Using Session Initiation Protocol with IBM® Lotus® Sametime®

Todd Page
Software Engineer
IBM Software Group, WPLC
Austin, TX

Kendell Zahn
Software Engineer
IBM Software Group, WPLC
Austin, TX
Introduction

The purpose of this article is to present a brief introduction to the Session Initiation Protocol (SIP) and then describe how IBM® Lotus® Sametime® uses this protocol to enable instant messaging and presence awareness between two or more Lotus Sametime communities. You will learn the steps necessary to connect two Lotus Sametime SIP communities. Once a basic setup is complete, you will also learn how to enable SIP features such as Transport Layer Security (TLS) encryption and how to use SIP with distributed and clustered Lotus Sametime environments.

We assume that the reader is already familiar with installing, configuring, and administering an IBM® Lotus® Domino® and Lotus Sametime server, and with how to use the Lotus Sametime Connect client. We also assume that you already have a Lotus Sametime community configured and that you wish to now add in SIP functionality.

For more information on getting started with Lotus Sametime, the Administrative and Install Guides can be found at the Lotus Sametime Product Documentation site.

IBM currently uses the SIP protocol for instant messaging within an IBM Workplace™ environment and for connecting Lotus Sametime to other external Sametime servers. This paper will focus only on SIP within the context of Lotus Sametime; using SIP with a Workplace environment will not be discussed. If your goal is to connect Lotus Sametime with a Workplace environment, there is another product available, the Lotus Instant Messaging Gateway that facilitates communication between the Workplace SIP infrastructure and Lotus Sametime. You can find more information on setting up and configuring the Lotus Instant Messaging Gateway in the IBM Redpaper “Integrating IBM Workplace with IBM Lotus Instant Messaging and Webconferencing (Sametime)”.

Overview of SIP

What is SIP?

SIP is a standardized protocol developed by the Internet Engineering Task Force (IETF). It was developed as a mechanism to initiate, terminate, and modify interactive user sessions involving audio and video, voice, and instant messaging. On a network level, SIP looks much like the HTTP (Web) and the SMTP (email) protocols. A SIP session could be a phone call over IP, an instant messaging session, or a multimedia session such as an audio/video meeting. The original SIP specification was defined in RFC 2543, but this has since been made obsolete by RFC 3261.

SIP can be used for:
- Internet conferencing
- Telephony (calling a user from the Internet)
- Presence
- Events notification
- Instant messaging

Methods Defined

The SIP RFC defines a number of methods (or commands) that each SIP endpoint will use during the session. The session methods for SIP include the following:
- **INVITE** Invites a user to a SIP session
- **ACK** Facilitates reliable message exchange for INVITEs
- **BYE** Terminates a connection between users or declines a SIP session
- **CANCEL** Terminates a request, or search, for a user
- **OPTIONS** Solicits information about a server's capabilities
- **REGISTER** Registers a user's current location
- **INFO** Is Used for mid-session signaling

Lotus Sametime SIP also uses RFC 3265, “Session Initiation Protocol (SIP)-Specific Event Notification”, which includes these two event methods:

- **SUBSCRIBE** Informs external communities that a local user is attempting to resolve a SIP user’s status on their Buddy list.
- **NOTIFY** Indicates a user’s online status. A NOTIFY will be sent to the other SIP Community if a local user has just logged in, gone into an active, away, or Do Not Disturb (DND) status, or has logged of.

The following are SIP responses:

- 1xx Informational (e.g., 100 Trying, 180 Ringing)
- 2xx Successful (e.g., 200 OK, 202 Accepted)
- 3xx Redirection (e.g., 302 Moved Temporarily)
- 4xx Request Failure (e.g., 403 Forbidden, 404 Not Found, 482 Loop Detected)
- 5xx Server Failure (e.g., 501 Not Implemented)
- 6xx Global Failure (e.g., 603 Decline)

SIP runs on the Application Layer of the TCP stack. Other familiar Application Layer protocols include HTTP, FTP, SSH, TELNET, and POP3. Information in SIP packets normally includes:

- The type of request (**SUBSCRIBE, MESSAGE, and so on**).
- A **To**: and **From**: address containing an email addresses of the two SIP users in session, not including the header field.
- An **Expires** showing the number of seconds the content of the message is valid.
- A **Max-Forwards** field that is decremented between each SIP server. Not normally used in Lotus Sametime SIP configurations; instead, used for loop-detection.
- An **Accept** that lists what type of content is acceptable.
- A **Via**, the address at which responses should be directed (usually a host name, IP, and port combination, along with a branch to identify the transaction).
- A **Contact**, the address at which the SIP user can be directly reached. Lotus Sametime only uses SIP for server to server, so this will always contain a SIP Connector address.
- A unique **Call-ID** used to track the SIP session.
- A **Content-Length** to show how much data is contained in the SIP packet.

Overview of SIMPLE

What is SIMPLE?

SIMPLE is an extension to the SIP protocol. It is standardized and defined in RFC 3428, and its purpose is to define the MESSAGE method to leverage text-message sending capabilities. Since the MESSAGE request is an extension to SIP, it inherits all the request routing and security features of that protocol. Lotus Sametime uses the MESSAGE method to send messages to users in external SIP communities.

When to use SIP

A Lotus Sametime community can be established between multiple servers residing in different geographical areas, behind firewalls and Network Address Translation (NAT) devices, or even across Lotus Sametime servers having different Lotus Domino domains. However, in order for instant messaging and presence awareness to work correctly, both servers must share the same directory. Lotus Sametime SIP is used when two or more organizations do not wish to share directory information with one another, but would still like to have presence awareness and instant messaging capabilities between their organizations.

How IBM Lotus Sametime Uses SIP

The Lotus Sametime SIP architecture consists of two main components: a Lotus Sametime server, and a Lotus Sametime SIP Connector server. The Sametime server, by default, runs a service called StGateway and is also referred to as the “Lotus Sametime Gateway” or “Gateway” server. The Lotus Sametime SIP Connector is also referred to as the “Connector”.

For the purpose of this article, we define a Lotus Sametime SIP Community as a combination of the Gateway and Connector servers on one side of a SIP connection. A single Lotus Sametime SIP Community could also be a combination of clustered or distributed Sametime servers within an organization, along with the SIP Connector or Connectors that belong within that organization. An “External Community” is a SIP Community to which your SIP Connector will be establishing a connection. The “External Community” could be within your organization, as in the case of organizational divisions or subsidiaries, or it could be another company.

Requirements for Lotus Sametime SIP

1. The Lotus Sametime Gateway and Connector cannot be installed on the same system. The Connector software must run on a separate server.

2. Users wishing to use SIP must have an email address defined. The user can be in a Lotus Domino Directory or an LDAP Directory. SIP uses the email address to resolve user names. Once the user is resolved, you may give the user any name of your choosing on your client’s buddy list.

3. The SIP Connector is recommended to run an external server in the DMZ, so that it can establish connections with other SIP communities while keeping the Lotus Sametime server protected behind a firewall. However, you may still run the SIP Connector behind a firewall, as long as the required ports are open.

4. A NAT device allows devices behind it to have one set of IP addresses, while network devices outside the NAT device see a different set of IP address. Firewall NAT’ing is currently supported with the Lotus Sametime SIP architecture. However, PAT (Port Address Translation) is not supported. For more information on PAT and why it is not supported, see Appendix A.
5. The Lotus Sametime SIP Connector, by default, will start automatically with the operating system unless configured otherwise.

6. The SIP Connector will not make any connections to any other SIP servers until a client in the Lotus Sametime environment requests a lookup to an external (SIP) user. A NetStat will simply show a port 5060 (from StSIPConnector.exe) in a LISTEN state until a connection is actually needed.

7. The SIP Connector (StSIPConnector.exe) will attempt to connect to port 1516 on the Lotus Sametime server. If the SIP Connector cannot connect to the Lotus Sametime server when it starts, it will automatically terminate with the following message in the SIP Sametime.log file:

   I SIPConnect [date/time] Setting exception handler for service SIPConnector
   I SIPConnect [date/time] Started, version 6.51.0.3
   I SIPConnect [date/time] Terminated

8. The communication between the Connector and Gateway is bidirectional, but the connection is only one way. Therefore, only one connection needs to be established from the Connector to the Gateway. Port 1516 does not need to be open to the SIP Connector from the Lotus Sametime server, but port 1516 does need to be open to the Lotus Sametime server from the SIP Connector.

9. Only one connection is needed between two SIP Connectors. Once a connection is established one way on port 5060, there will not be another port 5060 connection to the other server. The first server to request an external community lookup establishes the connection. The connection will be maintained until one of the SIP servers is restarted.

**Ports used**

The SIP Connector will attempt to connect to the Lotus Sametime Gateway server on port 1516. Once this TCP connection is established, the Connector will be sent its configuration information, such as what ports to use in making outbound connections and which servers to attempt to establish connections. By default, there will be an attempt to make outbound connections over port 5060 for unencrypted connections, and port 5061 for TLS encrypted connections, to other Lotus Sametime SIP Communities defined in the StConfig.nsf database.

**StConfig.nsf documents modified**

Configuring the Lotus Sametime Gateway and Connector servers consists of adding and modifying Configuration documents to the Lotus Sametime StConfig.nsf database, and modifying the Sametime.ini files on both the Gateway and the Connector. You must modify the StConfig.nsf database manually, using either an IBM Lotus Notes® or Lotus Domino Administrator client.

**Introduction to a Basic Lotus Sametime SIP configuration**

Example 1: Basic Configuration

In this example, as illustrated in figure 1, there are two external communities, Alphatest and Bravotest. Each domain contains one Lotus Sametime server and one Connector. Each domain maintains awareness with only one other external community. When a user from either community adds an external person to their contact list, a SUBSCRIBE request is sent to the
Gateway process on their home Lotus Sametime server. The Gateway then forwards this request to the Connector.

**Figure 1. Basic SIP Configuration**

![Basic SIP Configuration Diagram]

**Introduction to Encrypting SIP Connections**

What does IBM Lotus Sametime use to encrypt connections?

The IBM Global Security Kit (GSKit) is the component used by Lotus Sametime to encrypt connections with other SIP communities. The SIP Connector will use TLS v1.0 encryption to secure SIP connections, where the RSA public-key encryption algorithm is used for key exchange. The cipher and message digest algorithms are negotiated during the initial handshake, but the server will attempt to select the cipher suite in this preferred order:

- 128-bit RC4 encryption with SHA-1 message authentication
- 128-bit RC4 encryption with MD5 message authentication
- 168-bit Triple DES encryption with SHA-1 message authentication
- 56-bit RC4 encryption with SHA-1 message authentication
- 56-bit DES encryption with SHA-1 message authentication
- 40-bit RC4 encryption with MD5 message authentication
- 40-bit RC2 encryption with MD5 message authentication
- 56-bit DES encryption with SHA-1 message authentication

If both SIP communities are Lotus Sametime SIP communities, the encryption will be 128-bit RC4 encryption with SHA-1 message authentication and RSA key exchange.

**What is a client certificate?**

A client certificate authenticates a SIP server to another SIP server. Suppose you have organizations A and B. Although it is sufficient for A to use B’s certificate to encrypt messages, B has no way of verifying A’s authenticity. It could be that organization C also has a copy of B’s certificate and is using it to gain access into B’s SIP environment. Using a client certificate is a means by which B is able to verify that organization A is actually sending the encrypted messages and not server C, similar to the idea of digital signatures.

**Why encrypt SIP Conversations?**

The unencrypted SIP network packets contain, in plain text, information that could be used maliciously by a user attempting to gain access to confidential information. This information
includes email addresses of users in your environment and all instant messaging conversations held between the users in the connected SIP communities. Most organizations will use SIP to connect two communities over the Internet. Because of this, it is important that both organizations agree to use the encryption feature available in the Connector software to help secure the connection and prevent eavesdropping. For instructions on setting up TLS to encrypt SIP traffic, refer to the section below titled “Installing the Encryption Components”.

Configuring your Lotus Sametime SIP Gateway and Connector

Introduction

Lotus Sametime SIP should always be configured in a series of stages to reduce the number of variables involved, should there be an issue. It is highly recommended that an administrator first set up a single Gateway server to connect to a single Connector, and avoid introducing SIP or Lotus Sametime clustering, TLS encryption, and other SIP features all at once. If possible, try to minimize the amount of firewall devices between the Gateway and Connector and between your Connector and the other organization to which you are attempting to connect. Once a basic configuration is tested, then add in the firewall, followed by clustering, distribution, and then finally TLS encryption.

Let’s begin by installing the SIP Connector and then configuring the Lotus Sametime server (the Gateway) for one SIP community. These same instructions will then be used to set up another separate community’s SIP Connector and Lotus Sametime server. Once the two SIP communities have been configured, we will then test our setup to verify that they are able to communicate.

Setting up a basic SIP configuration

Requirements

- **Lotus Sametime and SIP Connector must be installed on a separate Microsoft® Windows® operating system from the Lotus Sametime Gateway server:**

  The SIP Connector and Lotus Sametime Gateway server cannot run within the same operating system environment.

- **Email addresses are required for all SIP users:**

  While Lotus Sametime instant messaging and Web logins can use any number of formats, including email address, surname, full name, short name, or a unique ID, the SIP protocol relies solely on the email address for person lookups. Because SIP uses email addresses to resolve names, when using Lotus Domino directory authentication, you must ensure that a user’s Person document include an Internet address if an external SIP user wants to add that person to their list.

- **Considerations for LDAP directories:**

  When a Lotus Sametime server is pointing to LDAP for authentication, you must tell Sametime which LDAP attribute contains the email address format for your users:

  1. On the Lotus Sametime server using LDAP for lookups, open the StConfig.nsf database.
2. Open the LDAPServer document. Find the field labeled “Attribute of a person entry that defines the person's e-mail address”. By default, this field is blank because it is only used by Lotus Sametime when processing SIP requests.
3. Populate this field with the LDAP attribute containing the email address format. Most LDAP implementations, including Netscape, iPlanet, Sun Open Net Environment (ONE), Microsoft Exchange 5.5, Lotus Domino, and IBM SecureWay®, use mail as the attribute to define the email address. Microsoft Active Directory typically uses userPrincipalName.

**Finding the SIP Connector Software**

The Lotus Sametime SIP software is found in the Lotus Sametime Microsoft Windows Components package, which can be downloaded from the IBM Passport Advantage Web site. The part number for the SIP Connector software for Lotus Sametime 6.5.1 is C57LAML. Once you extract the files, you will need to run the setup.exe program for the SIP connector in the folder called SIPConnector.

**Installing the SIP Connector**

1. Run the setup.exe for the SIP Connector software in the SIPConnector directory.
2. The installation will ask for the “Name of IP Address of the Sametime Server”. Add either the fully qualified hostname or IP address of your Lotus Sametime server.
3. The installation will also ask for the “Name for the SIP Connector”. Add an arbitrary SIP Connector name. The host name (minus the DNS suffix) is recommended by some installation guides, but it is not required.
4. The above two parameters affect only the Sametime.ini file. This is what the Sametime.ini looks like on a SIP Connector once the setup is complete:

   [ExternalCommunity]
   ConnectorName=BravoSip
   [Connectivity]
   VPS_HOST=sip.bravo.com

   The VPS_HOST parameter points to the Lotus Sametime server (the Gateway) for the SIP Connector.

**Setting up the SIP Gateway**

First, add the following three documents to the StConfig.nsf:

1. **CommunityConnector**
   - **Connector Name**: SIP Connector name, from Step 3 above, for example, BravoSip
   - **IP**: IP or host name of Connector, for example, sip.bravo.com
   - **Port**: 5060
   - **TLS IP**: blank
   - **TLS Port**: blank
   - **Supported Communities**: Add the exact name corresponding to the Community Name in each Extern Community document. You can list multiple entries separated by a semicolon.

2. **CommunityGateway**:
   - **Support external communities**: True (set to False to turn off all SIP connectivity, or just stop the ST SIP Connector service on the SIP Connector)
   - **Convert ID**: True
3. **Extern Community:**
   - **Community Name:** Name of External Community
     
     Note: This parameter is only used to match up with the Supported Communities field on the CommunityConnector document. It is case sensitive, so they must match exactly.
     
   - **Domains:** Email address domains, comma separated, no spaces allowed.
     
     a. **alpha.com** would allow local users to communicate with external users with an email address `sample@alpha.com` but not `sample2@ext.alpha.com`.
     
     b. ***.alpha.com** would allow local users to communicate with users with both email addresses `sample@alpha.com` and `sample2@ext.alpha.com`.
     
     c. You may add as many email domains as you would like to this line.
     
     For example: `alpha.com,*.bravo.com,ext.charlie.com,*delta7.com`
     
     NOTE: In this example, place no spaces between the domain names, either before or after the commas. If you have any spaces or use any other characters, the configuration will not be read, and you will not be able to see any external users online. It is also important to maintain consistency with respect to upper and lower case. It is recommended that the domains are listed in all lower-case letters, and that your directory also contains all lower-case letters in email addresses.
     
     - **DNS:** DNS name or IP address of the other organization’s SIP connector
     
     - **Port:** Port used for listening connections on the other organization’s SIP connector, normally configured to 5060
     
     - **Encryption:** Disable
     
     - **Certificate distinguish name:** blank
     
     Then, modify the CommunityConnectivity document to include the SIP Connector’s hostname or IP address in the Community Trusted IPS list.
     
     Finally, restart the Lotus Sametime server, and then restart the SIP server after the Sametime server has fully initialized.

**Important Notes:**

- Each external SIP Community must have its own Extern Community document on your Lotus Sametime server that connects to your SIP Connector. You will have as many Extern Community documents as you do external SIP communities.
- The Connector Name in the CommunityConnector document matches up with the SIP Connector name in the SIP Connector installation. These are the only places these names appear or are used.
- The only place the Community Name is used other than the Extern Community document is the CommunityConnector document. Removing and adding these names from the CommunityConnector document is how you can quickly add and remove external communities to your Lotus Sametime environment.
- You may be able to restart just the SIP Connector for any changes made to the three added documents, CommunityConnector, CommunityGateway, and Extern Community.

Now that one side of the setup is complete, you need to install and configure the SIP Connector for the second community, using the same instructions above.
Testing your setup

Restart both Lotus Sametime servers. Once the Sametime servers have completely come up, start both SIP Connectors.

Adding SIP users to your Lotus Sametime buddy list

- Open the Connect client and log into the Lotus Sametime server.
- Select People -> Add.
- In the “Add Person or Group” window, select “External” from the Community drop-down box (the default selection here is “Sametime”).
- Type in the Email Address of the person you would like to add from the other SIP Community and, optionally, give this person a Nickname. If you do not give a nickname to the user, they will be displayed in your buddy list by their email address. You may give a nickname to a user at a later time if you choose.
- If SIP is configured correctly and the external user is online, the buddy will appear with a globe icon beside their name indicating both their online status (Available, Away, or DND) and that they are an External user. Figure 2 shows what the online status looks like for the external users in the alpha, long.alpha.com, and alpha2.com email domains.

![Figure 2: Contact list with SIP presence awareness](image)

Notes about External Users:

- Note that, when you select the “External” from the Community drop-down box, you may add names that don’t necessarily exist in any external domain. You will not receive any type of warning indicating that a user doesn’t exist.
- If both users have a microphone and/or video camera, then you may hold an audio and/or video conference with an External user. You may not use whiteboard or screen-sharing with External users, nor can you send files or announcements. Connections will be made from the Lotus Sametime client to the Sametime server.
- If the users in each community are online, then they will be able to see one another. Awareness is a symmetric relationship; it is not possible to add an External user to your buddy list without them also being able to see your online status, unless they are blocked in your Privacy list.
Installing the Encryption Components

Requirements

The following are requirements for setting up TLS encryption for Lotus Sametime:

- The GSKit must be installed on the Connector. The GSKit contains libraries that will be referenced by the Connector when establishing an encrypted connection to another SIP Community.
- A key database must be created and stored on the Connector.
- Optionally, you may import a client certificate into the key database so that the other SIP community can not only encrypt the connection but also verify your authenticity.

Section 1: Configuring the Connector

The following example assumes you have a Lotus Sametime Standalone Connector installed on the system in the C:\Program Files\Lotus\Sametime folder and a Lotus Sametime 6.5.1 FP1 or later server.

This first section involves adding the IBM Java™ Virtual Machine (JVM) to the Connector machine and configuring the JVM for the GSKit application setup steps to follow in Section 2. The IBM JVM is not installed by default.

1. Go to the IBM Lotus Sametime server and make a copy of the "ibm-jre" directory. The default location is

   C:\lotus\domino\ibm-jre

2. Copy the ibm-jre directory from the Lotus Sametime server to the Sametime Connector installation directory on the Connector machine. For example,

   C:\Program Files\lotus\sametime\ibm-jre

3. Delete the gskkm.jar file from the \ibm-jre\jre\lib\ext folder. For example,

   C:\Program Files\lotus\sametime\ibm-jre\jre\lib\ext\gskkm.jar

4. Add a new line to the java.security file found within \jre\lib\security:
   a. Open the java.security file in C:\Program Files\lotus\sametime\ibm-jre\jre\lib\security\java.security.
   b. Find the following section in the java.security file starting with "# List of providers and their preference orders".
   c. Add "security.provider.5=com.ibm.spi.IBMCMSPProvider" to the last line of this section. This line is case sensitive. If the provider section of the java.security file has more or less than four previous lines, change the number of the provider to be the last number in the list of security providers (see figure 3).

Figure 3. Configuring security providers

```bash
# List of providers and their preference orders (see above):
#security.provider.1=com.ibm.jsse.IBMJSSEProvider
security.provider.2=com.ibm.crypto.provider.IBMJCE
security.provider.3=com.ibm.security.jgss.IBMJGSSProvider
security.provider.4=com.ibm.security.cert.IBMCertPath
security.provider.5=com.ibm.spi.IBMCMSPProvider
```
d. Save and close the java.security file

5. Create a new system variable named JAVA_HOME on the operating system of the connector machine:
   a. Right-click My Computer, select Properties, click the Advanced tab and then the Environment Variables button.
   b. Under the System Variables section, click the “New” button.
   c. Set the value of this variable to: `C:\Program Files\lotus\sametime\ibm-jre\jre` (see figure 4).

Figure 4. Configuring environmental variables

Section 2: Installing GSKit6 and Creating the Key Database

In this section we will install the GSKit6 application on the Connector machine. GSKit will use the IBM JVM to create a keyfile database to store the certificates.

1. Copy the GKit directory to the c:\temp directory on the Connector machine. This file is included on Lotus Sametime 6.5.1 Win32 CD2 or the Sametime 7 Component CD.
2. Open a command prompt on the connector machine and go to c:\temp\gskit.
3. Enter “setup.exe GSKit -s f1setup.iss”. For example:

   `C:\temp\Gskit\> setup.exe GSKit -s f1setup.iss`

4. This will perform a silent install using the configuration variables included in the setup.iss file. To ensure the install was successful, verify the gsk6 directory was created in

   `C:\Program Files\ibm\gsk6 [the default installation path]`

5. Launch the GSKit program by starting the GSK6km.exe application located in

   `C:\Program Files\ibm\gsk6\bin.`

6. Click the paper icon (or select Key Database File -> New, from the menu) to create a new database.
7. Under **Key database type**, select CMS. If the CMS option does not exist, verify that Steps 3 and 4 were completed successfully in Section 1.
8. Name the file "key.kdb".
9. In the Location field, point to the Sametime directory and click OK (see figure 5).

**Figure 5. Creating the key database**

![Creating the key database](image)

10. Provide a password and select the option to stash the password to a file. It is not recommended to set an expiration time (see figure 6).

**Figure 6. Saving and stashing a password**

![Saving and stashing a password](image)

11. Click OK; a message box displays telling you where the stash file was saved.

**Figure 7. Confirmation of stash file**

![Confirmation of stash file](image)

**Note**: If you receive the message "Error occurred while inserting keys into database" after clicking the OK button, go to Help/About and verify GSKit is at least version 6.0.5.41.

**Section 3: Generating the Server Certificate Request**
To operate with a TLS connection, the SIP Connector must have access to two certificates. The key database created in Section 2 will hold the certificates used for the TLS handshake.

1. If you are setting up a test environment and want to make use of a self-signed certificate, refer to Section 3a below.
2. The following Certificates are needed:
   a. SSL signer (or "trusted root") certificate signed by a specific Certification Authority (CA), such as VeriSign.
   b. Server certificate signed by one of the trusted Certificate Authorities. A trusted Certificate Authority is considered to be any CA that is listed in the Signer Certificate section of the key database.
   c. In the Key Database Content drop-down list, select "Signer Certificates" to display the list of CA trusted root certificates provided by default (see figure 8).

**Figure 8. Trusted certificate authorities**

<table>
<thead>
<tr>
<th>Certificate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Domain2</td>
</tr>
<tr>
<td>VeriSign Class 2 OwnSite Individual CA</td>
</tr>
<tr>
<td>VeriSign International Server CA - Class 3</td>
</tr>
<tr>
<td>VeriSign Class 3 Public Primary Certification Authority - 62</td>
</tr>
<tr>
<td>VeriSign Class 2 Public Primary Certification Authority - 62</td>
</tr>
<tr>
<td>VeriSign Class 1 Public Primary Certification Authority - 62</td>
</tr>
<tr>
<td>VeriSign Class 1 CA Individual Subscriber-Persona Not Validated</td>
</tr>
<tr>
<td>Thawte Personal Premium CA</td>
</tr>
<tr>
<td>Thawte Personal Freemail CA</td>
</tr>
<tr>
<td>Thawte Personal Basic CA</td>
</tr>
<tr>
<td>Thawte Premium Server CA</td>
</tr>
<tr>
<td>Thawte Server CA</td>
</tr>
<tr>
<td>RSA Secure Server Certification Authority</td>
</tr>
<tr>
<td>VeriSign Class 1 Public Primary Certification Authority</td>
</tr>
<tr>
<td>VeriSign Class 2 Public Primary Certification Authority</td>
</tr>
<tr>
<td>VeriSign Class 3 Public Primary Certification Authority</td>
</tr>
</tbody>
</table>

A signer certificate is from a certification authority (CA) or from another web site.

d. If you are using one of these specific Certificate Authorities, this step is complete. If you are using another CA, add the Certificate Authority to the key database by selecting the add button and following the steps.

3. Now that the key database has been created and the signer (or "trusted root") certificate has been added, the next step is to generate a Personal Certificate request. This certificate contains the connector server’s unique fingerprint and needs to be sent to a Certificate Authority to be digitally signed:
   a. Under the Key Database Content section, choose the Personal Certificate Requests option from the drop-down menu.
   b. Click the New button to create the request
   c. Fill in the mandatory fields and click the OK button (see figure 9).

**Figure 9. Create New Key and Certificate Request dialog box**
d. Once the .arm file is created, you must go to the Web site of your chosen Certificate Authority and submit a Certificate Signing Request (CSR). Follow the specified directions on the CA Web site. This usually involves copying and pasting the contents of the .arm file.

**Section 3a: Creating a Self-Signed Certificate for Testing (Optional)**

If configuring TLS in a test environment, follow these steps to create the self-signed certificate (see figure 10):

1. Select Create from the main menu and select New Self-Signed Certificate.
2. Provide a name for the key in the Key Label field (this setting is arbitrary).
3. In the Common Name field, add the Fully Qualified Host Name of the Connector server.
4. Provide the country or region and click OK.
   a. NOTE: Skip section 4 and go to section 5, if you have created a self signed certificate.

*Figure 10. Create New Self-Signed Certificate Request dialog box*
Section 4: Import Server Certificate into Key Database

After receiving the signed certificate from the CA (or self-signed CA in a test environment), you must import the file into the key database on the SIP Connector machine:

1. Switch the Key Database Content drop-down list to Personal Certificates.
2. Select the Receive option and fill in the following fields (see figure 11):
   a. **Data type**: Use the default “Base64-encoded ASCII data”.
   b. **Certificate file name**: Enter the name of the file that contains your signed certificate.
   c. **Location**: Enter the drive and directory where your certificate is located.

Figure 11. Receive Certificate from a File dialog box

---

Section 5: Exporting an .arm File

Now that the key database has been created and the connector can access the two required certificates, it is time to export an .arm file from your domain and provide it to the administrator for the domain for which you wish to set up a TLS connection. This section explains how to use the server certificate from Section 3 or 3a to create and export an .arm file. The .arm file from the Connector machine in Domain 1 will be imported into the key database on the Connector machine in Domain 2.
Note: This step is not necessary if the CA signer of your server certificate is already trusted, as defined in Section 3.

1. Select the Personal Certificates option under the Key Database Content section.
2. Highlight the personal certificate and select “Extract Certificate”.
3. Leave the default “Data-type to Base64-encoded ASCII data” (see figure 12).
4. Provide an arbitrary Certificate file name and location.
5. Send this file to the Administrator of Domain2; this will be the file you will be exchanging with the external SIP Community. In turn, you will also be receiving the foreign community’s *.arm file.

Figure 12. Extract Certificate to a File dialog box

Section 6: Importing the Root Certificate from the External Community

In this section we will import the .arm file from Domain 2 into the key database of Domain 1 so that the two servers are cross-certified.

Note: This step is not necessary if the CA signer of the server certificate from the external community is already trusted as defined in Section 3.

1. Open the GSKit application and select the Signer Certificates option under the Key Database Content section.
2. Click the Add button.
3. Enter the path and the name for the Certificate file received from Domain 2 and click OK (see figure 13).

Figure 13. Add CA’s Certificate from a File dialog box

4. When prompted to enter a Label, enter an arbitrary name you’d like to call this certificate. For example, Domain1 Trusted CA.

Section 7: Configuring Documents on IBM Lotus Sametime to Enable TLS

1. Open the Stconfig.nsf on the Lotus Sametime server.
2. Open the “Community Connector” document and add the IP address of your connector server to the TLS IP field (this will typically be the same IP address listed in the IP field on the same document).
3. Enter 5061 into the TLS Port field.
4. Save and close the CommunityConnector document.
5. Open the External Community document for this domain.
6. Change the Port field to 5061, and set the Encryption field to “Mandatory”.
7. Save and close the External Community document. Domain 1 is now configured for TLS.
8. Ensure that the administrator for Domain 2 has followed the same steps and has successfully imported the .arm file from Domain 2, so that the servers are cross-certified, and then restart both connectors.

**Introduction to More Advanced SIP Configurations**

To fully understand how to configure the SIP Connector and Gateway within a specific environment, it is important to understand some fundamental principles.

**Fundamental Principles**

1. Each Gateway is aware of every Connector in the Sametime Community and what domains these Connectors service.
2. Each Connector can service multiple external communities.
3. The Connector cannot be configured to route requests to more than one IP address per external community.
4. The Connector cannot be configured to send external community requests from one IP and receive requests from this external community from another IP.

Note that Principles 3 and 4 are important to understanding failover and will be discussed more in Examples 4 and 5 below.

**Example 2: Basic Configuration with Multi-Server Environment**

In Figure 14, you can see that Domain 1 is a distributed environment with multiple Lotus Sametime servers, but only one Connector. In this configuration, users on Sametime Server A will send requests to the Gateway on Server A, and users logged into Server B will route requests through the Gateway on Server B.

**Figure 14. Basic SIP configuration with multi-server environment**

It is important to understand that only one server in the Lotus Sametime community needs to contain the External Community, Community Connector, and Community Gateway documents in the StConfig.nsf. In figure 14, Server B would contain only the Gateway document, while Server A would contain all three configuration documents. The Connector’s Sametime.ini would be configured to connect to Server A to receive its configuration information. The Community Gateway documents on Server B allow users logged into Server B to access the Gateway.

**Example 3: Single Connector Binding to Multiple External Communities**
In figure 15, Domain 1 is connecting to multiple external communities. The only configuration change that Domain 1 would need in order to add an additional external community is an extra External Community document in the StConfig.nsf. Testing in the lab has confirmed that each Connector is capable of hosting up to 10 external communities.

Figure 15. Single connector binding to multiple external communities

Example 4: Single Connector Hosting a Global Clustered Lotus Sametime Environment

In figure 16, Domain 1 has two Sametime clusters with two Sametime servers in each cluster. The Connector and Cluster A are located in North America, and Cluster B is located in Europe. If a load balancer is added between the Connector and Sametime Cluster A, failover capability is provided, ensuring that the Connector can always contact one of the Sametime servers to read the three SIP configuration documents. In this case, both Sametime servers in Cluster A would contain the three configuration documents.

Figure 16. Single connector hosting a global clustered Sametime environment

Remember that the Connector is dependent on a Lotus Sametime server to receive its configuration information, and only one IP can be specified in the Connector’s Sametime.ini. The Sametime.ini on the connector would point to the load balancer; however, it isn’t necessary to use a load balancer, and this is an optional configuration.

In this global Lotus Sametime example, it would be more efficient to add a second Connector to route the European office through a connector in Europe rather than having all European requests
travel from the European Gateway to the Connector in North America. Ideally, the second connector could also provide failover for the Community in case the first Connector was down.

Example 5: Can Lotus Sametime SIP be Configured for Failover?

To answer the failover question, you must understand the various configuration choices that could be used to achieve failover. Examples 5, 5a, and 5b explore these configuration options and their outcomes.

**Option 1: Adding a Second Connector**

Let’s continue with the problems faced in Example 4 and say that Domain 1 adds the second connector to allow European personnel to route through a European Connector while North American personnel continue to route through the North American Connector (see figure 17).

**Figure 17. SIP failover scenario with dual connectors**

**Configuration Changes**

In this example, Domain 1 would need to add a second Community Connector document for the second connector. Domain 2 would need to add an additional External Community document pointing to the second Connector that Domain 1 added.

**Outcome of Adding a Second Connector**

The Connector in Domain 2 will read the two External Community documents in its STConfig.nsf and bind to whichever Connector in Domain 1 has the highest last octet. For example, the last octet in 207.10.10.2 is 2. So in figure 17, Domain 2 would bind to 207.10.10.3.

Since the EMEA Connector’s IP has the highest last octet, the Connector in Domain 1 will only respond and send requests to this IP address. It will not route requests to the North American Connector, even though it is able to bind to and receive requests from it. This configuration will not work in terms of failover as the SIP Connector in Domain 1 cannot be configured to route requests to more than one IP address per external community (Principle 3).

**Option 2: Adding a Load Balancer in Addition to the Second Connector**

Since the Connector in Domain 1 cannot be configured to route requests to more than one IP address per External Community, would it be possible to put a load balancer between the two Domains, so that Domain 1 simply pointed to the IP of the load balancer (see figure 18)?

**Figure 18: SIP failover scenario with dual connectors and load balancer**
Configuration Changes
Domain 1 would be configured identically to Example 5, Option 1, and have two Community Connector documents. Domain 2 would remove the second External Community document and simply change the DNS field in the existing External Community document to point to the IP of Domain 1’s load balancer.

Outcome of Adding a Load Balancer
Using this configuration, Domain 2 will route requests to Domain 1 on IP 207.10.10.1, but it will be receiving requests directly from each of Domain 1’s Connectors (207.10.10.2 or 207.10.10.3). This process flow breaks Principle 4 in that a Connector cannot be configured to send External Community requests from one IP and receive requests from another IP. In order to achieve failover, you must also add Bidirectional NAT in addition to the second load balancer and Connector.

Option 3: Adding Bidirectional NAT

Bidirectional NAT can be configured so that it modifies the Source IP of all outgoing traffic from either of the SIP Connectors. In that way, they always appear to be coming from the same IP to which the Connector in Domain 2 is sending requests (see figure 19).

Figure 19. SIP failover scenario with dual connectors, load balancer, and NAT

Configuration Changes
The only change required for Domain 1 is to configure NAT to modify the Source of all outgoing IPs to 207.10.10.1. Domain 2 will not require any changes.

Outcome of Adding Bidirectional NAT, Dual Connectors, and a Load Balancer
With this configuration, if one of the SIP Connectors in Domain 1 goes down, requests will automatically route from the Gateway to the second Connector and over to Domain 2. Domain 2 will route requests to the NAT device, load balancer, and then to one of the two Connectors.

Example 6: Using Multiple Connectors without Failover

It is possible to add multiple SIP Connectors to a Sametime Community in order to bind to multiple external communities. In this example, the SIP Connector in EMEA is binding to Domain 3, while the Connector in the US is binding to Domain 2 (see figure 20).

Figure 20. Using multiple connectors to connect to multiple external communities

Let’s take a look at how the process will flow if a user in Domain1 adds jsmith@domain3.com to their contact list. The subscribe request will route through the user’s home Lotus Sametime server’s Gateway and then to the EMEA SIP Connector. Users who add jdoe@domain2.com will route through their home server’s Gateway and then to the North American SIP Connector. Adding more Connectors is not necessary to connect to multiple communities. However, it can be beneficial for troubleshooting and scalability to have one external community per connector.

Troubleshooting Lotus Sametime SIP

An essential component in configuring SIP in your environment is to understand how to apply debug to the Connector and Gateway processes to see what is actually happening. To troubleshoot problems successfully, it is often necessary to trace an individual SIP transaction between servers and domains to learn where the transaction is failing and why. Debugging SIP problems can be complicated, but this section provides a general guide to applying debug to the Connector and Gateway and then tracking individual SIP transactions between servers and domains.

Debugging the Gateway on the IBM Lotus Sametime Server

The following parameters should be added to the [Debug] section of the Sametime.ini:

```
[Debug]
GATEWAY_PLACE_DEBUG =1
BL_GW_CHANNEL =1
GATEWAY_DEBUG =1
GATEWAY_CONFIG_DEBUG =1
GATEWAY_TO_GATEWAY =1
MESSAGE_DEBUG =1
```
The output will be written to the StGateway_<date>_<time>.txt file in the trace directory (Lotus\Domino\Trace). A new file will be created each time the Gateway service starts.

**Debugging the SIP Connector Machine**

These parameters should be added to the [Debug] section of the Sametime.ini:

```
[Debug]
UCM_DEBUG=1,
UCM_KERNEL=1,
UCM_TCP_DEBUG=1,
UCM_TLS_DEBUG=1,
SIP_CONNECTOR=1
MESSAGE_DEBUG = 1
GATEWAY_CHANNEL = 1
SIP_MESSAGE_DEBUG = 1
```

The output will be written to the StSIPConnector_<date>_<time>.txt file in the trace directory (\Sametime\trace).

Other SIP Connector Files are as follows:

- **Inbuf.txt**: Located in the Sametime directory, it contains every incoming transaction that is sent to it from another SIP Connector.
- **Sametime.log**: Also located in the Sametime directory, it contains a record of Connector restarts and errors.
- **Trace directory**: Contains the individual StSIPConnector_<date>_<time>.txt files.

**Following a Transaction**

To track an individual transaction, you must first start with the Gateway debug from the Domain that generated the request. Remember that the process flow for a transaction goes from the Gateway on a user’s home server to the Connector, and then to the other Domain’s Connector and Gateway. Figure 21 illustrates the flow of a Lotus Sametime SIP conversation.

**Figure 21. Example SIP transaction**
As we discussed earlier, there are three methods used to provide presence awareness and instant messaging. SUBSCRIBE and NOTIFY are used for presence awareness, and MESSAGE is used to send instant messages. Figure 21 shows a sequence of events in which each of these methods are used. Let’s take a look at the first request:

SUBSCRIBE sip:bravotest5@217.10.10.2 SIP/2.0
To: <sip:bravotest5@217.10.10.2>
From: <sip:alphatest5@194.10.10.1>;tag=34594
Call-ID: 13_09357114.34595

Here the SUBSCRIBE method is used along with a “To” and “From”. The request is generated after user alphatest5 adds bravotest5 to his buddy list. The IP address that is appended to alphatest5’s email address is the address of the SIP connector for Domain 1 where this user resides. Likewise, the IP address for bravotest5 is the IP address of Domain 2’s Connector. The last piece of data in the SUBSCRIBE request is the Call-ID, which is important and can be used like a transaction ID to trace this transaction across all servers involved.

The first alpha-numeric characters that begin the string and precede the underscore character are used by the Connector to differentiate the Gateway that created the request. For example, in the Call-ID above, 13 denotes the ID of the Gateway. If a Lotus Sametime community has more than one Lotus Sametime server, this unique gateway identifier will be used to route the responses back to the correct Gateway. The rest of the string is used as a unique identifier for the transaction.

The second operation in this sequence is Domain 2 responding to the request by issuing a 202 Accepted response, followed by the NOTIFY method. Domain 1 then responds with a 200 OK to inform Domain 2 that the message was received. At this point, alphatest5 will show presence awareness for bravotest5 and will see this user online.
Getting the presence awareness between external communities is probably the most common problem faced. If presence awareness is working, then there will generally not be any problems delivering instant messages. To troubleshoot specific awareness problems, it is necessary to apply the full set of diagnostics described above and collect all the relevant trace files from both external communities.

The next step would be to find the original SUBSCRIBE request, follow the flow using the Call_ID for this transaction, and then make sure that the Connector on Domain 2 received the request. If the request is not found, check the inbuf.txt file on the Connector in Domain 2. If the SUBSCRIBE request is not there, the packets did not make it. If Domain 2 received the request, but found a problem with it, there would be an error reported. Whatever the problem may be, you can use the various output files to determine the error message or breakpoint in the transaction.

The rest of the transaction follows the original, but alphatest5 has sent bravotest5 an instant message:

```
MESSAGE sip:bravotest5@217.10.10.2 SIP/2.0
To: <sip:bravotest5@bravo.com>
From: <sip:alphatest5@194.10.10.1>
Call-ID: 13_09357114.34589
Hello!
```

In this request, the MESSAGE method (SIMPLE protocol) is being used. Again, the “To” and “From” elements are there along with the Call-ID. Notice the last section of the Call-ID has been modified slightly to denote the unique identifier for messaging transactions. Again we see Domain 2 responding with a 200 OK response using the same Call-ID. To troubleshoot Lotus Sametime SIP problems successfully, you must fully understand the process flow between the two communities and then apply debug to determine the break point or error message. For up-to-date information on understanding the error codes, refer to Lotus Technote #1232976 “Understanding the error codes associated with Sametime SIP problems”.

Summary

In this article, you learned how using SIP with IBM Lotus Sametime can be effective in connecting to other external Sametime communities. You also learned how to install, troubleshoot, and configure a security-rich SIP connection within a simple or highly complex Lotus Sametime environment. The SIP protocol will continue to increase in importance and popularity as the protocol of choice for a variety of applications. IBM currently plans to use SIP in its next-generation Real-time Collaboration (RTC) Gateway that is expected to provide connectivity from Lotus Sametime to other SIP-compliant applications, including IBM Workplace software, AOL Instant Messenger, and Yahoo! Messenger. The future of Lotus Sametime and SIP is bright. Use this article to get started now!

Resources


IBM Lotus Sametime Product Documentation site: http://www-10.lotus.com/ldd/notesa.nsfl/find/sametime

IETF Request for Comments: http://www.ietf.org/rfc.html
developerWorks article “Taking a tour of the new features and technology in IBM Lotus Sametime 7.5”:  

About the authors

Todd Page is a Software Engineer and Technical Team Lead for IBM Lotus Sametime Support in Austin, TX. He has been with IBM since 2002 and has been fortunate enough to spend several months of his tenure traveling through the U.S., Europe, South America, and the Middle East meeting with Sametime customers. Todd also co-authored the developerWorks article “Taking a tour of the new features and technology in IBM Lotus Sametime 7.5”.

Kendell Zahn is a Computer Science graduate of the University of Texas at Austin. He has worked as a Software Engineer with the IBM Lotus Sametime Support team in Austin, TX, since 2004 and is currently the Product Area Expert for SIP.

Acknowledgements

A special thanks to the following people for their help with this document:

- Kirti Gani
- Lilach Ofek
- Ran Bar-Lavie
- Andy Yiu
- David Tull
- Larry Wolfe

Appendix A: Known Issues

**IBM Lotus Sametime and Port Address Translation (PAT)**

Lotus Sametime does not currently support PAT, which allows a network device to statically map an IP and port combination to a specific internal server. When PAT is used, normally one IP address will be assigned to the PAT addresses, taking inbound connections, while a second IP address will be used for outbound connections from the network device. For example, a firewall device could assign inbound connections to 192.168.0.1:80 to route to an internal Web server but route 192.168.0.1:1533 to a Lotus Sametime server. This allows two servers to share the same IP address. Outbound connections from the firewall device would come from, say, 192.168.5.1:X.

In the current implementation, the SIP connector is given a single host name or IP address to attempt a connection for a given External Community. However, in our example here, if the External Community attempts to make a connection to your SIP Connector, the connection will come from the NAT address 192.168.5.1, rather than the PAT address 192.168.0.1, and thus, for security purposes, would be rejected by the server as an illegal IP.

It is not necessary for a Lotus Sametime SIP Connector to know the actual, internal IP address of another SIP Connector, which is the reason that bidirectional NAT is supported. The requirement is that incoming network packet from an External Community must have a source IP address that matches the IP listed on the External Community document. A firewall performing bidirectional NAT changes only the destination address of the packet on incoming packets and the source address on outgoing packets.
CISCO PIX Firewall Fixup service

The SIP Fixup service on the CISCO firewall performs application inspection on all SIP network packets. The Fixup service modifies the SIP headers in the user-data portion of the packet in order to replace IP addresses with a NAT address. This modification is not necessary, however, and modifying the Lotus Sametime SIP headers can result in a loss of connectivity. It is strongly recommended to disable the FIXUP service for Lotus Sametime SIP connectivity.

Appendix B: Frequently Asked Questions

Q: Does a User Interface or Web User Interface exist—other than the Lotus Notes client—that allows an administrator to configure the Lotus Sametime Gateway or Lotus Sametime Connector?
A: No, currently the only way to modify the Lotus Sametime SIP configuration is to modify the INI files with a text editor and modify the StConfig.nsf database with a Notes client or Lotus Domino Administrator client.

Q: Can I connect a Lotus Sametime SIP Connector to a third-party SIP Community, such as AOL Instant Messenger, ICQ, or Yahoo! Messenger?
A: At the time of this writing, no public SIP connectors exist for these third-party instant messaging vendors. However, IBM has entered into agreements with both AOL and Yahoo! that would allow future versions of Lotus Sametime to interoperate with these vendors via the SIP protocol in the next version of Lotus Sametime. For more information, see Appendix C: RTC Gateway.

Q: Does Lotus Sametime SIP support any other types of encryption, other than TLS?
A: No, currently only TLS v1.0 encryption is supported.

Q: Do I need to run all the Lotus Sametime Meeting Services on my Gateway server?
A: No, only Sametime Community Services are required. You may disable Meeting Services if you do not wish to use them.

Q: Does Lotus Sametime SIP support the User Datagram Protocol (UDP)?
A: No, only the TCP protocol is supported at this time.

Q: Is it possible to browse the directory of the external SIP community?
A: No, this is not possible. If two users wish to talk to one another via SIP, at least one of the users will need to know the other users email address.

Q: Why does the padlock icon on the chat window appear unlocked when chatting with a SIP user?
A: The locked padlock indicates that the conversation is encrypted between Lotus Sametime users. When a Lotus Sametime message is sent from one user, that message is encrypted while passing from the client to the server, and again when passed from the server to the receiving client. However, Lotus Sametime cannot control whether a message sent to an external SIP user remains encrypted from the time it leaves the sender to the time it arrives at the recipient. For example, if the Lotus Sametime SIP Connector were to connect to a third-party SIP implementation, there is no way for Lotus Sametime to determine whether the conversation from the third-party SIP server to the receiving client is encrypted.
Appendix C: Future of Lotus Sametime SIP

IBM Lotus Real-Time Collaboration (RTC) Gateway

The RTC Gateway is currently planned as the next generation of SIP capabilities for Lotus Sametime. This new extensible architecture is expected to support the use of third-party plug-ins to interconnect with different instant messaging environments. Current versions of the SIP Connector will also fully interoperate with the new RTC Gateway when released (planned for 4Q2006).
in subsequent versions of the paper. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this paper at any time without notice.

Any references in this document to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.
IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
4205 South Miami Boulevard
Research Triangle Park, NC 27709 U.S.A.