



2sms Java SMTP Email Component

Target Audience

This document is a technical guide, targeted at a technical audience.

Introduction

The 2sms SMTP Email Component (email2sms) translates email messages into text messages. It does this by reading email messages from a mailbox and transmitting their content over the internet to 2sms using XML which the 2sms system translates into text messages and forwards to the cellular networks. This provides functionality very similar to that of cellular networks' own SMTP email gateways, but with many advantages, some of which are:

1. Security

Emails sent unencrypted over the public internet are relatively easy to intercept and read. It is also quite easy to 'spoof' so that anyone can send an email that appears to be from you.

The email2sms component uses only your internal systems for email and can be configured to encrypt the XML data that it sends over the public internet using industry standard SSL encryption. The 2sms system uses 128-bit public/private key pairs that are virtually impossible to crack, even with sophisticated hacking tools.

The mobile carriers SMTP gateways were implemented with no security. This made them vulnerable not only to the spoofing described above, but also to Denial of Service (DoS) attacks, whereby hackers can 'mail-bomb' an email server and overload it so that it cannot process legitimate emails, or that legitimate traffic is unacceptably delayed.

The email2sms component sends XML, rather than emails, to the 2sms system, thereby removing vulnerability to DoS attack on 2sms email servers. The 2sms system does not rely on the carriers' SMTP gateways for sending text messages, but instead maintains direct authenticated connections to dedicated messaging hubs with sub-second response times, to minimise delays.

Due to DoS attacks as described above, the cellular carriers' SMTP gateways are now being closed, or are having security retrospectively applied by 'whitelisting' only the

IP addresses of mail servers to which the carriers have explicitly granted the permission to send. This places a burden on customers, who must apply to the carriers for whitelisting, both initially and should they add or change IP addresses. Of course, not every customer will be approved and the remainder will be unable to send messages to those carriers' subscribers.

The email2sms component has built-in security that authenticates the customer on the 2sms system with each message request that it sends. This removes the need to whitelist customers and thereby the customers' overhead of requesting changes to their list of permitted email servers.

2. Reliability

Email is not a guaranteed delivery mechanism, nor is it tremendously reliable. Email servers may go down or filters may inadvertently block legitimate messages when attempting to counter spam. Even when emails do get through they are often delayed, sometimes for hours or even days.

The email2sms component only utilises email within your network, which should be much faster, more reliable and subject to far less filtering than external email. Because the email2sms component sends data to the 2sms system as XML, rather than emails, it does not suffer from the problems of speed, reliability and filtering that affect email-based messaging systems.

Where there are bottlenecks between your email server and the cellular carriers, these can be hard to isolate and harder to alleviate, as you are reliant on whomever owns the affected portion of the route taking action for your benefit.

If bottlenecks do arise in the email system, then because it is your own system you are able to troubleshoot such problems internally, rather than being reliant on a third party to do so on your behalf.

3. Scalability

Email is an inherently slow process and was never intended to handle real-time or time-critical messages, especially in high volumes. A cellular carrier that provides an SMTP email gateway must provide sufficient email servers to process as many messages as their customers wish to send and to do so as quickly as they require. Even then, performance will be marginal at best, since the email paradigm is not built on speed. Because the SMTP gateways are free and thus provide little or no revenue to the cellular carriers, they consider them a low priority for support and in terms of performance. They do not wish to expend capital on numerous email servers if they will not generate additional revenue. All of this is clearly bad news for any customer wanting to quickly send large numbers of messages.

The email2sms component leverages the email capacity of each customer that chooses to send messages in this way. The customer only needs sufficient email

capacity for their own traffic, whilst 2sms does not need to maintain numerous email servers to satisfy the collective demands of all its email customers. 2sms instead invests in a highly available and highly scalable infrastructure to which each customer sends messages as XML, meaning that high message volumes are handled far more quickly than were the 2sms system processing each customer's messages in email form.

The email2sms component is implemented using 100% Pure Java and is built upon the standard Java 2 Enterprise Edition (J2EE) JavaMail library. This makes it portable between any operating systems and environments that support J2EE, from Microsoft Windows workstations and server, to those running UNIX or Linux, Macs and RISC architectures.

System Requirements

JavaMail is part of J2EE, but is also available separately. This means that all the email2sms component requires for its operation are the following:

1. The supplied files, which include the JavaMail 1.4.1 library.
2. Java 2 Standard Edition (J2SE) runtime (JRE) version 1.5.0 (Java 5) or later. Download from <http://java.com/en/download/index.jsp>.
3. Access to a mail server that implements standard internet mail interfaces.
4. Access to the internet addresses <http://www.2sms.com>.

Installation Guide

1. Download and install a Java Virtual machine (JVM) visiting <http://java.com/en/download/index.jsp>.
2. On the same machine that you have installed the JVM, unzip and install the 2sms Java SMTP file email2sms.zip
3. From the extracted zip file, load, read and configure the config\email2sms.xml file. You can open the xml file using notepad or wordpad on a windows machine. This file needs to be configured. Open the file, and replace the following tokens with values relevant to your email system and 2sms account:

RECIPIENT_DOMAINS

A space-delimited list of the domains to which you will send emails that are intended to be sent on as text messages. For example, if you intend to send to 15551234567@mydomain.com, replace all occurrences of the string 'RECIPIENT_DOMAINS' with the string 'mydomain.com'. If you intend to send to 15551234567@mydomain.com, 15551234567@mydomain.net or 15551234567@mydomain.org, replace all occurrences of 'RECIPIENT_DOMAINS' with 'mydomain.com mydomain.net mydomain.org'.

2SMS_USER

The user name associated with the 2sms account from which you wish to send all the text messages generated by this instance of the email2sms application. 2sms usernames are usually an email address. For example, to send messages from the 2sms account 'sms@mydomain.com', replace all occurrences of the string '2SMS_USER' with the string 'sms@mydomain.com'.

To get a free no obligation 2sms trial account, visit www.2sms.com.

2SMS_PASSWORD

The password corresponding to the 2sms user name specified by the 2SMS_USER setting described above. For example, if the 2sms user

'sms@mydomain.com' has the password 'mypassword', replace all occurrences of the string '2SMS_PASSWORD' with the string 'mypassword'.

To get a free no obligation 2sms trial account, visit www.2sms.com.

MAIL_HOST

An email server on which the specified email user account may be accessed. This value may be a host name, e.g. mail.mydomain.com, a locally-accessible DNS name, e.g. MailServer, or the IP address of such a server, e.g. 196.254.13.8. This server must be accessible from the machine running this application.

MAIL_DOMAIN

The email domain from which you wish any acknowledgement emails to be sent. For example, if you wish acknowledgement emails to be sent to users from the address email@mydomain.com, replace all occurrences of the string 'MAIL_DOMAIN' with the string 'mydomain.com'.

MAIL_USER

The email user from which you wish any acknowledgement emails to be sent. For example, if you wish acknowledgement emails to be sent to users from the address email@mydomain.com, replace all occurrences of the string 'MAIL_USER' with the string 'email'. Note that, by default, this is also the user to whose mailbox the email2sms application connects both to read and send emails. If you wish to define different users for some or all of these purposes, refer to the separate sections below and change the relevant email user names to reflect your desired configuration.

MAIL_PASSWORD

The password corresponding to the email user name specified by the MAIL_USER setting described above. For example, if the email user 'email' has the password 'mypassword', replace all occurrences of the string 'MAIL_PASSWORD' with the string 'mypassword'.

By default, the application is defined to be as secure as possible. This means that email users are required to be authenticated by password, that SSL is used for reading email and TLS is used for sending email and that SSL is also used for communication with 2sms. To improve application performance you can elect to change these settings to some which are less secure.

The application as configured automatically starts checking for new emails in the 'inbox' folder of the specified email account on the specified email server every one second from the moment that it is opened. Acknowledgement emails appear to come from a user named 'Text messaging support' and have the subject line 'Text messaging acknowledgement'. (Acknowledgement emails are only sent if Reply-To header is set in the source email.)

Configuration properties whose names begin with '2sms.' or the name of a configured application, e.g. email2sms, are 2sms-specific. Those whose names begin 'mail.' Are standard values as defined by the JavaMail specification, available for download from <http://java.sun.com/products/javamail/>.

This application architecture has been tested with Java SE 5 and 6, JavaMail 1.4.1 and Log4j 1.2.15, but should function correctly with later versions. It may also function correctly with earlier versions, but to do so is not supported by 2sms.

Running JSEC

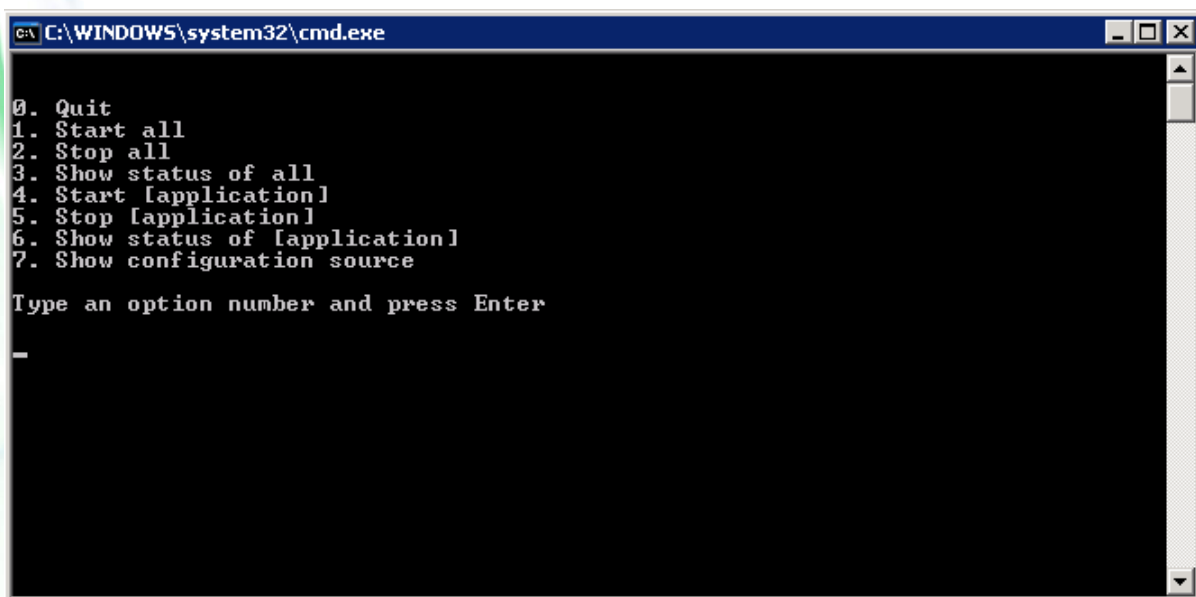
To run the JSEC application, follow these steps:

- 1) Run the batch file called Logger.bat.
- 2) Run the batch file called email2sms.bat.

To view any exceptions that are thrown in the program, there are 2 options available:

- 1) Log files stored in the Logs directory.
- 2) Running the Chainsaw.bat file. (This displays any exception after the program has started.)

To tell that the application has started without any problems, the following screen should be displayed:

A screenshot of a Windows command prompt window. The title bar reads "C:\WINDOWS\system32\cmd.exe". The window contains a menu with the following options:

```
0. Quit
1. Start all
2. Stop all
3. Show status of all
4. Start [application]
5. Stop [application]
6. Show status of [application]
7. Show configuration source

Type an option number and press Enter

-
```

Support

For support, visit www.2sms.com, email support@2sms.com or call 877 276 726 6 (USA) or +44 (0) 1234 757 800 (Rest of world)