Radius/LDAP authentication in open-source IP PBX

Ivan Capan, Marko Skomršić
Protenus d.o.o.
Telecommunications & networking department
Zrinskih i Frankopana 23, Varaždin, 42000, Croatia
ivan.capan@protenus.com, marko.skomersic@protenus.com

Abstract – In this paper the performance and scalability of a SIP (Session Initiation Protocol) registrar server integrated with an existing directory systems such as LDAP (Lightweight Directory Access Protocol), Active Directory, etc. will be explored and discussed. Applications based on open-source software will be used in all segments of the system, and open and standardized protocols will be used for communication. Security will also be discussed.

I. INTRODUCTION

The idea of this work was to use open and standardized communication protocol so we have decided to use SIP [1] which is an application layer control protocol developed and designed within the IETF (Internet Engineering Task Force, the Internet standards body). SIP is used for creating, modifying, and terminating sessions with one or more participants. By sessions we understand a set of communicating senders and receivers and the states kept by them during the communication process.

While going through a typical SIP session the caller doesn’t know the exact address of the callee initially. The job of finding out the exact location of the recipient is done by SIP proxy server. To accomplish this job successfully, every UAC (User Agent Client) registers its current location to a registrar server by sending a REGISTER message. Registrar stores the bind between the UAC and its current address in a location server which is used by proxies to locate the UAC. Each REGISTER message consists of fields and its attributes. Each registration has a limited lifespan. Expires header field or Expires parameter of Contact header field determines how long the registration is valid. The user agent must refresh the registration within the lifespan otherwise it will expire and the user will become unavailable. While registering UAC onto the system an UAC authentication can be included. UAC’s unique credentials are also included in its REGISTER message. In case of required authentication, registrar server must compare received credentials (username, password, realm, etc.) with unique UAC’s credentials stored in the registrar database [2].

In the telephone systems with large numbers of UAC’s there are peaks of incoming REGISTER messages which must all be handled by the registrar server. These requests generate a load on it, especially when authentication is required (non-anonymous access). To successfully accomplish registration or re-registration process registrar server must be properly scaled. This scaling can be done by choosing a hardware which is powerful enough to process registration requests in reasonable time, by carefully chosen architecture or by combination of the first two choices.

Integration of VoIP system with already present directory system is usually a requirement. These present directory systems are mostly LDAP based [3], such as OpenLDAP, Microsoft Active Directory, etc.

Open-source VoIP solutions are used because of its numerous advantages compared to proprietary solutions. For example there is more control of entire system, wider span of experts who can perform maintenance and integration, there is no need for licensing fees, and it’s easier to customize the source code [3]. One of main advantages of open source - availability of source code - can also be exploited in finding security issues by malicious hackers. Most of open source projects don’t have some commercial company or funding behind it. That does not necessarily influence quality of software, but can prolong development, as it is mostly driven by enthusiasm and free time of programmers. Because of that, roadmaps and strict schedules are not something that can be relied upon. Lack of funding can also mean that there is no staff to write proper documentation. That makes installing and integrating of software harder, but there is always plenty of people in the open source community willing to help. Some open source companies provide commercial support lines. In the extreme cases, source code and the comments within it can be read and better understand the program, but it usually requires good knowledge of programming.

Among large number of open-source solutions there are few which are well supported and tested such as Asterisk or OpenSER solution.

In this work we have made a stress test of those two by sending a large number of SIP REGISTER messages, and have measured RPS (Registrations Per Second) value for a different number of concurrently sent messages. RPS value is valuable parameter in scaling a registrar server which will be capable of successfully registering UAC’s within a required time. For example, if we have a registrar server which has RPS value 15 it can successfully register 15 UAC’s per second.
II. LEARNING AND TESTING

In our research we have used SIPp, standard open-source tool for testing SIP VoIP communication [4]. SIPp is capable of creating custom SIP scenarios which include sending and receiving SIP messages. It features the dynamic display of statistics about running tests. SIPp is also very useful for simulating thousands of UAC’s making calls to VoIP system. We have generated a CSV (Comma Separated Value) file which contains username – password pairs for each test UAC as shown in Fig. 1. We have also developed a custom register scenario described in Fig. 2. Scenario is created using XML (Extensible Markup Language) syntax.

2000;[authentication username=2000 password=1234]
2001;[authentication username=2001 password=4321]

Fig. 1. CSV file used by SIPp scenario with username-password pair for two users.

In our SIP scenario we have fixed T1 timer value to 500 milliseconds although T1 timer is dynamically changed by UAC’s. The RFC states that the T1 timer should be equal to “the estimated round-trip time to the destination” or it should be 500 milliseconds if the round-trip time is unknown. This is only for the value of T1 in first sent message, but it must be changed by every next sent message to a double value of previous T1 value [1]. This is not always properly handled especially by software SIP clients, who have often fixed T1 timer.

2000;[authentication username=2000 password=1234]
2001;[authentication username=2001 password=4321]

In our SIP scenario we have fixed T1 timer value to 500 milliseconds although T1 timer is dynamically changed by UAC’s. The RFC states that the T1 timer should be equal to “the estimated round-trip time to the destination” or it should be 500 milliseconds if the round-trip time is unknown. This is only for the value of T1 in first sent message, but it must be changed by every next sent message to a double value of previous T1 value [1]. This is not always properly handled especially by software SIP clients, who have often fixed T1 timer.

<send retrans="500"><![CDATA[
  REGISTER sip:[remote_ip] SIP/2.0
  Via: SIP/2.0/[transport] [local_ip]:[local_port]
  From: <sip:[field0]@[remote_ip]>
  To: <sip:[field0]@[remote_ip]>
  Call-ID: [call_id]
  CSeq: [cseq] REGISTER
  User-Agent: SIPp/Linux
  Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, NOTIFY, MESSAGE, SUBSCRIBE, INFO
  Contact: <sip:[field0]@[local_ip]:[local_port]>
  Content-Length: 0
  Expires: 300
]]>]]></send>
<recv response="100" optional="true"></recv>
<recv response="401" auth="true"></recv>
<send retrans="500"><![CDATA[
  REGISTER sip:[remote_ip] SIP/2.0
  Via: SIP/2.0/[transport] [local_ip]:[local_port]
  From: <sip:[field0]@[remote_ip]>
  To: <sip:[field0]@[remote_ip]>
  Call-ID: [call_id]
  CSeq: [cseq] REGISTER
  User-Agent: SIPp/Linux
  Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, NOTIFY, MESSAGE, SUBSCRIBE, INFO
  Contact: <sip:[field0]@[local_ip]:[local_port]>
  [field2]
  Content-Length: 0
  Expires: 300
]]>]]></send>
<recv response="100" optional="true"></recv>
<recv response="200" optional="true"></recv>

Fig. 2. Custom SIPp XML scenario for registration of SIP UAC, which is composed of two REGISTER messages.

Fig. 2. shows the SIPp scenario which consists of two blocks. First block contains a REGISTER message (without UAC’s credentials) sent to a SIP server which responds with two messages (100 Trying and 401 Unauthorized). Second block contains a REGISTER message with UAC credentials and is also sent to a SIP server which now responds with messages 100 Trying and 200 OK. When SIPp receives 200 OK message, registration process is successfully finished.

Registration flow process is described in Fig. 3. Using LDAP as an authentication server cannot be done directly, because authentication in LDAP is simply a binding procedure with UAC’s password. It cannot handle the SIP digest authentication mechanism. UAC’s authentication against an LDAP server can be achieved by putting RADIUS entities (client and server) in between the SIP registrar and the LDAP server, or in the case of Asterisk, a custom LDAP module. This specific architecture and authentication flow is shown in Fig. 4.

For the first and second measurement as a registrar server we have used the Asterisk IP PBX and OpenSER, respectively. We have conducted tests by generating a 1200 virtual UAC’s that were only defined in the LDAP directory. The next step was to generate 1200 UAC’s username – password pairs in the CSV file (Fig. 1.), which will be used by SIPp test tool.

To successfully connect Asterisk with RADIUS client it was needed to adopt a custom software patch that will force Asterisk to authenticate UAC’s against RADIUS/LDAP server.
Our goal was to measure RPS value for exact number of UAC’s when more REGISTER messages were sent to the registrar in the same time. In the case of Asterisk server results have shown that it was almost useless with larger number of UAC’s (e.g. more than 900). We have concluded that the main reason for lower RPS value is the number of user records stored in LDAP. Because of that, we wanted to find a dependency between that number and the RPS value with fixed speed of registering.

With OpenSER we had far better results that were not dependent with number of LDAP records. Because of that, we kept that number at arbitrary large value. Measurement results are shown in Fig. 6. to Fig. 11.

These figures show the dependency of proposed and measured RPS value while the total number of UAC’s REGISTER requests was kept constant for each measurement. For example, in Fig. 6. we have sent the total amount of 200 REGISTER requests, and in case of proposed RPS value set to 180, measured RPS was 33.65 so to successfully register all of the 200 UAC’s we’ll need 5.94 seconds. To successfully register 800 UAC’s with proposed RPS value set to 600 we’ll need 31.43 seconds. This is shown in Fig. 9. The large drop shown in Fig. 8. to Fig. 11. (every testing with more than 600 UAC’s) can be explained by pauses in retransmitting REGISTER messages. These pauses are caused by multiplying of T1 timer value in the n-th retransmission iteration. This is normal behaviour proposed in section 17 of RFC 3261. Pauses can be reduced by fine tuning the values of proposed RPS and/or the T1 timer.
Fig. 8. OpenSER test results for 600 UAC’s

Fig. 9. OpenSER test results for 800 UAC’s

Fig. 10. OpenSER test results for 1000 UAC’s

Fig. 11. OpenSER test results for 1200 UAC’s

### IV. SECURITY ISSUES

The main concern with UAC registrations is the plaintext nature of SIP network transport. TLS (Transport Layer Security) which encrypts network transport with SSL (Secure Socket Layer) support for SIP registrars (OpenSER has it natively, Asterisk only as an unofficial patch) is mainly for the inter-server communication and does not involve UAC registration.

Fortunately, SIP has a way of encrypting logins and passwords by hashing them with a one-way md5 algorithm [6]. Encrypted in this way (Digest authorization - logins, realms and password are stored in the same hash), they can be matched against a password stored in LDAP. They can’t be decrypted even if intercepted on a network, but can only be used in reply attacks (sending the same hash even without knowing the original password). Password in LDAP must already be encrypted, making it useless for other services which need authentication, if any. In that case, passwords in LDAP are in plaintext, vulnerable to sniffing.

Extra hashing of passwords on the client side does not produce measurable overhead so it is not included in our testing. Communication between Radius and LDAP directory can be natively TLS encrypted and it is not a security issue.
A comparative study of registration speed with Asterisk and OpenSER VoIP registrars with LDAP backend has been performed. It has been shown that the OpenSER has greatly outperformed Asterisk. More stable features of OpenSER were more optimized when scaling to a large number of UAC’s. We have founded that measured RPS value depends on T1 timer value (retransmission time) so in the future work, it’s needed to find a dependency of T1 timer and proposed RPS value in order to reduce those pauses.

Asterisk in the current development state of LDAP module is either very slow or almost useless when the directory is populated with larger number of entries (e.g. more than 1000). Hardware scaling didn’t help in this case because of poorly performing algorithms in LDAP connector. It also has issues with T1 timer, but overall they are negligible.

REFERENCES