

3 A number of reasons can make the verification unsuccessful. For example, failure of certificate downloading, or the recipient proxy does not trust the CA who signed the certificate.
3.3. Side effect 3: blocking attacks by certificate downloading

This attack is a Denial of Service attack on SIP proxies. A recipient proxy does not process a request unless this request can be successfully verified. To do so, the recipient SIP proxy needs the certificate of the original domain. Thus, the processing of a request is held on during the time interval of getting the certificate, either from a web location or local cache. This attack takes advantage of the high latency for certificate downloading. Considering the authentication operation, two basic options can be implemented on a proxy:

- **Synchronous operation:** In this case, the recipient proxy downloads the certificate from the URL in the Identity-info header field. Besides waiting the certificate, the process will not do any other further work. Actually, this proxy is blocked and unavailable to other requests during this time interval.

- **Asynchronous operation:** In this case, a proxy process downloads the certificate after temporally saving the state and data of the current processed request into an event queue. Thus, this process is able to process other requests during the time interval of certificate downloading. As long as the downloading procedure is finished, the process will continue to handle the suspended request. In this way, the process will not be blocked by certificate downloading. However, additional memory resources of the SIP proxy will be occupied to save information of suspended requests.

As a consequence, the longer time for a certificate downloading takes, the more negative impact on a SIP proxy can lead to. Depending on the implementation, either the proxy is blocked for a while or additional memory space of the proxy is consumed. In this paper, we mainly focus on the synchronous operation since most proxies are implemented in this way (e.g., SER). Given an attacker aiming at reducing the availability of a SIP proxy, he/she can realize the goal by tempting the victim proxy to do certificate downloading for nothing. In reality, there are two methods:

- An attacker can setup his/her own web server, with configurable artificial delay for each downloading attempt on purpose. The web server can be built on a compromised networking host to make the attacker untraceable.

- Instead of creating his/her own web server, an attacker can also collect a set of hosts which will take long time for the victim SIP proxy to connect to. For example, the latency between two Internet hosts can be up to 60 s according to the MIT King Dataset (The MIT "King" dataset, King is a method to estimate the latency between two arbitrary hosts on the Internet (Gummadi et al., 2002).

The attacker can thus craft an inter-domain SIP request targeting the victim SIP proxy and fill the Identity-info header field with the URLs of prepared web servers. In case of the result being cached, the URLs can point to randomized filenames at the web servers (e.g., http://latencyhost.com/[random_string]). Since the randomized files cannot be found on the web servers, the authentication for the attacking requests will certainly result in a failure. However, the victim proxy has already spent much time on connecting before it detects the failure. The procedure is illustrated in Fig. 4. We introduced some experiments for this attack in Zhang et al. (2010a). We built a testbed in which there is a proxy can handle 500 calls/s without the attacks. We configure the proxy with 4 and 16 parallel process queues. We have a web server which is configured to delay for \(d\) seconds to simulate the latency. Then we let an attacking tool to send attacking requests containing identity-info fields with the URL of the prepared web server. Thus the proxy will connect the web server after it gets an attacking request. The result shows the processing capability of the proxy reduces considerably by the attacks. For example, with \(d = 6\) and attacking rate 6 requests/s, the proxy (4 parallel process queues) can only process 50 calls/s. It means the proxy capacity reduces to only 10% of the benchmark.

Fig. 3 – The working flow shows the comparison of time cost in two situations, one with the certificate already cached, another not.
We use cache with priority mechanism to counteract this attack (Zhang et al., 2010a). We assume that some certificates of legitimate parties have been cached in a recipient proxy. Given the proxy with $n$ parallel processing queues, we take one of them as an emergency process queue. The emergency process queue only searches certificate in the local cache when other $n-1$ processes are blocked. If the certificate is not cached, the emergency process queue deals with the request as no certificate. In this way, the proxy produces a negative response for the request and continue to handle the next one. This design cannot eliminate the impact of the attack totally. Nevertheless, by using this design, a victim proxy will not be blocked: It at least can process the requests with their certificates which has been already cached.

3.4. Side effect 4: reflect attacks by certificate downloading

The attacking target in this case is not on SIP proxies, but the attackers use SIP proxies as reflectors to flood other hosts (e.g., HTTPS server) in the network. To do so, an attacker firstly selects a list of SIP proxies and a victim HTTPS server in the network. We assume that the victim host owns a domain name: victim.com. The attacker then crafts SIP requests with Identity-info header field containing https://victim.com/[random_string]. The attackers finally flood the SIP proxies using the requests. As a result, all the proxies will try to build HTTPS connections to the victim web server. Therefore, much resources of the HTTPS server are occupied so that the availability of the proxy is reduced. Fig. 5 illustrates the attacking procedure. The principle behind the attack is the unbalanced resource usage between the attacker and the HTTPS server: Attackers can send requests using SIP based on UDP; While the web server has to build connections for each request using HTTPS based on TLS. UDP is
a connectionless oriented protocol thus it consumes little resource. On the other hand, TLS is a connection oriented secure protocol with certificate exchanging and algorithm negotiating. Our experiments in (Zhang et al., 2010b) shows that the HTTPS server consumes much more CPU resource than the attack does (up to 20 times) due to cryptographic operations. We further configured a testbed with 4 SIP proxies as reflectors. Without attacks, the average responding time of our HTTPS server was a few milliseconds. However, the average responding time can be up to 20 s with attacks (Zhang et al., 2010b).

It is not easy to avoid this problem unless we modify the method of the certificate distribution in the identity management mechanism. According to the specification, it is the recipient proxy which is responsible for downloading the certificate of the originator domain. This action is qualified as pull a certificate by the recipient proxy. This pull action produces the vulnerability, which makes the victim proxy connect to a host arbitrarily controlled by the attacker. In contrast, we consider the push mechanism: an originator proxy actively pushes a certificate along with the request to the recipient proxy. For example, the certificate can be enco-
ded in a request as a message payload.

However, the push mechanism faces a variety of challenges. First, the originator proxy should know whether its certificate has been cached in the recipient proxy to avoid repeatedly sending certificate. Thus, a negotiation between the proxies is needed before sending a request. However, the negotiation takes time and also consumes bandwidth. Moreover, the maximum transmission unit (MTU) for a IP packet is 1500 bytes or so in an ethernet network. Nevertheless, the size of a SIP request plus certificate is usually more than MTU (Zhang et al., 2010b). Thus, if we use UDP, the reception of the fragmented packet may encounter potential errors (Kent and Mogul, 1995). As a result, the proxy only can send SIP requests based on TCP. Thus, the push mechanism will not be widely accepted if we take the scalability into account. Therefore, it is still an open question to solve this problem.

4. Conclusion

The identity management mechanism proposed in RFC 4474 is a promising scheme to counteracts identity fraud in SIP networks. However, there are several side effects in the mechanism. These side effects might be exploited by attackers to launch other attacks (e.g., calling record profiling and Denial of Service). In this paper, we have presented the four side effects based on our previous research work. The first introduced problem is caused by the computational overhead using RSA algorithm. We showed the quantitative comparison results of a SIP testbed with and without the identity management mechanisms implemented: the performance decreases sharply after the mechanism is deployed. The implementation employing ECC algorithm gives a better result, but still not satisfying. The second effect takes advantage of the certificate cache. Attackers can profile the content of a certificate cache on a proxy by comparing the round trip time of requests. The attacker can further guess the information of calling records from the cache entries. We discuss a solution to prevent by enforcing a fixed RTT to prevent this threat. The other two effects are caused by the impropriate design of certificate distribution: A proxy should download the certificate according to the URL specified in the SIP request. However, at the moment of certificate downloading, the SIP request is still unverified. Thus, attackers can tempts a proxy to connect any host in the network and it may result in Denial of Service on the proxy or other hosts. Although we proposed countermeasure solutions, they are still rather limited to eliminate the impact of attacks.

In the future work, we will be focused on exploring a unified and usable solution to prevent the threats introduced by the side effects. We will also focus on improving the certificate distribution mechanisms.

References


Peterson J, Jennings C. Enhancements for authenticated identity management in the Session Initiation Protocol (SIP); 2006. RFC 4474.

Ramsdell B. Secure/multipurpose internet mail extensions (s/mime) version 3.1 message specification; 2004. RFC 3851.


