IceWarp Unified Communications

SMS Service Reference

Version 11.4

Contents

SMS Service

The **SMS Service** incorporated into IceWarp Server is a gateway to mobile networks allowing IceWarp Server to:

- **send** and **receive** SMS messages

- use **SyncML SMS PUSH** technology to instantly synchronize server data to mobile devices

- be a bridge between **email** and **SMS** – using IceWarp Server's **sms:** protocol

- **extend** CRM and other **custom applications** by text messaging capabilities

Multiple SMS gateways can be configured within IceWarp Server, allowing a greater throughput of messages as IceWarp Server will automatically balance the workload between them.
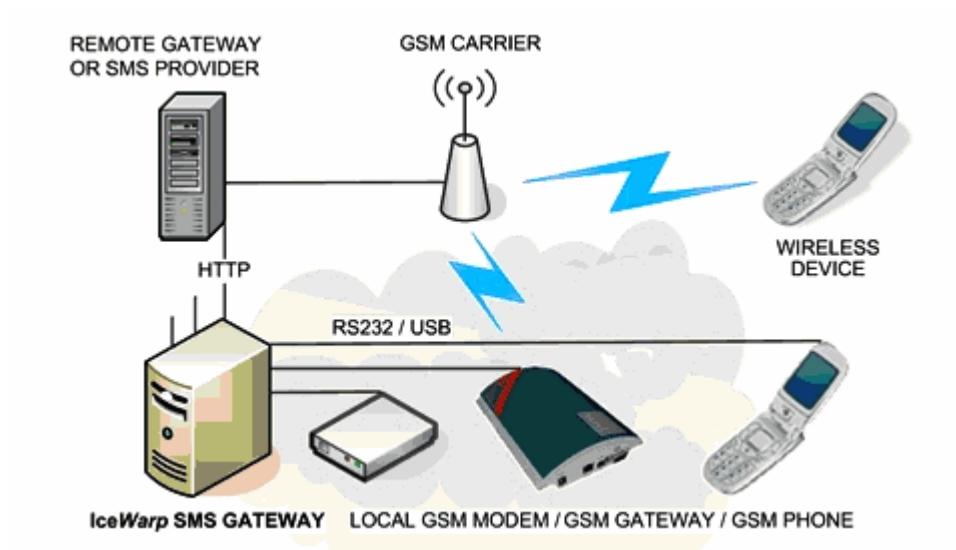
2N Voiceblue GSM Gateway. An example of a hardware gateway which can communicate over USB as a locally connected modem or over Ethernet as an HTTP remote gateway.

Legend

Icon	Description
⚠	Warning – very important!
📝	*Note or tip – good to know.*
NOTE: Areas...	*Note within a table.*
▶ Figure 4	Figure link – click the link to reveal the figure. Click it again to close it. (Works only in the **CHM** format.)

Registered Trademarks

iPhone, iPad, Mac, OS X are trademarks of Apple Inc., registered in the U.S. and other countries. Microsoft, Windows, Outlook and Windows Phone are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. Android is a trademark of Google Inc. IceWarp is a registered trademark in the USA and other countries.

About

Why Do I Need SMS Service?

You need a working SMS service if you want to:

- send or receive SMSs

- use SyncML SMS PUSH technology

- send bulk text messages

- extend web applications by SMS functionality

- enhance the WebClient/Outlook user experience with mobile communication

SMS is the best communication channel for direct marketing campaigns, social services, consumer competitions, market research and other applications.

Immediate and cost-effective at the same time, text messaging offers retailers a highly effective medium to deliver an attractive, personalized content with a greater reach and response than through print or email. According to a recently released report by the DMA, of the 800 U.S. mobile phone users surveyed, 70% said they had responded to a marketing text message whereas just 41% had responded to a survey and 30% to email offers.

http://www.mobilemarketer.com/cms/news/research/1336.html

What Hardware Will I Need?

Any standard GSM capable modem should work fine, including attached mobile phones with modem capabilities (e.g. Nokia E51). Note that some cellular phones only allow you to send out, but messages cannot be received by the means of AT commands. CDMA and other network standards were not tested, these are supported only if the modem is using the same standard AT commands to send and/or receive the messages. Please refer to the modem's specifications prior to use.

OR, none.

If you do not have a modem or need to send to an incompatible network, you can still take the full advantage of SMS service. A number of carriers offers bulk SMS services through an HTTP API, accessible remotely over Internet, so you only need to subscribe to an SMS plan with a provider matching your messaging needs.

Why Would I Want More Than One SMS Gateway?

You do not actually need more than one SMS gateway but if there are multiple gateways (any combination of remote HTTP gateways and/or USB/COM port connected modems), then the SMS workload will automatically be balanced between them.

This would give greater throughput of messages (if required) and provide service failover.

Reference

Before you can set up **SMS Service, y**ou will need to know the following information**:**

- which COM port your modem is using

  you can determine this by examining the modem properties in your computers operating system

- the PIN number of your modem (if set)

- your service providers messaging center number

  this may be already set up in your modem or you can ask your service provider

OR, if using a remote SMS gateway:

- URL of HTTP request containing, at the very least, the remote server hostname, authentication credentials, message payload format, optionally header information for binary and concatenated messages; please refer to the literature of your provider (usually found under developer/API resources).

Once you have this information, you can proceed to set up **SMS Service**.

General

Selecting the **SMS Service** node in the IceWarp Server console will display the **SMS Service** configuration area, with all current services listed:
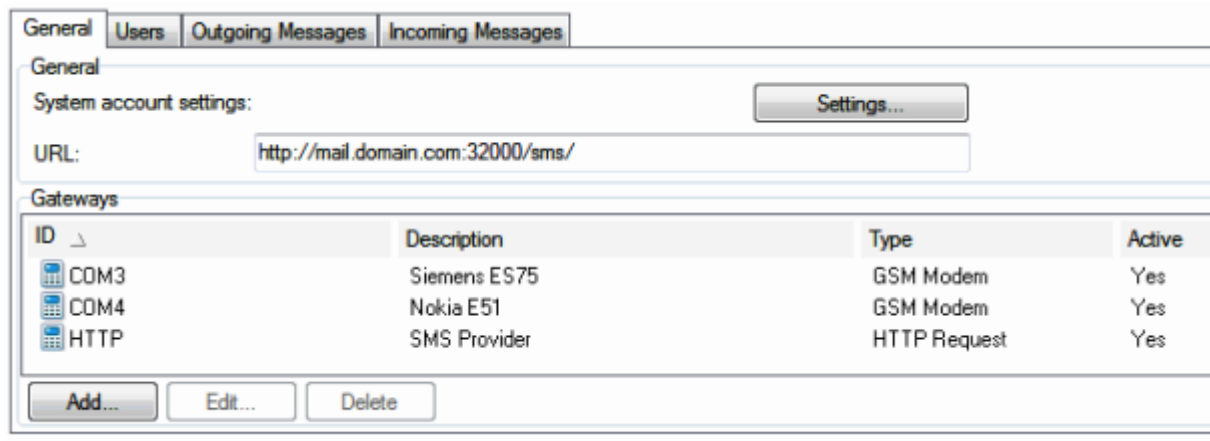
Field	Description
System account settings	This button opens the **SMS Account** dialog where you can set SMS account options. See the **Domains and Accounts – Management – User Accounts – Policies** chapter – **SMS Account Dialog** section.
URL	URL of the local HTTP gateway for integration with custom scripts or applications.
ID	This column shows the COM port that this service is using.
Description	Your free-form description for this service.
Type	The type of modem this service is using.
Active	Whether this service is active (set by you).
Add	Click this button to add a new service. The **SMS Gateway** dialog opens.
Edit	Select a service and click the button to modify settings of this service. The **SMS Gateway** dialog

	opens.
Delete	Select a service and click the button to delete this service.

> NOTE: Access mode to the service can be set on both domain and user levels. See the appropriate places (**[domain] – Policies**, **[user] – Policies**).

This dialog is used to setup or modify gateway's properties.

Field	Description
Active	Check this option to activate this SMS gateway.
ID	Unique identifier of the gateway, needs to be specified when multiple gateways are configured on the same server. Gateway ID is used to route messages to a specific gateway within gateway selection dialogs, in Outgoing/Incoming Messages rules or by URL parameter ?id=XXX. If not specified there, load-balancing takes places automatically and the first available gateway is used to send the message.
Type	Select the gateway type: **GSM Modem** This gateway will connect to a GSM local hardware modem. **HTTP Request** This gateway will connect to a remote HTTP server.
Device	**For a GSM Gateway** Select the COM port of the modem to use with this gateway. If your device is using virtual COM port rather than a standard one (COM1–4) then you should just type COM port number into the field (e.g. COM16). **For an HTTP Gateway** Select the URL given in the dropdown and modify it to suit your HTTP SMS gateway provider. The example given in the dropdown is designed to be used with a remote instance of IceWarp Server, and SMS messages will be sent by the remote server specified. Another example is set to suit to a remote HTTP gateway provided by Clickatell. For details about HTTP SMS syntax and examples, refer to the **HTTP SMS Service Setup** and

	HTTP Request chapters. For details about configuring of Clickatell HTTP gateway, refer to the **Configuring Clickatell HTTP Gateway** chapter. *NOTE: As IceWarp Server does not support SMS splitting, make sure your provider can deal with longer SMS messages (more than 150 characters) – is able to split them into more messages.*
Template	Use the list to select a pre-defined device.
Description	Enter a meaningful description for this gateway (for identification purposes).
PIN/Password	Enter any PIN for this modem (if required). Password is used for an HTTP gateway when SMS answers will be delivered via a delivery script. (See the **HTTP Request – Delivery Script** chapter. Example: *http://yourserver.com/sms/deliver.html?id=HTTP&pass=pinnumber&number=%from%3&data=* *%data% –pinnumber* is just the **PIN/Password** field within the HTTP gateway for incoming messages.)
E-mail From	This option facilitates SMS and email interoperability and must be used to enable the ***Reply-To*** feature. It allows you to reply to SMS received as emails into your mailbox, ensuring that your reply will be sent over the SMS gateway back to the originating sender's mobile number. The ***From:*** edit lets you rewrite the ***From:*** header of SMS received to the gateway, so that it can be routed to email or processed by incoming rules. Enter ***sms:%s@icewarpdemo.com*** to rewrite the ***From:*** header with an ***SMS to email*** address so that the gateway can route the message back to SMS when you reply to such email. If you do not wish to use the ***Reply-To*** feature, the header can include only ***%s*** and in such case the message will include just sender's mobile number in the ***From:*** header. *NOTE: **%s** will be replaced by the originating sender's mobile number.* The helper sms: email account needs to be configured only for use with desktop email clients, in this case all messages are routed through the internal sms: protocol.
SMS Center	Enter the message center phone number of the service provider for this modem. If in doubt (format etc.), ask your provider. *NOTE: You only need to set this if it is not already set in your modem.* *When using HTTP Request, you do not need to set it.*

Users

Authentication

There is a need for all services to be able to set the access directly per user in the accounts **Management** section. However, for SMS (and FTP) this is a challenge as they already have their own authentication settings through username/password combination. Sometimes you simply want to create specific accounts for SMS (and FTP) and not to be forced to create extra email users for them.

Access Mode

Service Access

Service access can be set on the domain level or the user one.

Each domain or user has general service *Policies*. You do not need to define any SMS (or FTP) accounts and simply use system users automatically.

If an SMTP session is authenticated, the authenticated user is used when the service is enabled in *Policies*. Therefore users do not even need to authenticate when sending SMS from their accounts and they simply only need to SMTP authenticate and SMS authentication is done automatically in the background. Limits and options are applied accordingly (globally, domain level wise or user level wise).

Username/Password

SMS gateway also implements its **Users** tab where you can define a list of users (accounts). By default, user authentication is required. The *SMS Account* feature (dialog) allows you to manage accounts, groups, number of messages sent monthly, expiration, etc.

SMS server supports "authenticated" URL variables. You can indicate that the sender email address is authenticated and must be processed as a local account. *sms: protocol* automatically uses *authenticated* parameter if SMTP session is authenticated.

Summary

Username/password authentication and access mode are integrated. If the sender is not SMTP authenticated user, the authentication against the list setup in **SMS Service – Users** is attempted. If username/password authentication is not found or invalid, the system authentication is applied. If successful, the **sms.dat** file is read from a location based on *Policies* (domain, user). If the user is not listed, he/she is denied access and cannot send.

sms.dat File

SMS configuration for IceWarp users is stored in the **sms.dat** file. There is top level configuration stored in the **config/sms.dat** file, followed by domain configuration stored in the **config/<domain>/sms.dat**, followed by user configuration stored in the **mail/<domain>/<user>/sms.dat**. (The file for the appropriate level is created when the default settings for this level is changed.) The server first checks for user configuration, if not found it tries domain configuration, if not found it tries global configuration and if this is also not found it uses default values. There are also SMS users defined within the **SMS/Users** tab. Their configuration is stored in the **config/smsusers.dat**.

The file contains colon separated data

user:password:gateway:description:maxsend:sent:lastsent:prevsent:expire:inactive

Short description:

- *user* – SMS user name, empty for IceWarp system/domain/user configuration
- *password* – SMS user password, empty for IceWarp system/domain/user configuration
- *gateway* – name of gateway specified for this user, empty for default
- *description* – SMS user description
- *maxsend* – monthly message limit
- *sent* – number of messages sent this month
- *lastsent* – unix time stamp of last message sent time
- *prevsent* – number of messages sent previous month

- *expire* – day when this account expire in Delphi date format

- *inactive* – set to 1, when account is not active

sent, *prevsent* and *lastsentdate* statistics for IceWarp users are stored within the **smsstat.dat** file. The values in the **sms.dat** are used for SMS users only.

There are **D_SMSContent** and **U_SMSContent** API variables for R/W access of this file (usually for webadmins).

Processing Order

SMTP AUTH – username/password authentication: users based on **Policies** – username/password authentication: users listed in **SMS Service – Users** (e.g. external users).

> *NOTE: On the domain or user level **Policies**, you can find settings for SMS and FTP services.*

SMS Account Setup

Field	Description
Add...	Click the button to add a new account.

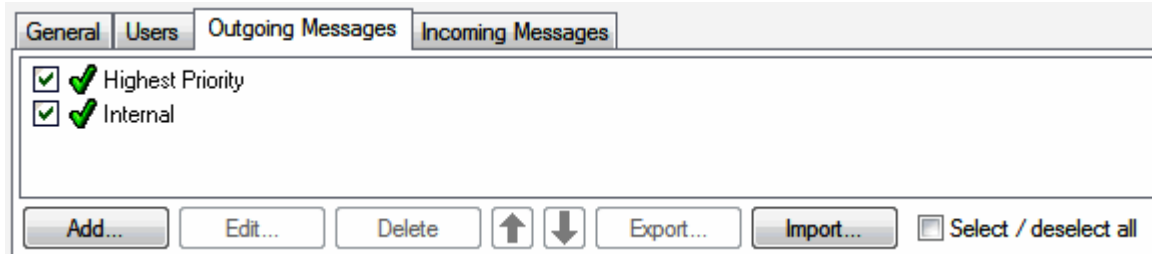| Edit... | Click the button to edit the selected account. |
| Delete | Click the button to delete the selected account. |

An SMS account serves the purpose of account management for SMS users. It offers similar functionality as email account options and includes a very basic billing. It lets you define groups of users (or individual users) which will share the same authentication, the same destination gateway and you can define a common expiration and send-out limits for them.
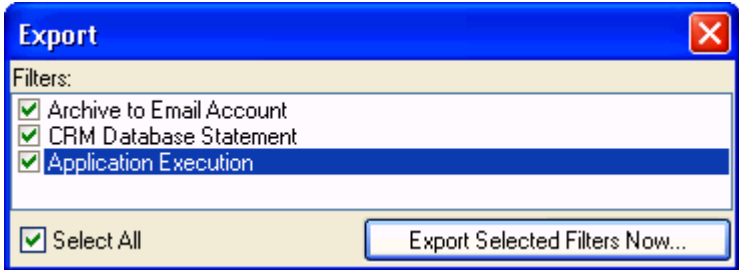

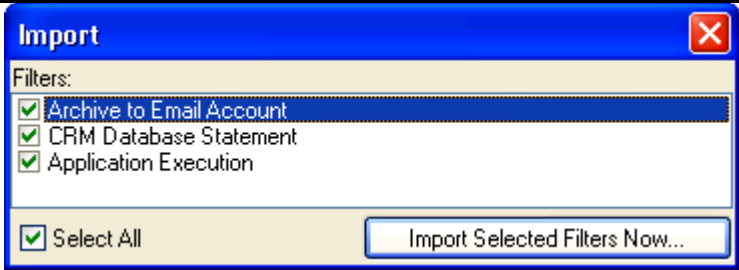| Field	Description
Active	Tick the box to activate the account.
User	Use the "..." button to select the user/account.
Password	Enter the password which will be used to authenticate the user for SMS service. It can differ from the user's email account password. If left blank, the user will not need to authenticate.
Description	Enter brief account description.
Gateway	Select a gateway from the list.
Expire	Select a date of the account expiration; if left blank, there is no expiration applied.
Monthly message limit	Enter the maximum number of messages that can be sent per month; if *0* (zero) is left, there is no limit set. For example evaluation accounts can be created by limiting a number of sent messages to 10 and/or setting a 7 day expiration.
Number of sent messages this month (Auto adjusted)	The use is two-fold: 1. **Message counter.The value shown here tells you how many messages have been sent this month by users or groups within this account. It is automatically reset to 0 at each month's end.** 2. **Counter reset. Type 0 or any other value and click OK to reset the monthly counter.**
Number of sent messages previous month (Auto adjusted)	1. **The value shown here tells you how many messages have been sent in the previous month by users or groups within this account. If there has been no activity in that month, it is automatically reset to 0 at each calendar month's end.** 2. **Counter reset. Type 0 or any other value and click OK to reset the monthly counter.**

Last sent	Shows the date and time when the last message was sent through this SMS account.

Outgoing Messages

The **Outgoing Messages** tab allows you to set rules for outgoing messages. This functionality lets you perform vast spectrum of filtering, routing, load-balancing, archiving, etc.

Field	Description
Rule Title	This field lists titles of the existing content filters for outgoing messages. Only filters with the left-hand check boxes ticked are applied.
Add	Click this button to set a new content filter. The **Rule** dialog appears. For more details refer to the **Mail Service – Content Filters – Filter Conditions** and **Filter Actions** chapters. *NOTE: Some options designed for SMTP service content filters are not available for SMS service because they are not meaningful here.*
Edit	Click this button to edit the selected rule. The **Rule** dialog appears. For more details refer to the **Mail Service – Content Filters –Filter Conditions** and **Filter Actions** chapters. *NOTE: Some options designed for SMTP service content filters are not available for SMS service because they are not meaningful here.*
Delete	Click this button to delete the selected content filter.
Up/Down arrows	Click one of these buttons to move the selected filter up/down in the list. Content filters will be applied in the order from the top down.
Export	Click the button to export one or more content filters. In the **Export** dialog, select *Filters* that you want to export and click the *Export Selected Filters Now* button. In the standard **Save As** dialog select the directory to save these filters to.
Import	Click the button to import one or more content filters. In the standard **Open** dialog, navigate to the directory with the *xml* file that contains desired content filters. In the **Import** dialog, select *Filters* that you want to import and click the *Import Selected Filters Now* button. Imported content filters will appear in the list.

Select/deselect all	Tick the check box to ease operations with more content filters (moving, deleting).

Applications

Outgoing Messages use the standard content filters system which gives you an infinite number of options, including but not limited to the following scenarios.

> A destination SMS number comes to a content filter in the following form: ***[number][@GateWayID]***. It means, the number is always there and the part with **@** (at) sign is there only, when a user has set a specific SMS gateway in user settings (**Domains and Accounts – Management – <domain> – <user> – Policies – SMS Settings – Gateway**). As a consequence, when an administrator wants to redirect some messages to a specific gateway using outgoing rule, he/she has to consider the *To:* header in the form ***number*** and ***number@userGateway***.
>
> When ANY outgoing rule condition is met, the server expects, that specific SMS gateway is defined by ***sender's_domain***. Even if the *To:* header is not modified by the rule. When none of filters conditions is met, the server will ignore the ***@sender_domain*** part and will use some gateway which the server prefers. This could lead to unexpected behavior when a user creates a rule which for example checks SMS content but server expects it also specifies a gateway.

Load-balancing

A gateway can be load-balanced with another gateway or external modem, and routing preferences configured using **SMS Service – Outgoing Messages** content filters. Gateways have their IDs that you can use to send SMS through a specific ID (id= parameter) or load-balanced automatically to any available gateway if not specified.

The message *To: header* contains the destination number + "@" and the ID of the destination gateway ID. If a gateway is not specified, the 2nd part with "@" is missing – you can use the **Edit Message Header** function to rewrite and specify the gateway ID. RegEx replace is recommended for this purpose.

You can check if the outgoing SMS is already going through a specific gateway and if not, based on the number prefix route it to a specific gateway (using the **Edit Message Headers** and the new **RegEx Rewrite**).

> *NOTE: If you have created a rule that assigns some message(s) only to one specific gateway and its delivery fails, the message stays in the queue as the server always tries to send through this gateway.*

Sent SMS Archiving

To store all SMS sent through the server, create a rule forwarding the content of SMS to an email address of a dedicated SMS archive mailbox. Use **Edit Message Headers** to record additional information about the sender and recipient.

Filtering the content of SMS

Also any content filter actions are supported, you can forward SMS to email addresses based on content filters criteria and any other scenario, you can even use the **Forward To** action to send an email back to the SMS gateway if required. To check whether @ is present, use the **(.?[\d]*)(?!@)** expression.

Least Cost Routing

You can check if the outgoing SMS is already going through a specific gateway and if not, based on the number prefix, route it to a specific gateway (using the **Edit Message Headers** and the new **RegEx Rewrite**).

> *RegEx rewrite is a very strong tool that lets you define a RegEx search pattern and, based on the pattern, it can create a rewritten result using parts of the original RegEx pattern. If the RegEx condition does not match, the **Edit Message Header** action is simply skipped. This provides great flexibility for custom content filters, especially with rewriting parts of a telephone numbers (prefixes or suffixes) in SMS and SIP gateways.*

Text message **To:** header contains the destination number + "@" + domain name or ID of the destination gateway. If the message is routed to email or to specific gateway, further routing cannot take place and you need to use the **Stop processing more rules** action. To check whether the message is already routed to email or specific gateway, look for presence of "@" using the following RegEx:

^(?!.@)(.*)$*

If gateway is not specified and the 2nd part with "@" is missing, you can safely use the **Edit Message Header** action to rewrite and specify the gateway ID, using RegEx Rewrite:

Add/Edit

Header: To:

Regex: False

Value: %%To%%@gatewayid

or

Add/Edit

Header: To:

Regex: True "^(?!.@)(.*)$"*

Value: $1@gatewayid

This action checks the presence of **number@domain** and then creates **number@gatewayid**. Substitute **gatewayid** for ID of the gateway where you want the text message to be routed to.

Typically you may want to route the message to a specific gateway based on the number format:

Add/Edit

Header: To:

Regex: True "^(\+1)(.)"*

Value: $1$2@usgatewayid

This action checks the number prefix and, if it begins with +1, routes the message to **usgatewayid**. Substitute **gatewayid** for ID of gateway where you want the text message to be routed to.

> ⚠️ BE AWARE: The syntax of regular expressions requires you to escape the special character "+" with a backslash to acquire its literal meaning.

Validating the sender

You can check the number of incoming message against a RegEx condition or a list of valid senders.

Each gateway is using the **From:** option defined in **SMS Service – General** tab – gateway properties. What you define here will appear as the **From:** header of the incoming message. You only need to create a content filter in **Incoming Messages** tab, which will validate the sender against a RegEx pattern or whether it **Contains a list from a file or pattern**:

If From RegEx: "(\+15551234567)|(\+15551238866)" (if the sender number is from the list) than for example: **Forward To: john.doe@icewarp.com**

This will route the message from one of those foreign numbers to the appropriate recipient.

Rewriting the Caller ID

You can rewrite the sender's **Caller ID** so that the message will appear to recipient as sent from a number you specify.

It must be supported by the wireless carrier (GSM or HTTP service provider). If not available, the number associated with the SIM card or SMS account will be used.

You can use the phone number global variable as found in **Management – [user] – General – Phone #** field and then create action which adds the ***&sender=%sender_phonealias%*** parameter.

Add/Edit

Header: To:

Regex: True "^(..@)(.*)$"*

Value: $1@$2&sender=%sender_phonealias%

Again, first you need to verify whether the message is not already routed to email – see **Least Cost Routing**.

Non-Delivery Reports

Thanks to integration with Mail Server, IceWarp SMS Server is using a bounce-back system for text messages sent via email or WebClient which works exactly the same as SMTP NDR:

1. A warning is sent by e-mail if the text message is still not delivered after some time (**SMTP – Delivery – Warning** option).

2. If the message cannot be delivered and expires (SMS API setting), sender will receive the final NDR on message delivery failure.
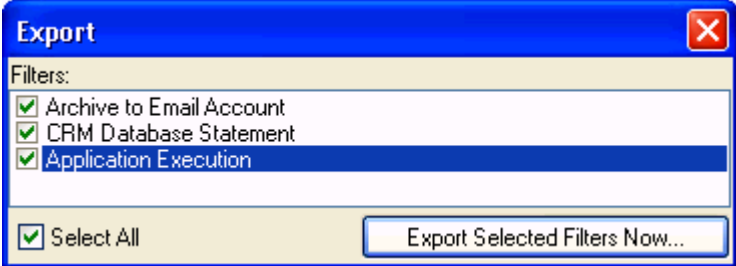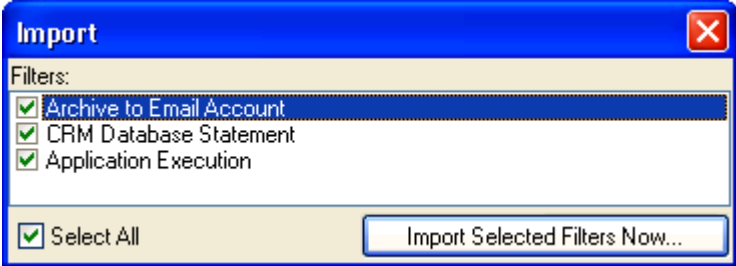
To override the default global message expiration (7200 minutes, i.e. 5 days), use the command line in root of IceWarp Server installation:

tool modify system C_SMSService_ExpireMins

The original text is always included. This applies only if the real sender is known (sent from WebClient, email or specified in the SMS API).

Incoming Messages

Field	Description
Deliver according to #email# in received messages	If an SMS is received containing the hash delimited email address anywhere in its body, the message will be routed to the corresponding user's mailbox. Users can use the rules to match incoming SMS messages and forward them or place them into special folders. Example: User can create a new email folder for SMS messages (e.g. **SMS**) and create a rule for moving these messages there: Condition: **From: contains sms:** Action: **Move to: <user's_email_address>/SMS**
Deliver to original sender of received reply message (keep a list of sent messages)	If enabled, the system creates a "call log" of sent messages history (links between sender's email and destination number) and if incoming SMS is received from a number we previously sent a message to, the sender from history is read and message is delivered to this sender (within a time frame). *NOTE: Each link between sender's email and destination number has its own expiration and when an incoming SMS is received, the links are checked. If there is none (or all expired), the usual incoming rules are used to process it. If there is one link, the message is delivered to the recorded sender's email. If there are more links, a bounce back SMS is sent to the original sender with the message defined in the GUI (see further).*
Keep the records for (Min)	Define the time (in minutes) how long the link between sender's email and destination number is to be active.
Conflict message sent back to sender when more than one record exists	Define a conflict message which is sent back to the mobile replier if there are more active links. Basically, it says that he/she should re-send the message with the **#email#** string inside the SMS body.
Title	This field lists titles of the existing content filters for incoming messages. Only content filters with the left-hand check boxes ticked are applied.
Add	Click the button to set a new content filter. The **Rule** dialog appears. For more details, refer to the **Filter Conditions** and **Filter Actions** chapters. *NOTE: Some options designed for SMTP service content filters are not available for SMS service because they are not meaningful here.*

Edit	Click the button to edit the selected content filter. The **Rule** dialog appears. For more details, refer to the **Filter Conditions** and **Filter Actions** chapters. *NOTE: Some options designed for SMTP service content filters are not available for SMS service because they are not meaningful here.*
Delete	Click the button to delete the selected content filter.
Up/Down arrows	Click one of these buttons to move the selected content filter up/down in the list. Content filters will be applied in the order from the top down.
Export	Click the button to export one or more content filters. In the **Export** dialog, select *Filters* that you want to export and click the *Export Selected Filters Now* button. In the standard **Save As** dialog, select the directory to save filters to.
Import	Click the button to import one or more content filters. In the standard **Open** dialog, navigate to the directory with the wished *.xml* file that contains desired content filters. In the **Import** dialog, select *Filters* that you want to import and click the *Import Selected Filters Now* button. Imported content filters will appear in the list.
Select/deselect all	Tick the check box to ease operations with more content filters (moving, deleting).

Reply-To Feature

This feature works like a bridge between email and SMS. It makes possible sending SMS messages over an SMS gateway (from WebClient or Outlook) and receive a reply to that SMS to the original sender's mailbox without defining any rules on the SMS server.

Recipients simply reply to the SMS received using the destination number of the SMS gateway, and the **Reply-To** mechanism automatically routes the message to the original sender's mailbox. All existing incoming content filters are skipped.

Alternatively, recipients can send messages or reply to received messages with #email# string included in the text message (e.g. *#demo@icewarpdemo.com# Hello world*) and the message will be routed directly to that email address, skipping the **Reply-To** mechanism.

Existing incoming rules are applied.

For each SMS coming to an email address, the resulting email has the **To:** header set to the value of the final recipient.

*NOTE: This is not applied if content filters are used to deliver the message and you need to define additional action to add the **To:** header if desired, using Condition: **All messages** and Action: **Edit message header : To:**.*

To use the **Reply-To** feature you need to enable these options:

1. **SMS Service – General**, in the gateway properties the ***From:*** header rewrite has to be configured such as: ***sms:%s@icewarpdemo.com***.

2. **SMS Service – Incoming Messages** – ***Deliver according to #email# in received messages*** must be enabled.

3. **SMS Service – Incoming Messages** – ***Deliver to original sender of received reply message (keep a list of sent messages)*** must be enabled.

GSM Service

The GSM gateway is used for connecting an external GSM modem device to the computer where IceWarp Server is running.

Using a GSM modem enables you to send and receive SMS messages from IceWarp Server directly to the wireless carrier.

Tested and Supported Products

The GSM SMS Service functionality has been fully tested with the following products:

- Siemens ES75

- Huawei E220

- Huawei E1556

- MTCBA-G-F2

but should work with any standard GSM modem device capable of serial communication with the computer through a set of AT commands.

Nokia phones usually support sending SMS via AT commands, but do not support receiving them.

If you want to test your modem, you can follow the link to a step by step description:

http://esupport.icewarp.com/index.php?/Knowledgebase/Article/View/196/20/how-to-find-out-whether-your-device-can-be-used-for-sending-and-receiving-sms-messages-via-icewarp-server

HTTP Gateway

An HTTP gateway allows you to use a remote SMS service to send SMS messages, as opposed to having a GSM modem connected locally to your server. This remote service might be provided by your GSM carrier, a third-party SMS service provider, or even another instance of IceWarp SMS Server.

This could be useful in a large distributed organization where you could have a single IceWarp Server set up to deal with all SMS requests, or even multiple services around the world to take advantage of cheaper national SMS pricing plans – using rules (see **SMS –Outgoing Messages**) to decide which provider to use.

Tested Products

The HTTP SMS Service functionality has been fully tested with the following carriers:

- Clickatell
- Routo Telecom's routomessaging.com

HTTP SMS Service Setup

To configure sending SMS messages over the **HTTP** protocol, you can follow the **SMS Service Setup**, but when selecting *Type* (the **SMS Gateway** dialog), choose the *HTTP Request* item.

In the next step, you have to select (and optionally modify) an URL in the *Device* field. There are two default URLs there:

1. The first one is for communication with another IceWarp Server and its HTTP SMS gateway. This supports all SMS types: ASCII, Unicode, binary.

2. The second one is for Clickatell HTTP SMS gateway. (See the **Configuring Clickatell HTTP Gateway** chapter.)

URL example:

http://<server>/sms/?number=%number%&data=%data%&udh=%udh%&binary=%binary%&pid=%pid%&dcs=%dcs%&sender=%sender%

This URL consists of the following parameters:

(You can use corresponding variables for them.)

Parameter	Variable
&number	%number%
&data	%data%
&udh	%udh%
&binary	%binary%
&pid	%pid%
&dsc	%dcs%
&sender	%sender% %sender_phonealias%
&parts	%parts%
&condition	%condition%

*NOTE: The %sender_phonealias% variable uses the value defined in the **Phone #** field (**Domains and Accounts – Management – <domain> – Users – <user> – User**) – provided that the **&sender** parameter is supported by your SMS provider.*

These parameters can be combined with the following attributes:

Attribute	Description
;unicode	encodes the parameter to the *Unicode* format.
;isunicode ;isascii ;isbinary	conditions that allow you selection from the attribute modifiers based on the encoding of a parameter, or to append a custom *value=* parameter only if a condition (isunicode, isascii, isbinary) is met.
;value=	adds text to URL parameters.
;hex	encodes the parameter to the *hexadecimal* format.
;len	adds the length character to any parameter, usually used for UDH, e. g.: *%udh;len;hex%* – udh parameter, its length, encoded to hexadecimal format. Required for proper UDH handling by Clickatell gateway. It can be used for any parameter such as

	%body;len%.
;b64	encodes the parameter to the ***base64*** format.
;charset=	is used to convert ***UTF-8*** text to a different charset.

Configuring Clickatell HTTP Gateway

This section describes configuration of the IceWarp SMS Server with a remote HTTP gateway provided by Clickatell, a bulk SMS provider.

IceWarp Server uses special kinds of SMS messages for the SyncML Push technology that requires special URL parameters for use with remote HTTP gateways:

- Configuration messages according to OTA (over-the-air) or OMA (Open Mobile Alliance) standards

- Push notification messages compliant to SyncML 1.1 or OMA DS 1.2 standards

These are binary (8-bit) messages, which need to be passed in a special way for the Clickatell HTTP gateway to accept them. In addition, configuration messages can span multiple SMS (only 140 bytes of payload are available), which requires additional URL parameter to support them.

Compatibility

Support for binary and long binary messages is officially available since IceWarp Server 9.4.0 and the preceding beta versions.

For older IceWarp Server versions, the Clickatell HTTP gateway works for short text messages (up to 160 characters). By adding an optional parameter (***&concat=%parts%***), the older version can be made to work for long (concatenated/spanned) SMS text (not binary) messages as well.

Special Variables

Use the ***%udh;hex;len%*** parameter for special way of construction of the UDH parameter for remote Clickatell gateway.

Use the ***%parts%*** parameter to specify the number of messages in total.

The ***%parts%*** parameter returns the number of standard messages or message chunks (with length of 160 for of text, 140 in case of binary message) a long/concatenated message will be divided to. Used by Clickatell, but can be handy for other remote gateways or in special situations.

The ***;len*** parameter attribute adds the length character to any variable, usually used for UDH, eg. ***%udh;len;hex%.***The ***;len*** attribute puts the length of the parameter value to the beginning of the parameter, as required for proper UDH handling by Clickatell gateway. It can be used for any parameter such as ***%body;len%***.

The ***;hex*** parameter attribute encodes the parameter to hexadecimal format. The ***;b64*** parameter attribute encodes the parameter to the base64 format. Both attributes are applicable to all HTTP parameters and can be used if required but the remote gateway.

The ***;unicode*** parameter attribute encodes the parameter to Unicode format.

All these attributes are applicable to any text parameters.

The set of query parameter modifiers ***;isunicode***, ***;isascii***, ***;isbinary, ;value*** allows to select from the following attribute modifiers based on the encoding of a parameter, or to append a custom ***value=*** parameter only if a condition (***isunicode, isascii, isbinary***) is met.

The dummy variable ***%condition%*** is replaced with empty string as the request is processed, and as such is suitable for creating conditional expressions, e.g.: %condition;isunicode;value=&unicode=1%.

UDH (Universal Data Header)

By default, the UDH parameter is formatted in base64 without the length prefix.

UDH SAR parsing addresses compatibility with long/concatenated binary messages and the general HTTP gateway, ensuring proper UDH construction in the case the HTTP gateway is selected to send the binary message.

Pre-requisites

- IceWarp Server 9.4.0 (2008-10-27) or higher

- Account registered at *http://www.clickatell.com/* and charged with a sufficient credit

- Login credentials to your Clickatell account: username, password

- API_ID created in your Clickatell account

API_ID is NOT your Clickatell account Client ID. Prior to using HTTP requests for sending messages via Clickatell, you need to create the API_ID:

1. Login to your Clickatell account at *https://www.clickatell.com/login.php*.

2. At the login page, select the *Clickatell Central (API)* item and enter your credentials as received in your account activation email and password.

3. From the top menu, select the *Manage My Product* item.

4. Lookup the HTTP/S line in the *Help Information* box and click *Add connection*.

5. In the *HTTP API* dialog, you need to provide a descriptive name for the connection only.

   – *IP Lock Down* limits the sending machine to the server's IP address.

   – *Dial Prefix* limits the message recipient to the selected country.

   – *Callback Type* should be set to *HTTP GET*.

– Callback URL (as well as Callback Username and Callback Password) is not supported and can be left blank.

6. When done, the new connection is created in the **Manage My Products – My Connections** list and also on the home page (Central Home) under **API Connections**.

   Note the API_ID.

Configuration of Binary/Text Message Gateway

In the IceWarp Server administration GUI, create a new HTTP gateway:

1. Access the **SMS Server** node.

2. Click the **Add...** button.

3. Use e.g. "Clickatell" as gateway ID to differentiate it from other settings.

4. Select **HTTP Request** from the **Type:** dropdown.

5. Select the pre-defined HTTP request URL from the **Device** field.

   http://api.clickatell.com/http/sendmsg?user=xxx&password=xxx&api_id=xxx&to=%number%&udh=%udh;len;hex%&data=%data;isascii%%data;isunicode;unicode;value=&unicode=1%&concat=%parts%

   Change "xxx" for the actual username, password and api_id.

6. Enter any comments to the **Description** field.

If you have multiple gateways, you will need to select this gateway in all SyncML SMS Push related settings dialogs, it is identified by its ID name.

Optionally, you can set the Clickatell **&concat=n** parameter to a preset number so that a long message cannot span more than this number of messages – longer SMS will be rejected and not sent. For correct handling of binary messages, set **&concat=2** or higher.

Also optionally (for Clickatell), you can use the **&from=[from_number]** parameter. This allows you to direct SMS replies to a mobile device defined by its **from_number**.

In the case of Clickatell, you have to request a **Sender ID** number and **From** that you will use when sending messages. Just login to Clickatell, go to **Settings/Sender IDs**, put your cell phone number and wait for the confirmation you will receive and have to type in.

Example for one user:

http://api.clickatell.com/http/sendmsg?user=john&password=john's_password&api_id=2324231&to=%number%&from=552199999999&data=%data%&concat=%parts%

For more users you have to:

1. Register all of them to Clickatell.

2. Add them as users of the appropriate SMS gateway.

3. Fill in their *Phone #* fields (**Management – <domain> – <user> – User**) – use the international phone number format with + e. g. +420776047467.

4. Into the URL, add the *&from* parameter. Use the following syntax: *&from=%sender_phonealias%*

Text Messages Only

You can use the same gateway with the above configuration also for text messages, the *hex* encoding will not be applied to the message body and the UDH value will be empty. If the SMS payload is binary data, it is automatically *base64* encoded.

Or you can configure multiple gateways with the same Clickatell account and API_ID but with a different URL Post request. The simplified URL for sending text messages only (no Unicode support) would be:

http://api.clickatell.com/http/sendmsg?user=XXX&password=XXX&api_id=XXX&to=%number%&data=%data&concat=%parts%

Configuration for other providers than Clickatell is quite similar, you only have to differentiate which parameters belong to IceWarp Server (those can be reused as) and which are required by Clickatell (these need to be replaced). The **HTTP Request** chapter will familiarize you with that.

Troubleshooting

Delivery Reports

Delivery reports can be found in the web interface to your Clickatell account – the **Message Reports** menu. The easiest access is via the *Last 10 Messages* left menu item.

Correct status is **Received by recipient** without further errors. Also the charge can be helpful – each part of a long message is charged as one message, so a two-part message will be charged like two messages and so on.

SMS Server Log

First, enable the SMS server logging in **System – Services – General – *Logging***. Then observe the logs under **Logs – Service: *SMS*** and/or ***Sync Push***.

HTTP Gateway with Client Certificates

Some phone operators (e. g. O$_2$ – Europe) can provide SMS gateways that require client certificates. As IceWarp Server supports them, it is easy to set such a gateway.

The whole settings are XML-based. There is a new gateway tag – ***<CLIENTCERT/>*** – in the **<inslall_dir>\config\smsgateway.dat** file. Simply specify the full path to the client certificate in PEM format which is used by OpenSSL.
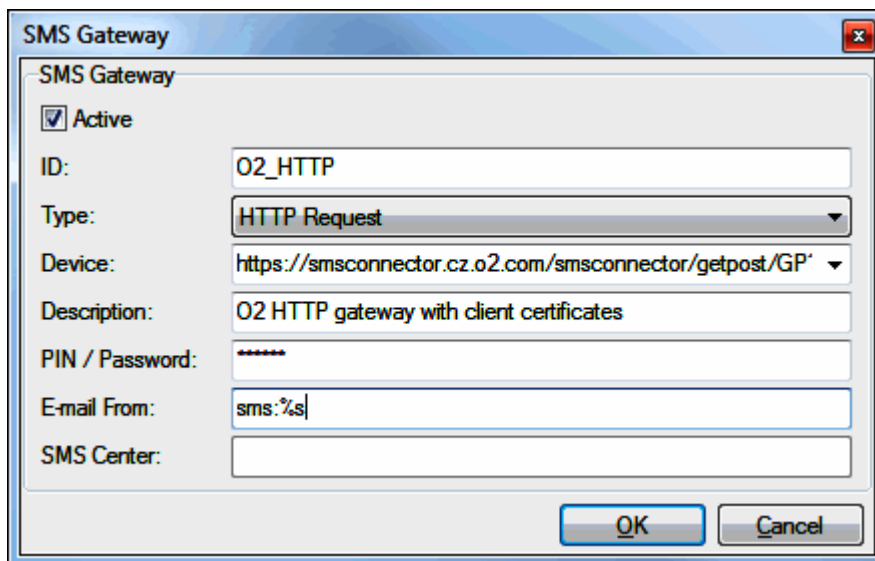
Example:

<CLIENTCERT>C:\certificates\cert_o2.pem</CLIENTCERT>

As long as this condition is met and you specify the full HTTPS URL the rest is up to the SMS service. From now on, you can send SMS messages from within IceWarp Servers and use the built-in SMS server and integrate it with O2 SMS HTTP gateway.

The URL for such a gateway would be e. g.:

https://smsconnector.cz.o2.com/smsconnector/getpost/GP?action=send&baID=1234567&toNumber=Nnumber%&text=%data%

Sending SMS from Server

This section describes two ways in which you can send SMS from IceWarp Server. **sms: protocol** is used in place of an email address within the server (i.e. locally). ***HTTP Request*** can be used both locally and remotely to reach the SMS gateway by the means of URL POST request. Both can use a set of ***optional parameters***.

sms: Protocol

IceWarp Server has an internal **sms:** protocol URI defined to allow you to forward messages (emails, XMPP, server variables) to SMS by replacing the recipient email address (***To:***) with ***sms:<number>*** URI.

The **sms:** protocol can be used anywhere in IceWarp Server where an email address can be used, within forwards, rules, filters, notifications, watchdogs and other actions, allowing an extremely flexible way to communicate instantly via SMS messages.

For example you, as an administrator, could set up the report email address of the remote server watchdog to send an SMS to your mobile so you are instantly aware of any problems found by the watchdog.

Another application of SMS gateway is for new email notifications – information about the arrived email (subject, sender, size of the body) that will be included in the SMS can be fully customized using the ***Notification*** account type. Then you only need to set a ***Forward To*** from the user's mailbox to this account, or the user him/her-self can set a rule in their e-mail client.

The basic format of the protocol is:

sms:<number>

Where:

> <number> – is the number that you want to send the sms to – REQUIRED.

Authenticated format of the protocol is:

sms:<number>?user=<username>&pass=<password>

Where:

> <username> – is a valid username for authentication.
> <password> – is the valid password associated with the username – only required if authentication is active.

The full format of the **sms:** protocol with all optional parameters is:

sms:<number>?user=<username>&pass=<password>&maxmsgs=<N>&binary=<binary>&udh=<udh>&pid=<pid>&dcs=<dcs>&sender=

<sender>&authenticated=<authenticated>&reply=<reply>&id=<ID>

> ⚠️ BE AWARE: If **@** (at) should occur within any parameter (username, password, etc.), it has to be replaced with ***%40***.

The sms: URI supports a set of additional HTTP-like parameters appended after the ? delimiter.

See the **Available Parameters** chapter for details.

sms: Protocol Examples

- sms:+15551234567

  forwards the first (by default 3) SMS-sized blocks of the email to number +15551234567.

- sms:+15551234567?user=john&pass=johnpw

  forwards the first (by default 3) SMS-sized blocks of the email to number +15551234567, if the user/password combination is defined in SMS service authentication.

- sms:+15551234567?user=john&pass=johnpw&maxmsgs=10

forwards the 10 SMS-sized blocks of message data to number +15551234567, but only if the user/password combination is defined in SMS service authentication.

HTTP Request

This method allows you to send SMS message over the **HTTP** protocol. This way your IceWarp Server can be used as a remote SMS gateway, which gives you infinite options to extend any web applications by SMS functionality, while the receiving, archiving, processing and sending is all taken care of by IceWarp Server. Server-side processing can be extended by the means of *Rules* and *Executable accounts*.

Basic format of the HTTP request (for an instance of IceWarp Server) is:

http://<YourDomain>/sms/?number=<number>&data=<message>

Authenticated format of the protocol is:

http://<YourDomain>/sms/?number=<number>&data=<message>&user=<username>&pass=<password>

Where:

<number> – is the cellular number of the receiving mobile device. You can use the ***%number%*** server variable here.

Also the following format is supported: ***%number;interprefix=<val>%***. If this option is used and the number starts with "+" , the "+" is changed to ***<val>***.

E. g. *num=%number;interprefix=00%*

and the number is +4205452544

it will get: *num=004205452544*

<message> – is the message you want to send. You can use the ***%data%*** server variable here.

<username> – is a valid username for authentication. Only required if authentication is active.

<password> – is the valid password associated with the username. Only required if authentication is active.

The full format of the **HTTP request** (for an instance of IceWarp Server) with all optional parameters is:

http://<YourDomain>/sms/?number=<number>&data=<message>&user=<username>&pass=<password>&maxmsgs=

<N>binary=<bin>&udh=<udh>&pid=<pid>&dcs=<dcs>&sender=<sender>&authenticated=<authenticated>

&reply=<reply>&id=<ID>

See the **Available Parameters** chapter for details.

> *NOTE: The above example is **only valid for an instance of IceWarp Server** (local or remote). If you are configuring an SMS gateway for use with a service of an external SMS gateway or provider, you should modify the URL data according to your providers requirements. The example shows a remote Clickatell gateway used to deliver messages through an external server, please read on for details.*

HTTP Request Examples

- *http://<YourDomain>/sms/?number=+15551234567&data="hello world"*

  sends the message "hello world" to number +15551234567

- *http://<YourDomain>/sms/?number=+15551234567&data="hello world"&user=john&pass=johnpwd*

  sends the message "hello world" to number +15551234567 using authentication

- *http://smsc5.routotelecom.com/SMSsend?user=xxxx&pass=xxxxx&number=%number%&message=*

  %data%&type=LongSMS&ownnum=mysenderid

  sends concatenated message over ***routemossaging.com***'s gateway using custom ***Sender ID*** (if supported by the carrier)

HTTP Request – Delivery Script

This feature allows you to send and receive SMS messages via HTTP request even from a server that is not directly connected to a GSM modem, but is connected to another IceWarp Server that is. Sending messages from such a server was possible (provided that there was a gateway defined on this sending server), but receiving not.

The new delivery script lets you to define a rule for *incoming messages* (on the server connected to a modem):
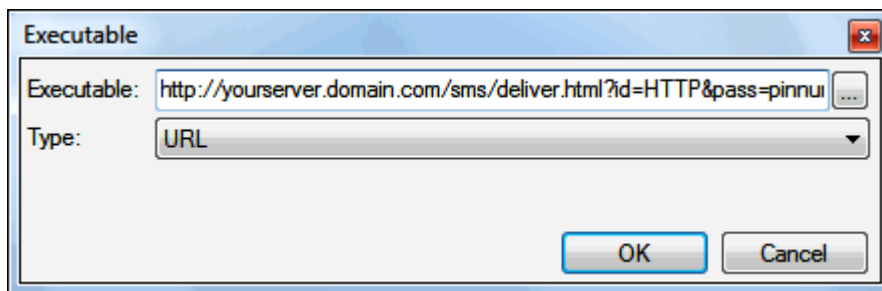
Condition: *All messages*

Action: *Execute application*

for *application* set:

Executable:
http://yourserver.com/sms/deliver.html?id=HTTP&pass=pinnumber&number=%from%3&data=%data%

Type: *URL*

This URL for SMS delivery requires the following URI variables:

- *id* – ID of the gateway to deliver the message via

- *pass* – gateway password (matches the PIN number)

- *number* – number of the SMS sender

- *data* – data of the sms sender (UTF-8)

- *binary* – 0/1 – boolean telling whether data is in binary format, no use for that currently

- *yourserver* – the server that is not connected to a modem

The only other necessary thing is to add a user on the server with a modem (either a usual one or within the **SMS – Users** tab) and pass the URI variables.
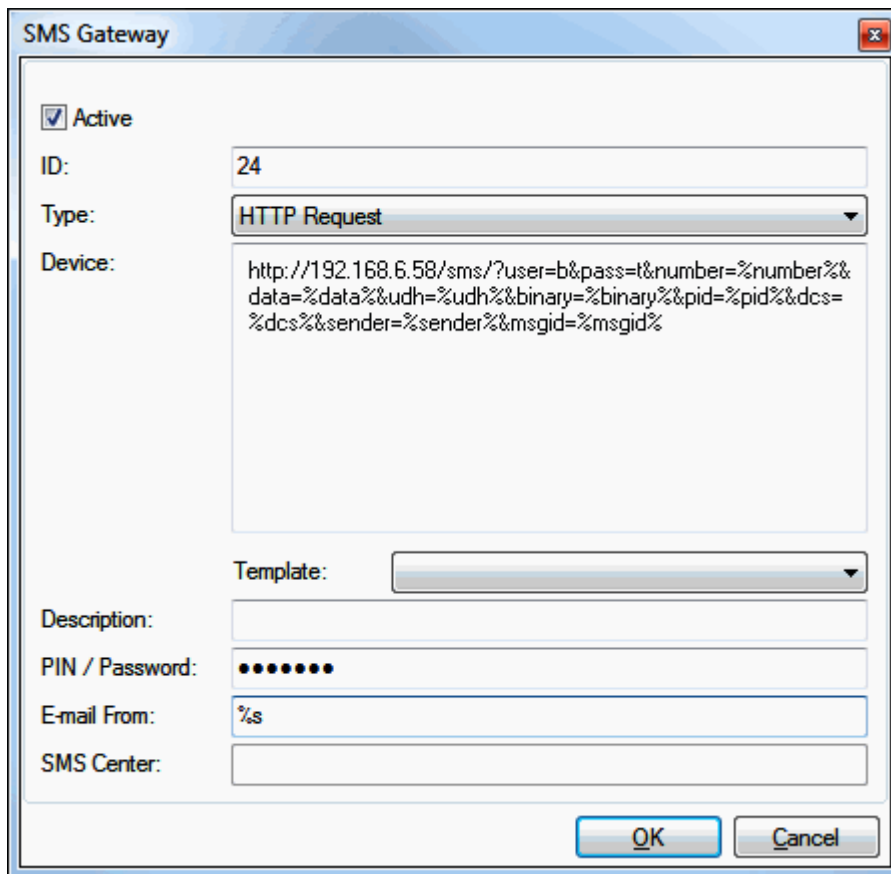
Scenario : SMS "Provider" Is another IceWarp Server

Say, you have one IceWarp Server (*server A*) without a modem and with a SMS HTTP gateway defined. The second IceWarp Server (*server B*) uses a modem.

How to set these servers the *server A* users to be able to send SMS messages and obtain answers:

Server A:

Create a SMS HTTP gateway (**SMS Service – General**) pointing to *server B*. Use settings similar to the ones in the figure:

Server B:

Create a rule for incoming messages (**SMS Service – Incoming Messages**). This rule will direct SMS answers to *server A* users. Use settings similar to the ones in the figure:
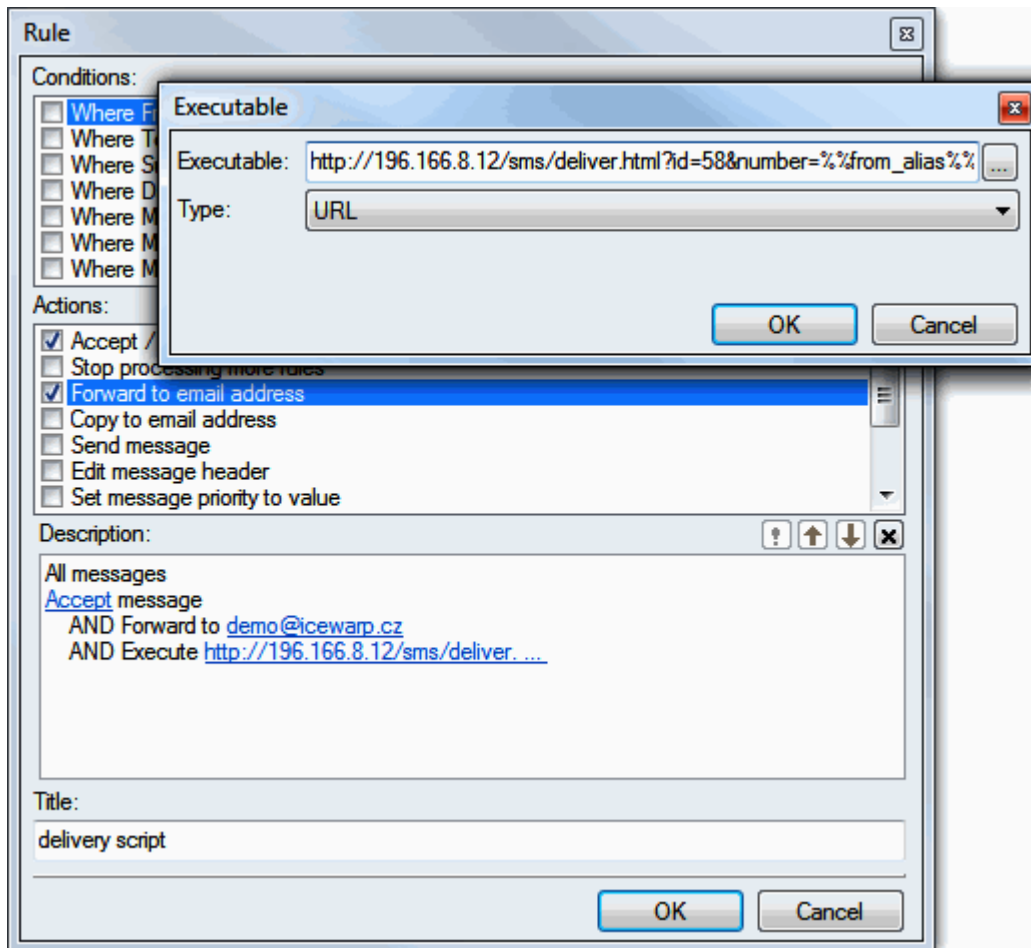
Syntax:

http://[server_A_IP_address]/sms/deliver.html?id=58&number=%%from_alias%%&data=%%body%%&pass=1234567

where:

    id – *ID* of the SMS gateway (24 in the first figure),

    pass – *PIN/Password* of the gateway (1234567 here).

All the parameters are obligatory.

Available Parameters

Parameter	Description
<number> or number=<number>	The number of the receiving mobile device. You can use the **%number%** server variable here.
data=<message>	The message you want to send. You can use the **%data%** server variable here. Not applicable for the **sms: URI**. You can use the **;hex** and **;b64** parameter modifiers here to convert the data to hexadecimal or base64 encoding and the **;len** modifier to insert the length of the data into the beginning of the data (if required). See the next chapter for details.
binary=<bin>	Used by system integrators for binary messages – ignore this unless you know what it is. SMS gateway supports UDH, 7bit, 8bit and Unicode encoding. **binary=1** indicates the message will carry a binary (8-bit) payload, and UDH must be constructed. By default it requires the <**message**> value to be base64 encoded and the <**udh**> value as well.
udh=<udh>	Used by system integrators for binary messages – ignore this unless you know what it is. You can use the **;hex** and **;b64** parameter modifiers here to convert the data to hexadecimal or base64 encoding and the **;len** modifier to insert the length of the data into the beginning of the data (if required). See the next chapter for details.
pid=<pid>	Used by system integrators for binary messages – ignore this unless you know what it is.
dcs=<dcs>	Used by system integrators for binary messages – ignore this unless you know what it is.
user=<username>	Valid username – REQUIRED if authentication is active.
pass=<password>	Valid password for the supplied username – REQUIRED if authentication is active.
maxmsgs=<N>	Maximum number of SMS messages that should be sent if the message is too long to fit one standard sized SMS message. When specified (an integer number), tells IceWarp Server to truncate the exceeding data and forward the first N standard SMS-sized blocks of data only. Allows you to control the maximum number of messages the long (concatenated, SAR) messages can consists of. By default, when this parameter is not specified, HTTP request is not limited, **sms:** protocol messages are reduced to one standard message, and text messages sent through WebClient are limited to five standard-sized messages. If you want to forward a complete message, you should set this option to a bigger number, the maximum acceptable value is 128. The text message *<data>* will be truncated according to this parameter and delivered as one concatenated message to the device. SMS gateway takes care of the concatenation with GSM modem, with remote HTTP gateway in use you might need to pass it also the **%parts%** server variable, which specifies how many standard-sized messages the **%data%** payload consists of. *NOTE: The maximum message body payload in the SAR specification is set to 2048 bytes, therefore the maximum value you should use is 12 – any longer message body can be truncated by the carrier, service provider or device itself.* *NOTE: This parameter does not apply to an HTTP request. The application or script sending the request must take care of any truncation itself, or use options offered by the remote HTTP gateway or service provider.*
sender=<sender>	The originating number. Allows to rewrite the sender's **Caller ID** so that the message will appear to recipient as sent from a number you specify here. *NOTE: This feature must be supported by the wireless carrier (GSM or HTTP service provider). If not supported or specified, the number associated with the SIM card or*

	provider's account will be used.
authenticated=<auth enticated>	Boolean value. **"1"** tells IceWarp Server that access to the gateway is already granted and authentication is to be skipped. If the parameter is **"0"** or omitted, authentication be evaluated before sending the message.
reply=<reply>	The sender's **Reply-To** address to be included in the message body, usually an email as this parameter is used by the **Reply-To** feature for bridging SMS and email. See **Incoming Rules** for details.
id=<ID>	Identifier of an SMS gateway in the case there are multiple gateways configured. It allows you to route messages to a specific gateway, for example by rules, to facilitate **Least Cost Routing** based on international/network prefix of the recipient's mobile number. Use hex encoding of HTML special chars (e.g. **%20** for space). When not specified, the first available gateway is used (load-balancing).
value=<string>	Optional parameter which allows passing custom parameters that are interpreted by the destination gateway, not by IceWarp SMS server. For example, if you wish to pass **unicode=1** parameter in HTTP request to a remote SMS gateway, you will specify **value=&unicode=1**. You can use the value parameter multiple times inside the same request.

Scenario Example

This is a scenario for sending and receiving SMS through external IceWarp Server.

Server #1 – a GSM modem is connected to this server.

Server #2 – this server wants to send/receive SMS, but has not any GSM modem connected.

Setup within IceWarp Servers

Server #1

SMS – Incoming Messages – Add Rule

- **All messages** (or different condition if needed)

- **Execute Application** (type URL) https://server#2#hostname/sms/deliver.html?number=%%from%%
   &id=<gateway#id>&data=%%body%%&pass=<gateway#password>

Server #2

SMS – General – Add Gateway

- **ID**: <gateway#id>

- **Type:** HTTP request

- **Device:** https://server#1#hostname/sms/?number=%number%&data=%data%&udh=%udh%&binary=
   %binary%&pid=%pid%&dcs=%dcs%&sender=%sender%&msgid=%msgid%&user=<authuser>&pass=
      <authpassword>

- **PIN**: <gateway#password>

Value	Description
<gateway#id>	Any value, but it has to be the same on both servers.
<gateway#password>	Password protecting the gateway from misuse.
<authuser> + <authpassword>	User on the server #1 which is allowed to send SMS (created for this purpose).

Receiving SMS from Third Party Gateways

This chapter describes receiving SMS messages coming from third party gateways as emails.

IceWarp Server user sends a SMS message via HTTP, this message is delivered by third party SMS gateway, the recipient answers from their cell phone, this answer is posted to IceWarp Server (script in the **<install_dir>/html/sms** directory – see further) and delivered to the user as an email message.

> NOTE: The **deliver.html** file is a raw example of such a script. There are more examples of scripts for other providers within this directory.

It is necessary to have an incoming number (called "MO") from your provider. Furthermore, you have to define a gateway (within **SMS – General**) and put both ID and password = 1 (as defined in the appropriate script – HTML file in the **<install_dir>/html/sms** directory as internal security).

Example of such an HTML file – **receive_routomessaging_uk.html**:

```
<?php

// PHP script called by Routomessaging (http://www.routomessaging.com/) via HTTP POST to deliver SMSs to IceWarp as e-mails.

// A special thanks to Marko Dukanac from Routomessaging

// Notice in this case that the gateway ID and password of the gateway in IceWarp both have a value of "1".

// More details in F1 help (HTTP Request – Delivery Script section)

// variables accepted by routo

//

// user

// pass

// org (body of message)

// dest (recipient number)

// dcs (default enconding)

// message (body of message)

// Shared libraries

define(SHAREDLIB_PATH, get_cfg_var('icewarp_sharedlib_path'));

include(SHAREDLIB_PATH . 'api/api.php');

//     Build $url with gateway's parameters to pass to IceWarp API

//id=1   ==> id of Clickatell gateway in Icewarp SMS/General admin

//pass=1 ==> password defined in Icewarp SMS/General admin

$url =
    'number=' . urlencode($_REQUEST['org']) .
    '&id=1' .
    '&pass=1' .
    '&data=' . urlencode($_REQUEST['message']);

// Deliver SMS to IceWarp

$api = new IceWarpAPI();

$result = $api->SMSIncoming($url);

// Result

if (!$result) { header('HTTP/1.1 503 Service Unavailable'); }
```

?>

At the HTTP gateway, you have to enable an option that does an HTTP POST to, for example,
http://mail.yourdomain.com/sms/receive_routomessaging_uk.html.

Logging

Every message is registered in IceWarp API log, which has to be enabled. If processing is OK, it returns ***True*** in a log,
otherwise it returns ***False***.

Example:

SYSTEM [1740] 12:10:27 <<< NIL->Create(icewarpserver.apiobject)

SYSTEM [1740] 12:10:27 >>> (0 ms) NIL->Create(158606136)

SYSTEM [1740] 12:10:27 <<< TAPIObject(158606136)-
>SMSIncoming(number=5511999999999&id=1&pass=1&data=SMS+for+%23emailaccount%40mydomain.com%23.,,,,)

SYSTEM [1740] 12:10:27 >>> (0 ms) TAPIObject(158606136)->SMSIncoming(True)

Technical Specifications

Hardware Modems

SMS gateway supports *Timer*.

For Linux, the *Serial* communication library is used.

SMS gateway supports the *SIM PIN*.

ReadTimeout is set to 16 seconds.

The SMS gateway supports *SMS Center* number.

Modem Recovery

Modem auto-recovery is supported. If more than two check calls fail, the modem is re-initialized and reopened. This fixes a problem with unplugged modem and plugged back in. SIM PIN has two second sleep applied, device close – 1 second sleep. If SMS sending fails, waiting for a new event is initiated (loop preserved).

The recovery mode fires up only if no data is read, the ERROR response does not trigger it.

- Enhanced modem communication logs – the complete message size is possible to find in logs [nnn].

- Can handle *+CME ERROR:* and *+CMS ERROR:* responses.

- GSM modem auto-recovery – if SMS cannot be sent 3 times then modem is re-initialized.

- *AT+CMGF* is a part of device initialization (applies to READ and SEND).

- Handles the *+CMS ERROR: 304* properly – deletes the SMS and removes it from the queue.

- *Ctrl+Z* is sent when initializing the device (to recover from receive mode).

Selective Gateways

Gateways have their IDs that you can use to send SMS messages through a specific gateway by specifying the URL parameter *&id=<ID>* or selecting the gateway in the GUI where available. **SMS Service – Logging** – each log contains the gateway ID prefix, so you can see which gateway the log belongs to.

Content filters

SMS gateway supports content filters for incoming and outgoing messages. Any existing SMTP rules are converted to XML format during upgrade to version 10.

This allows deeper system integration by the means of *Execute application*, *Execute SQL Statement* actions.

SMS gateway directs incoming messages without any body to trash.

SMTP Transport

Also on each gateway there is the *From:* edit which lets you rewrite the *From:* address of the SMS email. This allows you to reply to received SMS emails and the reply will be sent directly to the SMS sender.
Each incoming SMS to e-mail has the *To:* header set with the value of the final recipient (not applied if Rules are used to deliver message).

Non-Delivery Reports for text messages sent via email or WebClient work exactly the same way as SMTP NDR: a warning if message is still not delivered after some time (the *General – Delivery – Warning* option) and if the message cannot be delivered and expires (SMS API setting) you will receive the final NDR on message delivery failure. The original text is always included. This applies only if the real sender is known (sent from WebClient, email or specified in the SMS API).

In the **Incoming Rules** tab, there are two options. The first one checks for *#email#* inside the SMS and if found, the SMS will be delivered to that email account. The other option is a smart approach to record all sent messages history (sender and destination number) and if incoming SMS is coming from a phone number we previously sent a message to, the sender from history is read and message is sent to that sender (within a time frame). This makes it possible to send SMS over the SMS gateway and receive an answer to that SMS without defining any rules on the SMS server.

SIP Transport

Rules functions for SIP transport are implemented: METHOD (create a regex restriction to SIP method), STOP (stop processing the SIP packet) and SMS (send SMS with parameters e.g. **"1"** or **"maxmsgs=1"**).

The RESPONSE function is implemented (allows to send your own response to a SIP request). Example of SIP "MESSAGE" command with a SMS gateway:

<NUMBER>^([0-9]).*</NUMBER><METHOD>MESSAGE</METHOD><SMS>1</SMS><RESPONSE>200 OK</RESPONSE>

The SMS rule action integrates smoothly with the SMS service and its authentication just like SMTP or XMPP SMS gateway, that means users already authenticated and granted access to the SMS service do not need to authenticate to SMS service again.

XMPP Transport

XMPP – SMS gateway support is implemented**.** It shares the library with an email gateway (**email.dll**/**email.so**), supports **smsparams** module parameters which will be added to each sent SMS to the SMS service, if destination JID does not contain '**%**' for domain delimiter the destination is not an email address but SMS; e.g. JID: **john%doe.com@email.domain.com** (results in email delivery), JID: **123242342@sms.domain.com** (results in SMS delivery), you can use the email gateway service definition (**email.domain.com**) or create a new **sms.domain.com**.

Queue Handling

Queue mechanism is implemented which ensures that each item not sent gets re-queued as a new item.

If **Use MDA queue for internal message delivery** (**Mail Service – General – Advanced**) is turned on, forwarding to SMS and IM works properly too.

By default, SMS queue expiration is set to 5 days.

International characters

SMS gateway supports both the Unicode and GSM 0338 charsets. Conversion between these two charsets is implemented along with UDH 7bit 00h padding.

Concatenated (long) messages

The gateway allows sending long (concatenated) messages in Binary, Unicode and 8bit formats and also receiving of long messages. SAR support is implemented for:

- message delivery, long messages are automatically supported via the **SMSHTTP** function, if **data=variable** longer then allowed (Unicode 70, 8bit 140, 7bit 160) then the message is automatically split into smaller parts and sent with SAR (message concatenation - long messages).

- incoming messages.

SyncML Push

SMS service can be used for SyncML 1.2 notifications (e.g. SAN via SMS sent to a Funambol SyncML client).

SyncML Push supports both OMA and OTA configuration SMS, allows sending configuration SMS via GUI and also supports SMS gateway ID.

APIs

You can send an SMS via PHP (API->SendSMS), so HTTP gateway can be created easily. A new function – SMSHTTP can be used. It has the second parameter – **BypassAuth** – if authorization is required for SMS Gateway and **Bypass** is true, the request will still be processed.

ValidateAccount() supports SMS.

Sending SMS from Email Clients

First you need to set up a new user account that will auto-forward to the SMS Service. This will be a helper account which will simulate the functionality of the internal **sms:** protocol for use with external clients, by the means of the email address extension. This account should be created in the primary domain so that users do not have to specify its domain name.

In the administration console, create a new user account with the protocol prefix, e.g. **sms**, but any other name will work too.

On the **Options** tab – **Mailbox** section, select the **NULL** option so that messages to this address are not saved.

Into the **Mail** tab – **Forward To:** field, enter the following address: **sms:%%extensionnosep%%**

Now your users can send an SMS simply by sending an email to sms:+15551234567@icewarpdemo.com. The data between the colon (:) and the @ sign will be used as the parameter value in the **sms:** protocol in the following way:

The part before **?** (question mark) is used as destination number, the part after **?** (if any) will be used as parameter value,

e.g.:

sms:%%extensionnosep%%?maxmsgs=7

for email sent to to *sms:+15551234567@icewarpdemo.com* it will mean:

number :+15551234567

params:maxmsgs=7

i.e. it will allow to split the message to 7 parts (by default, only one part is allowed).

> *NOTE: Your users should be aware that to send concatenated (long) messages as multiple SMS messages they should ask their administrators for setting of maximum number of messages (the **maxmsgs** parameters).*
>
> You, and they, should also be aware of any costs incurred to send SMS messages.

SMS Mailing Lists

System users (i.e. non-user accounts) are not allowed to send SMS for security reasons (there is no password associated with these accounts).

However, you can setup a mailing list, so that external users can receive their subscriptions by SMS. You only need to include them as members to the list with username/password parameters of a user who is allowed access (by being an email user with SMS access mode allowed and using SMTP AUTH, or, more typically, through a dedicated SMS user setup in **SMS Service – Authentication**). The SMS limits will be applied to that user instead of the mailing list.

E.g. if you setup the following member of a mailing list, the alerts will be sent to the number you specify and accounted to the user 'smsalerts'.

sms:+15551234567?user=smsalerts&pass=alertpwd@icewarptest.com

> BE AWARE: The authentication parameters can also be entered directly in the 'sms' account **Forward to:** field, however this will have the consequence that anyone will be able to send through the 'sms' account without any authentication!

Outlook and Other Desktop Clients

1. In Outlook, it is always necessary to add a domain as Outlook does not accept addresses without domain parts. (Contrary to WebClient for example.)

2. You need to escape any special characters by enclosing that part in quotes – this means that the **sms:** prefix needs to be enclosed in quotes.

The correct formats of entries to the **To**, **Cc** and **Bcc** fields are as follows:

- *"sms:+15551234567"@icewarp.com*

- *John <"sms:+15551234567"@icewarp.com>*

- *"John Doe" <"sms:+15551234567"@icewarp.com>*

In the case there is not SMTP authentication within the session and the SMS server requires authentication, you can use the following syntax:

"sms:<phone_number>?user=<username>&pass=<password>"@<domain>

e.g.: "sms:+15551234567?user=john&pass=123"@icewarp.com

If whole email addresses are used to login, use this format:

"sms:<phone_number>?user=<alias>%<domain1>&pass=<password>"@<domain2>

e.g.:"sms:+15551234567?user=john%icewarp.com&pass=123"@icewarp.com

> BE AWARE: In this case the **Convert characters % and / to @ in usernames** option must be enabled (**Domains and Account – Policies – Login Policy**).

> NOTE: <domain1> can equal to <domain2>.

Directly from Outlook Contact Lists

This assumes you have already set up an "sms" address for your users.

They should simply set up contacts in their mail clients (MS Outlook, Mozilla Thunderbird, etc.) with the **To:** address set as specified earlier such as ***sms:+15551234567@icewarpdemo.com***.

They can then send text directly to a contact's mobile device number and receive mobile replies back to their mailbox if you have configured the ***Reply-To*** feature on the SMS gateway (see the chapter ***Incoming Rules***).

Directly from WebClient

To send SMS from WebClient, do the following:

1. In the main WebClient menu, select the ***New – SMS*** items. The **Message** window opens.

2. Into the ***SMS*** field, enter the recipient's phone number in the appropriate form (example: ***+15551234567***) or select it from the contact list. (Reveal the **Select Contacts** dialog by clicking the 📖 icon next to the ***SMS*** field. For more details about this dialog, refer to the **GroupWare – Reference – Public Folders – IMAP** chapter – **Sharing Folders** section.)

3. Fill in the ***Subject*** field, write the message and click the ***Send*** menu item to send the SMS.

4. It is also possible to use the ***To***, ***Cc***, ***Bcc*** fields. The recipients will obtain email messages instead of SMS ones.

SMS Synchronization

Native SMS synchronization is supported in the EAS protocol V 14.0 and later. SMS synchronization can be divided into three parts:

- SMS synchronization from a server to a client (SMS add, delete and update) with client delete and update support only (1)
- SMS adding by a client to a server (2)
- SMS sending (3)

Clients need not support any SMS synchronization part or can support following valid subsets of SMS synchronization parts: (1),(1,2),(1,3),(1,2,3). IceWarp Server supports automatic conversion of SMS items to mail items for all clients who do not support SMS the synchronization part #(1). Apple iOS, Google Android (native) and Microsoft Windows Phone do not support SMS synchronization. Some Samsung Android clients support all parts of SMS synchronization. 9Folders Nine client supports only part #(1).

ActiveSync Device as a SMS Gateway

For using a mobile device as a SMS gateway, a client MUST meet three following criteria:

- The mobile device MUST be in the *Allowed ABQ* state.
- The mobile device MUST send either the *Provision* or *Settings* command request to provision the server for sending outbound SMS messages through the mobile device. Outbound SMS messages are sent only through mobile devices that enable it. To enable outbound SMS messages, the *EnableOutboundSMS* element MUST be set to 1.
- The mobile device MUST synchronize the SMS items via the *GetItemEstimate* command or *Sync* one over the user's Outbox folder (= the folder must exist within the device). It SHOULD do so in regular intervals (about every 15 minutes) or use the *Ping* command or its equivalent – *Sync* command with the *Wait* element or the *HeartbeatInterval* element.

IceWarp ActiveSync SMS Gateway Management

SMS gateway management guarantees:

- Only one active SMS gateway can be set at a time.

SMS gateway management perfoms following actions:

- To find an active SMS gateway
- To check whether the found active SMS gateway is valid: meet three previously mentioned criteria (AS Device as a SMS Gateway)
- To find a candidate for a new active SMS gateway, if we have not a valid active SMS gateway
- To add or update the active SMS gateway to the found candidate and send the info email: *New SMS gateway is set*
- To send the info email *No SMS gateway is set* only, in the case we have invalid active SMS gateway and we cannot find a candidate

SMS gateway manager is started only in the following conditions:

- When a device is synchronizing or pinging Outbox via the *GetItemEstimate*, *Sync* or *Ping* commands
- When a device disables outbound SMS messages by sending the *EnableOutboundSMS* element set to 0.