SCALIX

Scalix Server Setup and Configuration Guide

Version 11.0.2

Server Setup and Configuration Guide

Published by Scalix Corporation
1400 Fashion Island Blvd., Suite 602
San Mateo, CA 94404-2061
USA

Product Version: 11.0.2

E: 2.28.2007

Notices

The information contained in this document is subject to change without notice.

Scalix Corporation makes no warranty of any kind with regard to this material, including, but not limited to, the implied warranties of merchantibility and fitness for a particular purpose. Scalix Corporation shall not be liable for errors contained herein or for incidental or consequential damages in connection with the furnishing, performance, or use of this material.

Unix is used here as a generic term covering all versions of the UNIX operating system. UNIX is a registered trademark in the United States and other countries, licensed exclusively through X/Open Company Limited.

Linux is a registered trademark of Linus Torvalds.

Red Hat, and Fedora are registered trademarks of Red Hat Software Inc. rpm is a trademark of Red Hat Software Inc.

SUSE is a registered trademark of Novell Inc.

Java is a registered trademark of Sun Microsystems Inc.

Microsoft, Windows XP, Windows 2000, Windows NT, Exchange, Outlook, Active Directory, and Internet Explorer are either registered trademarks or trademarks of Microsoft Corporation in the United States and other countries.

Restricted Rights Legend

Use, duplication, or disclosure is subject to restrictions as set forth in contract subdivision (c)(1)(ii) of the Rights in Technical Data and Computer Software clause 52.227-FAR14.

Contents

Introduction To This Guide

About This Guide

This guide outlines one-time setup and configuration procedures to get a the Scalix mail system up and running. That includes tasks such as installing anti-virus and anti-spam protection, setting up routing between multiple servers, integrating directories and more.

It is broken down into two sections: Basic setup and configuration tasks that all Scalix system administrators must do, and more advanced setup tasks that are optional.

This guide does not cover frequently-occurring tasks such as creating and managing users, groups, calendars and contact lists. For more information on those sorts of day-to-day administative tasks and ongoing maintenance issues such as backups, recovery and public folder maintenance, see the *Scalix Administration Guide*.

Contents of this Guide

Included in this guide are the following topics:

How to Use This Guide

This guide uses the following typographical conventions:

Table 1: Typographical Conventions

Typographical Convention	Explanation
Buttons	The boldface verdana type indicates a button, a link, a field or any other UI element to click or press as well as a keyboard stroke. For example: Click Finish. Or In the **User-name** field.
Italics	Indicates a directory path, a file or the name of a window or dialog box. For example: Go to */var/opt/scalix.* Or: You see the *Reply* screen.
Code	Indicates a piece of code to write or run. For example: `Launch scalix-installer.sh`
Document Names	References to other documents appear in italic font.
<Angle Brackets>	Values that you need to supply on your own are shown within angle brackets.

Using the CLI

As with any procedure done on the command line, there may be more than one way to accomplish many of the tasks outlined in this manual. In many cases, these procedures are intended only as examples of how to complete a setup or configuration. If another method is more comfortable or more in keeping with your unique setup, it may be the best approach.

In addition, Scalix offers complete man pages for all commands. Please consult them whenever needed.

Identifying the Instance Home Directory

Throughout the various setup procedures, there are repeated references to the instance's home directory, known as "~". The location of this directory varies depending on how you ran your initial setup. For example, if you named the instance when you created it, the home directory becomes /var/opt/scalix/<instance>/s, where <instance> is a two-letter code created from the first and last letter of the instance name. If the instance is unnamed, the home directory becomes /var/opt/scalix/<nn>/s where <nn> is the first and last letter of the host name for that instance.

To determine the home directory for a particular instance, look in /etc/opt/scalix/instance.cfg for the appropriate value of OMDATADIR.

Related Documentation

Other Scalix product manuals include:

- Scalix Installation Guide

- Scalix Migration Guide
- Scalix Administration Guide
- Scalix Client Deployment Guide
- Scalix API Guide
- Scalix Evaluation Guide

In addition, there are online help systems in:

- Scalix Management Console
- Scalix Web Access
- Outlook (if enabled for the Scalix connector)

Introduction To Scalix

This chapter introduces the Scalix system: Its different editions, access levels and licensing system.

Contents

This chapter includes the following information:

About the Scalix System

Capitalizing on a proven technology foundation and the openness of Linux, Scalix gives enterprise customers a simple to manage, highly reliable, and feature-rich Linux email and calendaring platform.This offers superior price and performance advantages with greater security, reliability, performance, openness and flexibility, when compared to other operating and messaging systems.

Based on open standards and a proven email server technology foundation, Scalix enables customers to create a robust and scalable environment that is flexible enough to adapt to their changing needs over time. The Scalix platform scales up to support organizations with hundreds of thousands of users and scales down for offices with fewer than one hundred users, making it a viable alternative for a broad range of organizations.

The Scalix architecture supports virtually any email client and device, without loss of functionality or data integrity. This means full-function support for popular clients like Microsoft Outlook and Novell Evolution, as well as the broad range of POP or IMAP clients available. Users can count on advanced features like enterprise calendaring and scheduling with real-time free/busy lookup, contact and task management, public folders, rich text formatting, offline folder synchronization, secure delegate access to calendar and email, email rules, resource booking and more.

About Scalix Product Editions

Scalix offers three editions of its powerful email and calendaring platform based on Linux and open systems: Scalix *Enterprise Edition, Small Business Edition* and Scalix *Community Edition*.

Scalix Enterprise Edition is the company's flagship product and is ideal for organizations that demand the full range of functionality in a commercial email and calendaring system. It includes multi-server support, unlimited number of *Standard* users, any number of *Premium* users, the full complement of Scalix advanced capabilities, and a wide variety of technical support options.

Scalix Small Business Edition targets organizations getting started with a commercial version of Scalix that do not have the higher end requirements of Enterprise Edition. It is functionally equivalent to Enterprise Edition except that it allows only single-server installations instead of multi-server, and does not include the capabilities for high availability and multi-instance support.

Scalix Community Edition is the free, single-server, unlimited-use version of the Scalix product and is great for cost-conscious organizations that desire a modern email and calendaring system but do not require advanced groupware and collaboration functionality for their entire user population. It includes unlimited Standard users, twenty-five free Premium users, a subset of Scalix functionality, and fee-based, incident-based technical support.

The following table compares the Scalix product editions in greater detail:

Table 1: Product Editions and their Features

Product Feature	Community Edition	Small Business Edition	Enterprise Edition
User Types			
Standard Users	Free, unlimited	Free, unlimited	Free, unlimited
Premium Users	Included: 25 Max: 25	Included: 50 Max: Unlimited	Min Purchase: 25 Max: Unlimited
Core Functionality			
Email & calendaring Server	Single-server	Single-server	Multi-server
Internal user directory	[X]	[X]	[X]
Choice of GUI-based or command line installation and administration	[X]	[X]	[X]
Unlimited POP/IMAP email client access	[X]	[X]	[X]
Native MS Outlook support (via MAPI)	Premium users only (max 25)	Premium users only	Premium users only
Fully functional AJAX web client (Scalix Web Access)	[X] (group scheduling in calendar for 25 premium users only)	[X] (group scheduling in calendar for all premium users)	[X] (group scheduling in calendar for all premium users)

Table 1: Product Editions and their Features

Native Novell Evolution support	[X] (group scheduling in calendar for 25 premium users only)	[X] (group scheduling in calendar for all premium users)	[X] (group scheduling in calendar for all premium users)
Public folders	Premium users only (max 25)	Premium users only	Premium users only
High availability	Not available	Not available	[X]
Multiple instances per server	Not available	Not available	[X]
Migration tools	Not available	[X]	[X]
Upgrade To Enterprise Edition	Via license key. Re-installation not required	Via license key. Re-installation not required	Not applicable
Mobile Access	[X]	[X]	[X]
Ecosystem Support			
Meta-directory support via LDAP	[X]	[X]	[X]
iCal support	[X]	[X]	[X]
Native Exchange Interoperability (via TNEF)	Not available	[X]	[X]
Active Directory integration with MMC plug-in	Not available	[X]	[X]
Anti-virus	Via flexible 3rd party interface	Via flexible 3rd party interface	Via flexible 3rd party interface
Anti-spam	Via flexible 3rd party interface	Via flexible 3rd party interface	Via flexible 3rd party interface
Archiving	Via flexible 3rd party interface	Via flexible 3rd party interface	Via flexible 3rd party interface
Wireless email & PIM	Email-only via POP/IMAP	Email & PIM via Notify	Email & PIM via Notify
Technical Support			
Community Forum	Free	Free	Free
Knowledgebase, Tech notes	Free	Free	Free
Incident-based Support	Fee-based	Fee-based	Fee-based
Software subscription	Not available	[X]	[X]
Premium 7x24 Support	Not available	[X]	[X]
Cost			
Licensing	Free, unlimited use	$995 for First 50 Premium Users	Per-user License; No Per-server Fees

About Scalix User Types

Scalix users can be defined as *Standard* or *Premium* users, as defined in the following:

Standard Users

Standard users gain access to a subset of Scalix functionality including email, personal calendar and contacts through Scalix Web Access and Novell Evolution as well as email access using POP/IMAP clients. The ability to deploy standard users is ideal for cost-conscious organizations with users who do not have high-end groupware and collaboration requirements. An unlimited number of standard users may be deployed with any Scalix edition for free.

Premium Users

Premium users have access to the full benefits and functionality of the Scalix email and calendaring system. The following Scalix product capabilities are available only to premium users:

- Native MS Outlook support (via MAPI)
- Group scheduling functionality including free/busy lookup in Outlook, Scalix Web Access and Evolution clients
- Access to public folders
- Wireless email and PIM

Any number of licensed premium users may be deployed with Scalix Enterprise Edition. Scalix Community Edition is limited to a maximum of twenty-five (25) free premium users, who enjoy many of the features available to Enterprise Edition premium users.

Flexible, Cost-Effective Email For Everyone

The distinction between standard and premium users provides organizations with the flexibility to cost-effectively provide email for all users. For example, manufacturers and retailers may desire headquarters staff to be designated as premium users as they require advanced groupware capabilities, while less demanding users, such as shop floor or store personnel, would be satisfied as standard users with only email and personal calendaring capabilities. Similarly, educational institutions may decide that faculty and staff are premium users that need advanced collaboration capabilities while students are standard users that just need email and personal calendaring. There is no cost for deploying standard users with either Scalix Community Edition or Scalix Enterprise Edition.

Required Licenses

Scalix *Community Edition, Small Business Edition* and *Enterprise Edition* use the same installer. The main difference is that Small Business Edition and Enterprise Edition require a license key while Community Edition does not. Additionally, if you are a Scalix Community Edition customer, you can only perform the “typical” installation, in which all the Scalix components are stored on a single host computer.

To activate your Scalix system as either a Small Business or Enterprise Edition system, you must enter a license key at a strategic point in the installation process. Please obtain your Scalix license key and have it ready for use before installing.

You may proceed with the installation without a license key, however, your system is treated as a Community Edition system and your users as Standard users until the correct license key is entered by means of the *Scalix Management Console*.

Additionally, you can install Scalix Enterprise Edition onto a single host, or distribute the primary components onto separate hosts—both of which are detailed fully in this guide.

Scalix Architecture

This chapter introduces the Scalix architecture: Its components, ecosystem, clients and more.

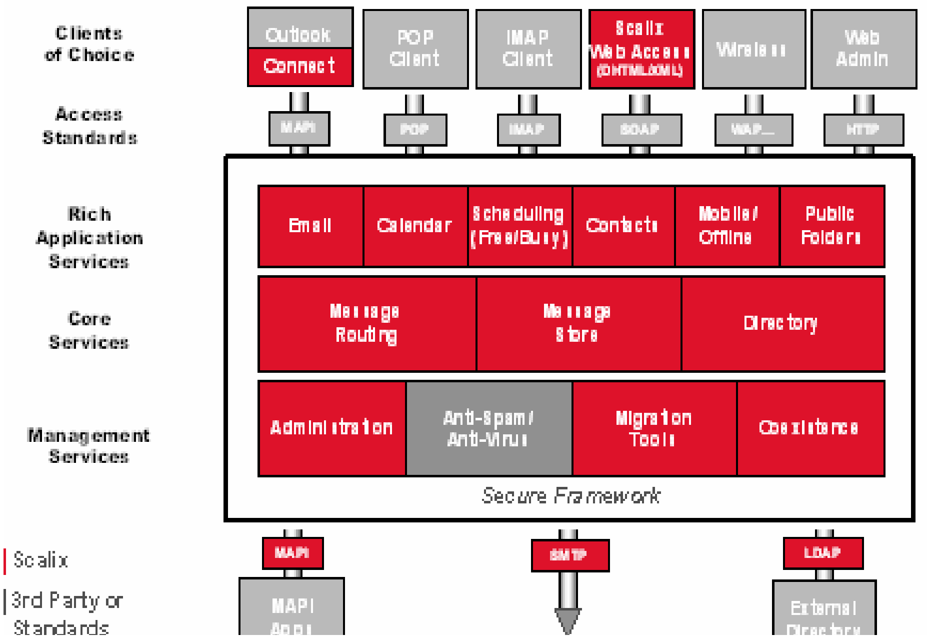
Contents

This chapter includes the following information:

About Scalix Architecture

The Scalix mail system is a client-server architecture based upon international standards and an open architecture that allows the flexibility to use many different client and third-party applications to send and receive messages between multiple Scalix servers, either inside or outside a company’s network.

A Linux operating system environment establishes the base for the actual Scalix platform.



Scalix Components

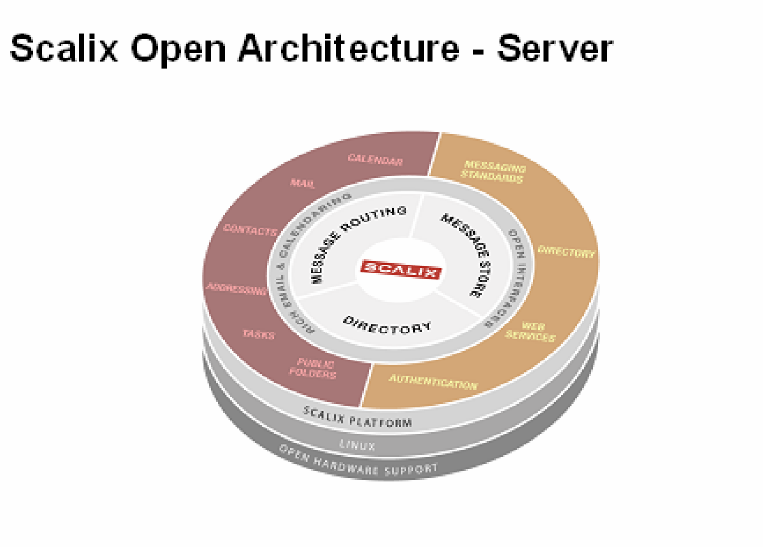
Scalix has two main components, the server and its clients:

The Scalix Server(s)

The Scalix server provides message storage, collection, dispatch, routing and delivery. It not only contains eMail messages, but also PIM/Groupware information such as calendaring data, contacts and task lists. In addition, it manages message delivery and provides or integrates with add-on services such as virus scanning, anti-spam or content-type conversion.

Server management is done in two places:

- Through the Scalix Management Console (aka SAC or Scalix Management Console), an easy-to-use GUI for frequently-undertaken, day-to-day tasks such as creating users, managing public distribution lists, assigning permissions and more.
- On the command line for more advanced configurations such as backups, integration of anti-virus and anti-spam applications, setting up authentication, etc.



Clients

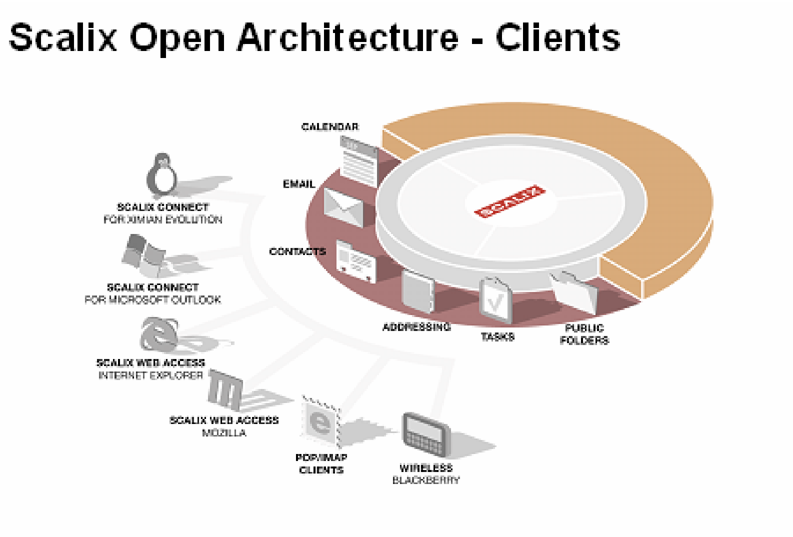
The clients are applications that allow users to create, view and manipulate messages, notify users when new mail arrives, access address directories, track the progress of message delivery, configure auto actions and more. They use the IMAP, POP and UAL (User Access Layer) protocols to connect into the Scalix server, where they access the message store, directory and personal folders. They are handled by remote client interfaces.

Scalix operates seamlessly and transparently with many different clients, including:

- Microsoft Outlook
- Novell Evolution
- IMAP and POP clients such as Mozilla Thunderbird, Outlook Express and Eudora
- Its own native client, Scalix Web Access (SWA).

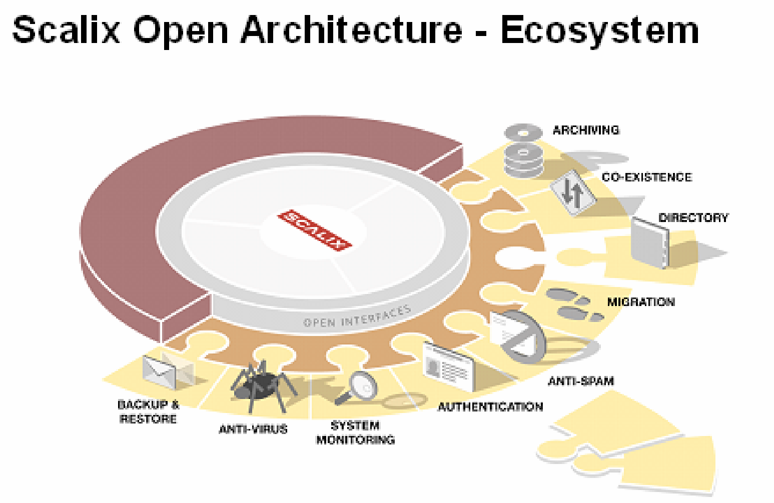
Client management is done in five places:

- The Management Console to set access levels, global server properties and more
- CLI to set access levels and more
- Configuration files to set properties, logging customizations and more
- Scalix Connectors to enable the use of the Scalix server with clients such as Microsoft Outlook and Novell Evolution
- 3rd Party Administration Interfaces


Scalix Open Architecture - Clients
SCALIX
CALENDAR
EMAIL
CONTACTS
ADDRESSING
TASKS
PUBLIC FOLDERS
SCALIX CONNECT FOR XIMIAN EVOLUTION
SCALIX CONNECT FOR MICROSOFT OUTLOOK
SCALIX WEB ACCESS INTERNET EXPLORER
SCALIX WEB ACCESS MOZILLA
POP/IMAP CLIENTS
WIRELESS BLACKBERRY

The Ecosystem

The ecosystem surrounding the Scalix server places a strong emphasis on open interfaces. This provides flexibility for integrating with a variety of best-of-breed solutions in important areas such as anti-virus protection, authentication, backup and recovery tools. The system broadly complies with messaging standards ranging from RFC 822 and continues to include MAPI, POP3, IMAP4, MIME, SMTP, and LDAP.



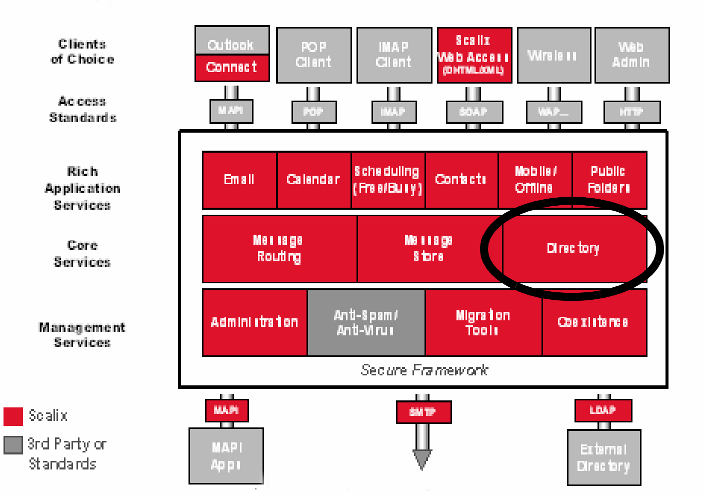
Directories

Scalix also holds a directory of known users that enables auto-finishing features for addressing of email messages.

The address directories are databases that clients use to look up names and addresses. The Scalix address directories can contain Scalix and non-Scalix users, other administrator-configurable information such as job titles and phone numbers, and can be shared with other Scalix directories or synchronized with MS Exchange servers.

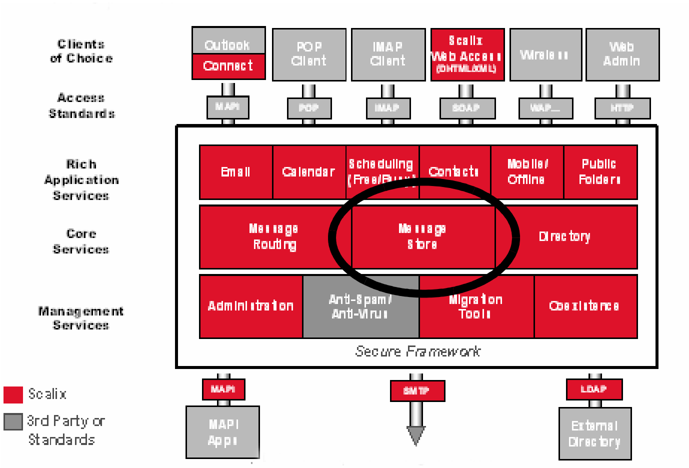
Directories are searchable by any number of attributes. They contain many standard attributes and also some that are rarely used.

Note	Scalix is a system that has grown up over time and it's good to note that it used to be (and still is to some respect) based on X.400. Addresses are still based on X.400 OR names and the X.400 nomenclature.



The Message Store

The message store is not a database. It's a collection of flat Linux files, held in file system directories on the Scalix server. It holds new messages received as well as messages in transit For clients that use the message store (server-based clients), it also holds old messages that are files for reference in folders, copies of outgoing messages, draft messages, in preparation, private distribution lists, personal information such as calendaring, tasks and journaling information and Bulletin Boards, or public folders, which are accessible to multiple users.



Routing and Local Delivery

The Service Router is the process on the Scalix Server that decides (or routes) where a message is supposed to go.

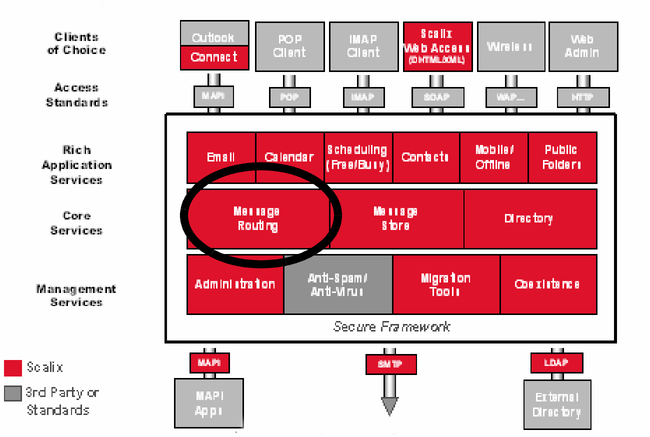
The Local Delivery process is the process on the Scalix Server that determines where a message ends up in the message store on the local machine for a local user.

Scalix's routing services check the recipients in a message, and then send it on to be either delivered locally to another Scalix system, or to leave Scalix entirely via a gateway. These routing services also create NDNs, or Non Delivery Notifications, when a message cannot be delivered due to an addressing fault. These NDNs go to the originator of the message as well as to the configured error manager.

Once the message has arrived at its destination, the local delivery process places it in the recipient's in tray.

Local Delivery and Service Router, together, also handle Public Distribution List Expansion and address resolution, up to the point where they can try to correct misspelled email addresses by phonetic matching.

As all messages in the system must pass through the Service Router, this also becomes the preferred point of integration for virus scanners, filtering rules and message archiving.

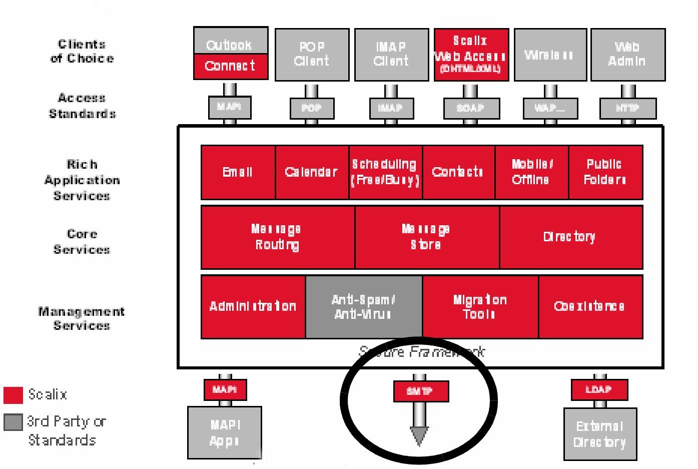


Gateways

Gateways are a way of passing messages out of the Scalix network to different mail environments. The gateway must convert the outgoing message from a Scalix format to one that an external service can send, and then convert the addresses into a format that the target environment can handle, such as an SMTP address.

Scalix comes with a standard SMTP gateway that converts Scalix formatted messages to SMTP formatted messages and vice-versa. This gateway is called the “Unix Mail Gateway” or “Internet Mail Gateway” on Scalix, but, because SMTP is the most important standard in messaging interoperability, it also connects to almost any other messaging system.

Other gateways can be written for connection to other mailing systems.



Transports

The Transport Service on the Scalix server is called the “Sendmail Interface”.

Transports are services that Scalix uses to pass Scalix-formatted messages to other Scalix services. Scalix uses Sendmail and SMTP format to send messages between servers in the Scalix network, but other connections can be written.

Search and Index Service

The Scalix Search and Index Service provides realtime indexing of all private and public folder messages. Built on the open-source Lucene technology, it enables sub-second, mailbox-wide message retrieval. It is localizable, and its Web services interface is available.

Messaging Service

Scalix Messaging Services are server-based REST APIs for email and calendaring application integration. They enable integration of Linux messaging with critical applications such as content management, mobile solutions, customer relationship management (CRM) software, or enterprise resource planning (ERP) packages. Calendaring functions and data can be integrated directly into other applications, or the data from other applications can be directly integrated into email and calendaring.

Basic Setup and Configuration

About This Section

The tasks outlined in the next two chapters are of a more “basic” nature and apply to all setups. They include external authentication, and integration of anti-spam and anti-virus software.

This Section’s Contents Include:

Included in this section are the following topics:

- “Verifying Your Setup and Installation” on page 21
- “Virus Protection” on page 23
- “Spam Protection” on page 32
- “Authentication” on page 52
- “Securing Scalix” on page 71

Verifying Your Setup and Installation

This chapter outlines a series of tests you should run to ensure the system is up and running smoothly before beginning further setup and configuration tasks.

Contents

This chapter includes the following information:

Overview

Before beginning any of the setup and configuration tasks outlined in this guide, test your system to ensure that you have proper connectivity and communication between all servers and clients.

Testing Connectivity with Outside Systems

The first verification is whether you can send and receive messages to clients outside the Scalix system.

To test whether you have connectivity:

1. Using the Scalix Management Console (SAC), create a user. For more on how to use the Management Console, see the SAC online help system.
2. Log in to the Scalix Web Access interface (SWA) as that user. For more on how to use Web Access, see the SWA online help system or the *Scalix Client Deployment Guide*.
3. From that account in SWA, send a message to an outside user such as your ISP address. For more on how to compose and send a message, see the SWA online help system.
4. Log in to the outside account to verify that the message arrived.
5. Reply to the message and check SWA to make sure that the message also arrived.

Testing Connectivity Inside the System

The second verification is whether you can send and receive messages inside the Scalix system.

To test for connectivity inside the system:

1 Using the Scalix Management Console, create a second user.

2 Log in to the SWA interface as the first user and second a message from the first user to the second.

3 Sign in to SWA as the second user and verify that the message arrived.

4 Reply to the message.

5 Go back to SWA as the first user and check that the reply arrived.

Virus Protection

This chapter covers virus protection on Scalix. It describes which anti-virus products Scalix works with, how they work, and how to install and configure them.

If you do not intend to use virus protection, skip this chapter.

Contents:

This chapter includes the following information:

Anti-Virus Overview

The Scalix virus protection framework integrates with the following third-party anti-virus applications to enable scanning within the service router, which is superior to "gateway" solutions because it also scans internal email:

- Clam Anti-Virus
- McAfee VirusScan for Linux
- Trend Micro InterScan VirusWall

The Scalix framework accomplishes this by extending message delivery rules to include additional rulesets and a special "mapper" script that detect and delete infected messages.

How It Works: Message Delivery Rules

Scalix runs anti-virus software as a set of rules. The rules tell the service router to test a message and carry out specific actions based on the results. In the case of anti-virus software, the most effective rule is simply: If infected, delete the message.

The way Scalix is set up, rules are contained in "rule sets," which are text files located in the directory *~/rules* in a file to be named ALL-ROUTES.VIR. Each rule set can be associated with one or more Scalix routes. However, the virus scanning rule set applies to all routes.

How it Works: Mapper Scripts

All incoming messages go through the service router, which you configure to perform virus-scanning tasks based upon those rules. The router instructs a “mapper” script (omvs-can.map) to invoke the third-party anti-virus software, which performs the scan then returns the results to the router.

If the anti-virus software detects a virus, the service router refers to the rule sets and they determines whether the message should be discarded.

Performance Considerations

- Do not forget to turn off or lower logging and auditing levels once installation and configuration testing is done as high logging and auditing levels impact performance.
- Virus scanning adds a performance overhead, but because the architecture keeps the scan script running at all times, the overhead rarely creates performance problems.
- If you do detect an impact on performance, a binary version of the mapper script (or one written in Perl) may help.
- To prevent the service router from scanning text-only or distribution list message parts, set the option SR_VS_IGNORE_ITEM_TYPES in the file *general.cfg*. This option contains a colon-separated list of Scalix body types that do not need virus scanning. For example: 1166:1167 (Distribution Lists and Text Files)
- A list of the Scalix body types can be found in the file ~/nls/C/filetype.

Integration Overview

The basic process for installing, configuring and testing an anti-virus application on your Scalix server includes:

- Acquire and install the anti-virus engine
- Set up the service router rules file to run messages through a third-party anti-virus engine
- Set up and configure integration through a mapper script
- Restart the service router to activate these configuration changes

Each of these processes is outlined below.

Installing Anti-Virus Protection

Installing ClamAV

The easiest and most cost-effective method of anti-virus scanning is the ClamAV program. ClamAV is an open source package available for RedHat and SuSE Linux. Scalix provides hooks for it so that any message passing through the Scalix system is automatically scanned for viruses.

Prerequisites and Versions

Before installing, make sure you have one of the following required versions:

- Redhat Linux EL 3.2.3-34
- SuSE SLES 9
- Scalix Server 9.1.0.81 or above
- Clamav-0.80-1.1 or above
- Clamd-0.80-1.1 or above
- Clamav-db-0.80-1.1 or above

Installing

The first step in provisioning ClamAV on a Scalix server is to install the software packages. Follow the ClamAV documentation with these exceptions:

To install ClamAV on Scalix:

1 Download and install the ClamAV RPMs.

These RPMs are readily available on the Internet and can be easily located using one of the rpmfind web sites.

```
rpm –iv clamav-db-0

rpm –iv clamav-0

rpm –iv clamd-0
```

2 Once the rpms are installed, a new user and group is created. On RedHat, it is called “clamav'. On SUSE, it is called “vscan”.

3 Add this new user to the group, scalix, either through the Unix User Manager or by editing the file */etc/group* by appending clamav or vscan to the scalix entry.

Note	On some versions of SuSE, simply adding the user to the group file doesn’t give the user the required group rights. If so, change the group for the vscan user to be the group, scalix. To do this, edit the file */etc/passwd.*

Installing Other Anti-Virus Programs

If you prefer to use an anti-virus program other than ClamAV, you can. To install other anti-virus software for use with Scalix, follow any manufacturer installation instructions with one exception: Because the service router calls the virus scanner while running as the Scalix user, you must change the permissions on the virus scanner.

To change permissions on the virus scanner:

1 On the virus scanner, run the following lines:

For Trend: `chmod a+rx /opt/trend/ISBASE/IScan.BASE/vscan`

For McAfee: `chmod a+rx /usr/local/bin/uvscan`

Configuring Anti-Virus Protection

Configuring ClamAV for Use with Scalix

If you are working with ClamAV, follow the documentation provided with ClamAV to configure the software with these exceptions:

- On SuSE, the configuration file is sometimes called */etc/clamav.conf.* This name can cause problems because clamd cannot parse the configuration file. Instead, name the file *clamd.conf* and place it in the subdirectory */etc*. Then edit the file */etc/init.d/clamd* by modifyin the start section so that the clamd daemon starts with the added parameter "-c /etc/clamd.conf".
- Verify that the "User" parameter in the file *clamd.conf* is set to the same user you added to the scalix group above.
- Configure the freshclam software to keep the known virus database up to date. Follow the ClamAV documentation to configure freshclam to run as either a daemon or via a cron job.

Creating a Virus-Scanning Ruleset

Regardless of the anti-virus package you use, the first step is to create a virus-scanning rule set.

To do this, create a file in the directory *~/rules* called ALL-ROUTES.VIR, which controls virus protection on the Scalix server. This file contains a message delivery rule set that applies to all routes. If the mapper script detects a virus, the service router refers to these rule sets. They determine whether the message attachment is discarded.

You can use two attributes in virus scanning rules:

- VIRUS-FOUND: Causes the service router to test each message for the presence of viruses.
- VIRUS-UNCLEANED: Causes the service router to test each message for the presence of viruses, and then if needed, remove the infected attachments.

An example of a rule set is:

```
VIRUS-UNCLEANED=1 ACTION=REJECT NDN-INFO=!ndninfo.txt

VIRUS-UNCLEANED=0 VIRUS-FOUND=1 ACTION=ALLOW NOTIFY="A virus was
fuond in your message. It was successfully cleaned and sent to the
recipient. However we highly recommend that you install or update
your virus protection software and scan your computer for viruses."
```

Where...

The first line describes the action the anti-virus software takes if a virus is detected, but the virus cannot be cleaned. In this example, the message is rejected and a non-delivery notification goes to the sender.

The second line describes what action the anti-virus software takes if a virus is detected and the virus can be cleaned out. In this example the rule allows the message to be delivered to the recipient, and a notification is sent to the originating address.

Alert	Each rule must be on a single line and there cannot be any blank lines.

To create the virus scanning ruleset:

1 Determine whether you want the service router to:

- Repair and deliver the infected message
- Prevent the delivery of infected messages

This choice determines which virus scanning attribute you use in the rule set.

2 Create a text file containing the virus scanning rules you want to use. Each rule is a single line of text as shown below:

```
message-attribute=mvalue action-attribute=avalue action-
attribute=avalue ...
```

where *message-attribute* is either VIRUS-FOUND or VIRUS-UNCLEANED and *mvalue* is a numerical value specifying the number of viruses detected/uncleaned. Enter 0 to indicate none, or enter 1 to indicate one or more.

action-attribute and *avalue* can be one of the following:

```
ACTION=ALLOW
ACTION=DISCARD
ACTION=REJECT
ACTION=DEFER
ACTION=RETURN
```

3 Name the file ALL-ROUTES.VIR (all upper case) and save it as a text file to the directory ~/rules.

4 Restart the service router:

```
omoff -s sr
```

5 `omon -s sr`

Note	After starting the service router, a test is done to ensure that the virus scanner can access a Scalix-owned file. If not, the router aborts. Check the event logs (omshowlog) and use the debug logging configured in the file ~/sys/omvscan.cfg.

Configuring Non-Delivery Notification

You can send a non-delivery notification to the address where the infected file originted, but because most viruses come from spoofed addresses, Scalix does not recommend this.

Copying and Modifying the Scan File

Regardless of the anti-virus software you select, the next step in anti-virus configuration is to copy and modify the scan file (omvscan), which provides the necessary information for the anti-virus software to scan all messages sent to Scalix users, even messages sent from one Scalix user to another.

To configure the scan file:

1. Enable the script in the scan file by copying the *omvscan.map* file from ~/examples/general, where the Scalix installation wizard put it, to *~/rules*, where it becomes active.
2. Make sure the file is owned by root and has permissions set to 555.

```
cp /opt/scalix/examples/general/omvscan.map ~/rules

chown root omvscan.map

chmod 555 omvscan.map
```

Setting Up the Mapper Script

Next, you must set up a mapper script. The omvscan.map is the virus scanning mapper script that links Scalix and the following third-party virus scanning applications:

- McAfee VirusScan for Linux
- Trend Micro InterScan VirusWall
- Clam Anti-Virus

The omvscan.map is enabled when the service router process begins upon startup. The script remains active (enabled) until the service router is shut down. If you configure auxiliary service router processes, each service router process starts its own instance of omvscan.map.

Setting up the Mapper Config File

Finally, set up the mapper configuration file. The configuration file **omvscan.cfg** defines the anti-virus application to scan messages and defines the various options to be used. This file is located in the directory ~/sys.

The mapper script config file must be set up as *~/sys/omvscan.cfg*. A sample file is provided in */opt/scalix/examples/general*.

The default omvscan.cfg file looks like:

```
ANTI_VIRUS_ENGINE="ClamAV"

OMAV_LOGFILE=$(omrealpath '~/logs/omvscan.log')

# 0 is off, 1 is ERRORS, 2 is ERRORS & WARNINGS, 3 is same as 2 +
DEBUG

OMAV_LOGLEVEL=0


[Trend Micro InterScan VirusWall]
```

```
TREND_ENGINE=/opt/trend/ISBASE/IScan.BASE/vscan

TREND_SCAN_OPTIONS='-p/etc/iscan -v0 -za'

TREND_CLEAN_OPTIONS='-p/etc/iscan -v0 -za -c'

TREND_LOGPFX=$(omrealpath '~/tmp/trendvs.log')

TREND_USE_LOCKING=no

TREND_LOCK_FILE=trendvs.lock


[McAfee Virus Scan]

MCAFEE_ENGINE=/usr/local/bin/uvscan

MCAFEE_SCAN_OPTIONS='--secure --noboot --mime'

MCAFEE_CLEAN_OPTIONS='--secure --noboot --mime --norename -c'

MCAFEE_LOGPFX=$(omrealpath '~/tmp/mcafee.log')

MCAFEE_USE_LOCKING=no

MCAFEE_LOCK_FILE=mcafee.lock


[ClamAV]

CLAMAV_ENGINE=/usr/bin/clamdscan

CLAMAV_SCAN_OPTIONS='--stdout'

CLAMAV_CLEAN_OPTIONS='--stdout'

CLAMAV_LOGPGX=$(omrealpath '~/tmp/clamav.log')

CLAMAV_USE_LOCKING=no

CLAMAV_LOCK_FILE=clamav.lock
```

The configuration file includes a “[GENERAL]” section followed by application-specific sections. The options you configure in the “[GENERAL]” section apply only to Scalix and not to any virus-scanning application. The following table lists the parameters in the omvscan.cfg file.

Table 1: Parameters in the Anti-Virus Scan File

Parameter	Option
ANTI_VIRUS_ENGINE	Specifies the virus scanning engine. Must correspond to one of the following: [Trend Micro InterScan VirusWall] [McAfee Virus Scan] [ClamAV]
OMAV_LOGFILE	Specifies the file to which the omvscan.map script outputs logging information. NOTE: The service router also logs information to various other Scalix log files. You can set service router logging using the command `omconflvl`.

Table 1: Parameters in the Anti-Virus Scan File

Parameter	Option
OMAV_LOGLEVEL	Specifies the amount of logging information the mapper script (omvscan.map) outputs: 0: No logging (default) 1: Error conditions only 2: Warnings and errors 3: Debug information (such as responses from the scanning application), and warnings and errors. This is the maximum logging level.
{*product*}_ENGINE	Specifies the location of the command-line virus scanning application.
{*product*}_SCAN_OPTIONS	Specifies the scanning options sent to the virus scanning application.
{*product*}_CLEAN_OPTIONS	Specifies the cleaning (repair) options sent to the virus scanning application.
{*product*}_LOGPFX	Specifies the name of the log file to which the temporary output from a virus-scanning application is stored (when the application is invoked to scan or clean a file). This log file is parsed by omvscan.map to get information (such as the name of the virus) if an attachment is infected.
{*product*}_USE_LOCKING	Specifies whether to force omvscan.map to pause while another instance of omvscan.map is already using the virus scanning application to perform a request. All three third-party anti-virus applications can have multiple instances operating at the same time, respectively. For increased Scalix Server performance, the default for this parameter is set to "`NO`".
{*product*}_LOCK_FILE	If you enable virus-scanning application locking, this parameter specifies the name of the lock file used by the one or more instances of omvscan.map (which are sharing access to the virus-scanning application).

Table 2: If you install virus scanning software to its default location, you only have to modify the parameter "ANTI_VIRUS_ENGINE".

General.cfg options

The following optional modification to the `general.cfg` file is useful:

```
SR_VS_IGNORE_ITEM_TYPES
```

You can create a colon-separated list of file codes to exclude from virus scanning. For example, setting this parameter to `1166:1167` excludes Scalix distribution lists and text file (**.txt**) attachments from scanning. This can improve router performance.

Testing

Before moving on to the next step, test the anti-virus installation:

To test your ClamAV installation and configuration:

1 Turn up audit logging for the service router.

```
omconfaud router 13
```

2 Turn up debug logging for the service router.

```
omconflvl router 15
```

3 Stop/restart the service router

```
omoff -d 0 rtr
```

```
omon rtr
```

4 If you download the source tarball from the ClamAV site, attach some of the files provided in the test subdirectory of your ClamAV installation.

5 Look in the file ~/logs/audit log, where you see something like:

```
message-filter-info +VIRUS-UNCLEANED=REJECT
```

6 If that doesn't provide the information you need, check the log *~/logs/fatal* for something like:

```
504 anti-virus engine "ClamAV" exhibits unexpected behavior
```

It is likely the clamd user does not have sufficient permissions to access the subdirectory *~/data*. If so, see the information above to insure that the clamd/vscan user is configured properly.

7 Once you have confirmed that ClamAV is working properly, reduce log levels to 7.

```
omconfaud router 7
```

```
omconflvl router 7
```

```
omoff -d 0 rtr omon rtr
```

Microsoft Outlook Security Model

The Outlook E-mail Security Model provides protection against viruses that come in to a user's Inbox as attachments. For more on how to set this up, see the Scalix Client Deployment Guide.

Spam Protection

This chapter covers how to integrate SpamAssasin into your Scalix system. If you do not intend to use spam protection, you can skip this chapter.

Contents

This chapter includes the following information:

Overview

When protecting against spam in the Scalix environment, there are several approaches:

- Install any anti-spam package on a server that sits in front of the Scalix system. This method is not covered in this manual.
- Install the free, open source SpamAssassin package in the Scalix system, itself.
- Set up a DNS blacklist of IP addresses known to be the source of spam.

How Spam Assassin Works

When SpamAssassin is installed, client processes communicate with a daemon (spamd) to check whether messages are spam. In most cases, the client hands the daemon a complete message to check. The daemon then returns the message with a series of header lines which indicate the "spamminess" of the contents.

For more information about Spamassassin, refer to the Apache Foundation website; http://spamassassin.apache.org.

Integrating Spam Assassin

The traditional route for outside mail to get in to the Scalix system is through the SMTP relay. So the best place to integrate SpamAssassin is there. This way, it routes messages through Sendmail first to be filtered.

There are four basic procedures involved in integrating an anti-spam scanner. They are:

1. Acquire an anti-spam product
2. Install the product on the Scalix server
3. Add one option to the smtpd.cfg file
4. Configure Sendmail for use with SpamAssasin
5. Configure SpamAssassin to start up upon boot

These steps are explained in greater detail throughout this chapter.

Versions and Prerequisites

- SuSE 9.3 Professional
- perl-spamassassin-3.0.4-1.1*
- spamassassin-3.0.4-1.1*
- spamass-milter-0.3.0-1**
- sendmail-8.13.3-5*
- sendmail-devel-8.13.3-5*

* Included with standard SuSE distributions

** Must be downloaded separately from the Scalix software

Installation

Download and install sendmail-devel and spamass-milter RPMs. These RPMs are readily available on the Internet and can be easily located using one of the rpmfind Websites.

Configuration

Now you can configure Scalix to filter mail through Sendmail and then SpamAssassin. This involves adding one option to the smtpd.cfg configuration file.

To configure the smtpd.cfg file for SpamAssassin:

1. Make a copy of the current configuration file.

   ```
   # cp ~/sys/smtpd.cfg ~/sys/smtpd.cfg.orig
   ```

2. Open the configuration file with your favorite text editor. (This example uses vi.)

```
 # vi ~/sys/smtpd.cfg
```

3 Add the following line

```
SMTPFILTER=TRUE
```

Above the line

```
RELAY accept 127.0.0.1
```

4 Save the file.

Configuring SendMail

In addition, there are several Sendmail configurations you must do.

To configure SendMail for use with SpamAssasin:

1 Back up the sendmail.

```
cf #cp /etc/mail/sendmail.cf /etc/mail/sendmail.cf.orig
```

2 Edit the file /etc/mail/sendmail.cf to make the following changes:

Change OperatorChars from:

```
O OperatorChars=.:%@!^/[]+=
```

To

```
O OperatorChars=.:%@!^/[]+
```

3 Uncomment the following line:

```
#O InputMailFilters
```

and change it to:

```
O InputMailFilters=Spamassassin
```

4 Immediately below that line, add the following:

```
#Milter options

#O Milter.LogLevel

O Milter.macros.connect=b, j, _, {daemon_name}, {if_name},
{if_addr}

O Milter.macros.helo={tls_version}, {cipher}, {cipher_bits},
{cert_subject}, {cert_issuer}

O Milter.macros.envfrom=i, {auth_type}, {auth_authen}, {auth_ssf},
{auth_author}, {mail_mailer}, {mail_host}, {mail_addr}

O Milter.macros.envrcpt={rcpt_mailer}, {rcpt_host}, {rcpt_addr}
```

5 In the section MAIL FILTER DEFINITIONS, add the following line:

```
Xspamassassin, S=local:/var/run/spamass-milter/spamass-mil-
ter.sock, F=, T=C:15m;S:4m;R:4m;E:10m
```

Adding SpamAssasin to Startup upon Boot

You must add spamd to Startup upon boot-up and then restart all services.

To add SpamAssassin to Startup upon Boot:

1 Run the following commands:

```
#chkconfig --add spamass-milter

#chkconfig --level 345 spamass-milter on

#service spamass-milter start

#chkconfig --add spamassassin

#chkconfig --level 345 spamassassin on

#service spamassassin start
```

2 Insure all services are now set up to start on bootup.

```
# chkconfig --list|grep 'spamassassin\|spamass-milter'
```

3 You see output that looks like:

```
spamass-milter  0:off   1:off   2:off   3:off   4:off   5:on 6:off

spamassassin    0:off   1:off   2:off   3:off   4:off   5:on 6:off
```

See the chkconfig man page to understand changing run levels

4 Start SpamAssassin.

```
 #service spamassassin restart
```

5 Start spamass-milter.

```
#service spamass-milter restart
```

6 Restart Sendmail.

```
#service sendmail restart
```

7 Restart the Scalix SMTP Relay.

```
#omoff -d 0 smtpd #omon smtpd 6.
```

Confirming SpamAssasin is Working

Before moving on, test that SpamAssasin is working properly.

To confirm that Spamassassin is working ccorrectly:

1 Run the following command:

```
 #tail –f /var/log/maillog
```

2 Successful Spamassassin configuration should produce this type of output in the log file if it is working correctly.

```
scalix.local@MHS>, proto=ESMTP, relay=root@localhost

Nov  3 09:39:56 scal4 sendmail[27547]: jA3Hdueo027547:
from=<Kent.Brake@scalix.com>, size=2089, class=0, nrcpts=1,
msgid=<H00000b60014d0c8.1131039536.hagrid.scalix.local@MHS>,
proto=ESMTP, daemon=MTA, relay=localhost [127.0.0.1]

Nov  3 09:39:56 scal4 spamd[24498]: connection from localhost
[127.0.0.1] at port 59807

Nov  3 09:39:56 scal4 spamd[24498]: info: setuid to root succeeded

Nov  3 09:39:56 scal4 spamd[24498]: Still running as root: user not
specified with -u, not found, or set to root.  Fall back to nobody.

Nov  3 09:39:56 scal4 spamd[24498]: processing message
<H00000b60014d0c8.1131039536.hagrid.scalix.local@MHS> for
root:65534.

Nov  3 09:39:56 scal4 spamd[24498]: clean message (-1.0/5.0) for
root:65534 in 0.1 seconds, 2338 bytes.

Nov  3 09:39:56 scal4 spamd[24498]: result: . -1 -
ALL_TRUSTED,WEIRD_QUOTING

scantime=0.1,size=2338,mid=<H00000b60014d0c8.1131039536.hagrid.sca
lix.local@MHS>,autolearn=failed
```

SMTP Authentication and Spam Protection

Scalix supports SMTP authentication to allow accurate identification of the users of the SMTP service. In addition, Scalix allows you to configure anti-spamming measures to prevent abuse of the Scalix system.

Both of these security measures are implemented as part of the SMTP Relay (omsmtpd), and are configured by adding entries to the SMTP Relay configuration file:

~/sys/smtpd.cfg.

To configure how the SMTP Relay manages incoming connections, you must specify an action that the SMTP Relay performs in response to an event for each address or address pattern.

When an event occurs, the SMTP Relay checks the relevant entries in the configuration file for matching event/pattern entries. The check is done sequentially, from top to bottom. When it finds the first match, the SMTP Relay takes the action specified. If the SMTP Relay does not find a match, it processes the message normally.

The default configuration file included with Scalix causes the SMTP Relay to accept all relay attempts from hosts in the local domain, and reject all unauthenticated relay attempts from outside the local domain.

Table 1: The following table lists possible values for the options in the SMTP Relay configuration file.

Table 2: Options and Values in SMTP Relay Configurations

Event	Description
SUBMIT	An attempt is made to submit a message from the host specified in *pattern*. If the *action* specified is `Header`, the wording of the header inserted is: X-Scalix-Suspicious-Host: *hostname-or-IP-address*

Table 2: Options and Values in SMTP Relay Configurations

Event	Description
ANONYMOUS	An attempt is made to submit a message sent without authentication or after a failed authentication. If the *action* specified is `Header`, the wording of the header inserted is: X-Scalix-Anonymous-Message: from *hostname at date*
AUTH_SUCCESS	An attempt is made to submit a successfully authenticated message. Normally only used with the `Accept` and `Header` actions. If the *action* specified is `Header`, the wording of the header inserted is: X-Scalix-Authenticated-Sender: *email-address; authenticated by hostname at date*
AUTH_MISMATCH	An attempt is made to submit a message which was successfully authenticated, but the originator name does not match the authenticated user name. If the *action* specified is `Header`, the wording of the header inserted is: X-Scalix-Authentication-Mismatch: Message originated from *email-address*, but authenticated identity is *email-address*
RELAY	An attempt is made to relay a message through the SMTP Relay. Normally, you would specify all local hosts in the associated *pattern*, so that they can all send messages to any external host, and any external host can send messages to the local hosts. A relay attempt from the host on which the SMTP Gateway is running is always accepted. The SMTP Relay always inserts a standard `Received:` header in the message, so a `Header` *action* is not required.
ORIGINATOR	An attempt is made to submit a message from a user whose e-mail address matches the *pattern* specified. Use this event to block mail from known sources of spam. If the *action* specified is `Header`, the wording of the header inserted is: X-Scalix-Suspicious-Originator: *email-address*
RECIPIENT	An attempt is made to submit a message to a user whose e-mail address matches the *pattern* specified. Use this event to block mail to nonexistent addresses. If the *action* specified is `Header`, it will be ignored.
SMTPFILTER	SMTPFILTER has only one state and that's TRUE. So, if you add the following line: SMTPFILTER=TRUE to the ~/sys/smtpd.cfg file, it will cause the SMTP Relay to hand the incoming message off to Sendmail. As a result, inbound messages will be processed through any installed sendmail milters (mail filters) such as Spam Assassin.
Action	**Description**
Accept	The message is unconditionally accepted and processed normally.
Defer	The message is deferred, with a `4XX` code. Eventually, the sending host will cease transmitting it, and will reject it.
Discard	The message is accepted but then discarded, with no indication to the sending host that it was not delivered.
Header	The message is accepted, but an extra header is inserted. The wording of the header depends on the *event*.
Reject	The message is rejected, with a `5XX` code.

Table 2: Options and Values in SMTP Relay Configurations

Event	Description
Pattern	Description
Hostname-pattern	The hostname pattern identifies the originating host (or the destination host in the case of the SMTP Relay event). It is used for all events except `ORIGINATOR` and `RECIPIENT`. Possible values are described in the `Hostname Pattern` section.
Email-address-pattern	The e-mail address pattern identifies the source or destination e-mail address, used only for the `ORIGINATOR` and `RECIPIENT` events. For example: `*@*.spam.net` matches `iama.spammer@lotsof.spam.net.` * matches all e-mail addresses.

Note that all actions can be prefixed by the string Log_ to cause the action to be recorded in the Scalix log file. In addition, SMTP Relay information is logged in the Audit Log. See the Audit Log configuration file for more information.

The following information shows the ~/sys/smtpd.cfg configuration file. You configure SMTP Authentication and Anti-spam protection in the second part of the smtpd.cfg file (format: event action pattern pattern ...).

```
##################################################################
#######
# SMTP Relay Configuration
#
##################################################################
##
#
# For details please see Scalix Administration Guide
#
##################################################################
#######

##################################################################
#######
# Relay Configuration
# ###################
#
...

##################################################################
#######
# Authentication and Anti-SPAMming Measures
#
##################################################################
##
#
# Each line is of the form:
# EVENT ACTION PATTERN PATTERN...
# When an event happens the SMTP Relay checks for a matching event/
pattern
```

```
# sequentially in this file. When it finds the first match, it
takes the
# action specified.
#
# ######
# EVENTS
# ######
#
# AUTH_SUCCESS      An attempt is made to submit a successfully
#                   authenticated message.
# AUTH_MISMATCH     An attempt is made to submit a successfully
#                   authenticated message but the originator name does not
#                   match the authenticated name.
# ANONYMOUS         An attempt is made to submit a message sent without
#                   authentication or after failed authentication.
# SUBMIT#           An attempt is made to submit a message from the host
#                   specified in pattern
# RELAY             An attempt is made to relay a msg through the SMTP
#                   Relay
# ORIGINATOR        An attempt is made to submit a message from a user
#                   whose email address matches pattern
# RECIPIENT         An attempt is made to submit a message to a user whose
#                   email address matches pattern

#

# #######
# ACTIONS
# #######
#
# Accept
#               The message is unconditionally accepted and processed normally.
# Defer         The message is deferred with a 400 code
# Discard       The message is accepted but then discarded
# Header        The message is accepted, but an extra header is inserted.
# Reject        The message is rejected with a 500 code

# If Log_ added to the start of an action, then the action is also
recorded
# in the SMTP Relay log file.
#
# ########
# PATTERNS
# ########
#
# Hostname Patterns
# - an IP address, eg 123.234.132.231
# - an IP subnet and mask, eg 123.234.200.0/255.255.240.0
# - a hostname, eg bert.loc.co.uk
# - the end of a domain, eg .spammer.net
# - the start of a domain, 123.234.
# - the keyword ALL matches all hosts
# - the keyword LOCAL matches all hosts that do not contain a .
#
# Email Patterns - used by ORIGINATOR and RECIPIENT
# - *@*.spam.net
#
##################################################################
#######
RELAY accept domain
RELAY reject ALL
#
# extra rules to prevent open relay usage
RECIPIENT Reject *@*@*
RECIPIENT Reject *%*
RECIPIENT Reject *!*
```

There can be several configuration entries for the same event, but only one of them applies to any particular message. For any event, the SMTP Relay scans all configuration entries (from top to bottom) and looks for the first match. Any other configuration entries for this event are ignored.

Note the following:

- AUTH_MISMATCH: this event is an attempt to submit a message which was successfully authenticated but the originator name (FROM: in the RFC 822 header) did not match the authenticated user name.
- Header: In this action, an extra header is inserted into the message. The header name and the value are fixed and depend on the event type.
- Defer: In this action, an SMTP 400 code is returned to the sending server. This means that the message remains on the sending server and is repeatedly retried until it times out and is rejected by the submitting server. This means that the SPAM message occupies disk space on the sending host, but might cause problems for uninvolved third parties.
- Reject: In this action, the message is rejected and an SMTP 500 code is returned to the sending server. For most positively identified SPAM originators and recipients, this is the preferred action since it requires little processing power.
- If debug logging is enabled and any of the action keywords is prefixed with Log_, this action will also be recorded in the SMTP Relay log file, ~/tmp/smtpd.log. You enable debug logging by adding the line debug_log=true to smtpd.cfg.
- Hostname patterns should be used for the ANONYMOUS, AUTH_SUCCESS, AUTH_MISMATCH, RELAY and SUBMIT events.
- If a hostname cannot be looked up in the DNS, it will not match a domain name pattern or an explicit hostname.

- A subnet and mask separated by a / (for example, 15.145.200.0/255.255.240.00) will match all IP addresses in the 15.145.200.0 to 15.145.207.255 range. Note that the mask need not correspond to a "real" subnet.
- A string that begins with an @ character is treated as an NIS (YP) netgroup name. A hostname is matched if it is a host member of the specified netgroup.
- A string that begins with a / character is treated as a file name. A hostname or address is matched if it matches any hostname listed in the named file. The file format is zero or more lines with zero or more hostname patterns separated by white space.
- E-mail address patterns should be used for ORIGINATOR and RECIPIENT events.

How To Prevent Message Spoofing From Internal Hosts

The following example shows an example of a person in the intranet using SMTP commands to send a message to the server and appear to be a user they are not. For instance, you can telnet to a Scalix Server named scalix1, simulate being the user Bob, and send a message to tom:

```
$telnet scalix1 25
Trying...
Connected to scalix1.pwd.scalix.com.
Escape character is '^]'.
220 scalix1.pwd.scalix.com Hello from the Scalix Lab
ehlo scalix1
250-scalix1.pwd.scalix.com Hello scalix1.pwd.scalix.com
[15.145.204.43], pleased to meet
you
250-AUTH LOGIN PLAIN DIGEST-MD5 ANONYMOUS CRAM-MD5
250-AUTH=LOGIN
250-DSN
250 8BITMIME
mail from: bob@scalix.com
250 bob@scalix.com... Sender ok
rcpt to: tom@scalix1.pwd.scalix.com
250 Ok
data

354 Enter mail, end with "." on a line by itself (relay)
subject: an important meeting
Please attend the meeting taking place in the boardroom at 9am
tomorrow
.
250 Ok
```

The message received by Tom appears to be legitimate:

```
Return-Path: <bob@scalix.com>
Received: from scalix1 (scalix1.pwd.scalix.com 15.145.204.43)
by scalix1.pwd.scalix.com via ESMTP; Tue, 17 Apr 2001 14:21:37
+0100 (BST)
Date: Tue, 17 Apr 2001 14:21:40 +0100
From: bob@scalix.com
```

```
Sender: bob@scalix.com
Message-ID: <1642.987513700.scalix1.pwd.scalix.com@MHS>
Subject: an important meeting
MIME-Version: 1.0
Content-Type: text/plain;charset="US-ASCII"
Content-Transfer-Encoding: quoted-printable
Content-Disposition: inline
Please attend the meeting taking place in the boardroom at 9am
tomorrow
```

To prevent such messages from being accepted by the SMTP Relay, you can configure an ANONYMOUS event which instructs the SMTP Relay what to do when an attempt is made to submit a message without authentication (or which failed authentication).

In the current example, you want to reject all anonymous connections from clients in your domain. However, we have a server set up to relay messages, scalixopp.pwd.scalix.com, and it will try to connect anonymously, so you need to allow for this in your configuration:

```
ANONYMOUS Header scalixopp.pwd.scalix.com
ANONYMOUS Reject .pwd.scalix.com
ANONYMOUS Accept ALL
```

Note that the example asks the SMTP Relay to insert a header in anonymous messages which it relays to users from scalixopp. This header takes the following form (where sender is the message sender and addr is the IP address of the sending host):

```
X-Scalix-Anonymous-Message: from sender at addr
```

(In the current example, the addr will be 15.145.205.23 which is the IP address of scalix-opp.pwd.scalix.com.)

Alternatively, you can set up your relaying servers to be within a certain IP range and specify the range using the IP subnet and mask.

When you try using a telnet session to make the message appear to be from "bob@scalix.com", the second configuration line is executed and you are asked to authenticate.

```
$telnet scalix1 25
Trying...
Connected to scalix1.pwd.scalix.com.

Escape character is ’^]’.
220 scalix1.pwd.scalix.com Hello from the Scalix Lab
ehlo scalix1
250-scalix1.pwd.scalix.com Hello scalix1.pwd.scalix.com
[15.145.204.43], pleased to meet you
250-AUTH LOGIN PLAIN DIGEST-MD5 ANONYMOUS CRAM-MD5
250-AUTH=LOGIN
250-DSN
250 8BITMIME
mail from: bob@scalix.com
530 Authentication required
...
```

When you send a message to Tom on scalix2.pwd.scalix.com. It gets relayed via scalixopp, so the SMTP Relay executes the first configuration line and you receive the message with an extra header, showing that this message came from scalixopp (15.145.205.23) and is unauthenticated:

```
Return-Path: <tom@scalix2.pwd.scalix.com>
Received: from scalixopp.pwd.scalix.com (scalixopp.pwd.scalix.com
15.145.205.23)
by scalix1.pwd.scalix.com (Scalix SMTP Relay B.07.00.00)
via ESMTP; Wed, 18 Apr 2001 16:31:04 +0100 (BST)
Received: from scalixopp.pwd.scalix.com (root@localhost)
by scalixopp.pwd.scalix.com (8.9.3 (PHNE_18546)/8.9.3 SMKit7.01
Scalix) with ESMTP id QAA19330 for <tom@scalix1.pwd.scalix.com>;
Wed, 18 Apr 2001 16:31:03 +0100 (BST)
Received: from scalix2.pwd.scalix.com (scalix2.pwd.scalix.com
15.145.204.249)
by scalixopp.pwd.scalix.com (Scalix SMTP Relay B.07.00.00)
via ESMTP; Wed, 18 Apr 2001 16:31:04 +0100 (BST)
Received: from scalix2.pwd.scalix.com (root@localhost)
by scalix2.pwd.scalix.com (8.8.6 (PHNE_17190)/8.8.6) with ESMTP id
QAA03025
for <tom@scalix1>; Wed, 18 Apr 2001 16:31:03 +0100 (BST)
Received: from joyford3 (scalixpwdl86.pwd.scalix.com
15.145.205.60)
by scalix2.pwd.scalix.com (Scalix SMTP Relay B.07.00.00)
via ESMTP; Wed, 18 Apr 2001 16:31:03 +0100 (BST)
Date: Wed, 18 Apr 2001 16:31:02 +0100
From: "tom" <tom@scalix2.pwd.scalix.com>
Sender: "tom" <tom@scalix2.pwd.scalix.com>
To: tom@scalix1.pwd.scalix.com
Message-ID: <001401c0c81c$938a7ca0$62cc910f@pwd.scalix.com>
Subject: some headers
X-MSMail-Priority: Normal
X-Priority: 3
X-Scalix-Anonymous-Message: from <tom@scalix2.pwd.scalix.com> at
15.145.205.23
X-Mailer: Microsoft Outlook Express 5.50.4133.2400
X-MimeOLE: Produced By Microsoft MimeOLE V5.50.4133.2400
MIME-Version: 1.0
Content-Type: multipart/alternative;
boundary="----=_NextPart_000_0011_01C0C824.F520F6D0"
...
```

If you have several hosts set up to relay messages around your intranet, you will need to include them in the ANONYMOUS line to allow them to connect without authenticating.

Remember that for each connection, the SMTP Relay reads down through the configuration file from the top and execute the first line that matches. So the selective line:

```
ANONYMOUS Header scalixopp.pwd.scalix.com
```

must come before the more global line:

```
ANONYMOUS Reject .pwd.scalix.com
```

Verifying the Identity of a Sender

You might want the SMTP Relay to accept all successfully authenticated messages, but for tracking purposes, you want a header added to messages from your domain (pwd.scalix.com), to confirm the identity of the authenticated sender and the address of the sending client/host.

The event, in this case, is AUTH_SUCCESS, the action is Header and the pattern is .pwd.scalix.com. Therefore, you can add following lines to the configuration file:

```
AUTH_SUCCESS Header .pwd.scalix.com
AUTH_SUCCESS Accept ALL
```

This instructs the SMTP Relay to add an extra header (X-Scalix-Authenticated-Sender:) to authenticated messages from any host ending in .pwd.scalix.com and to accept authenticated messages from any other hosts (LOCAL or not in the pwd.scalix.com domain).

For example, you can add the above code to smtpd.cfg on the server, scalix1, and use Outlook on a separate PC (IP Address = 15.145.205.60) to send a message from kelly on scalix1 to Fred on scalix1. Below is the message received by Fred, showing the added header with kelly’s address and the IP address of the connecting machine:

```
Return-Path: <kelly@scalix1.pwd.scalix.com>
Received: from scalix1.pwd.scalix.com (root@localhost)
by scalix1.pwd.scalix.com (8.8.6 (PHNE_17190)/8.8.6) with ESMTP id
LAA03978
for <fred@scalix1>; Wed, 18 Apr 2001 11:21:43 +0100 (BST)
Received: from joyford3 (scalixpwdl86.pwd.scalix.com
15.145.205.60)
by scalix1.pwd.scalix.com (Scalix SMTP Relay B.07.00.00)
via ESMTP; Wed, 18 Apr 2001 11:21:43 +0100 (BST)
Date: Wed, 18 Apr 2001 11:21:41 +0100
From: kelly@scalix1.pwd.scalix.com
Sender: kelly@scalix1.pwd.scalix.com
To: fred@scalix1.pwd.scalix.com
Message-ID: <002f01c0c7f1$5cb78270$62cc910f@pwd.scalix.com>
Subject: hi
X-MSMail-Priority: Normal
X-Priority: 3
X-Scalix-Authenticated-Sender: <kelly@scalix1.pwd.scalix.com> at
15.145.205.60
X-Mailer: Microsoft Outlook Express 5.50.4133.2400
X-MimeOLE: Produced By Microsoft MimeOLE V5.50.4133.2400
MIME-Version: 1.0
Content-Type: multipart/alternative;
boundary="----=_NextPart_000_002C_01C0C7F9.BE468290"
...
```

AUTH_SUCCESS should only be used with the Header action to insert the additional header verifying the user's identity.

Handling Messages With Sender/Authenticated User Mismatch

We looked at how a "spoofer" might try to send an unauthenticated message. The following information describes how to prevent the "spoofer" from connecting to `scalix1`, authenticating successfully, and sending a fake message to kelly.

To prevent this, use the `AUTH_MISMATCH` event. `AUTH_MISMATCH` describes an attempt to submit a message which was successfully authenticated, but the originator name (`FROM:`) did not match the authenticated user name. Such messages should be rejected.

For example, you can add the following lines to **smtpd.cfg**:

```
AUTH_MISMATCH Reject LOCAL
AUTH_MISMATCH Header scalix2.pwd.scalix.com
AUTH_MISMATCH Reject ALL
```

The first line causes all authenticated messages from the local host (no . in the address) to be rejected (with a SMTP 500 response), if the sender does not match the authenticated user.

The second line causes a header to be added to any similarly deficient message from the host, scalix2.pwd.scalix.com. The message is not rejected.

The third line causes any other message with this defect to be rejected.

When you telnet to scalix1, authenticate as tom, and try to send a message as Bob:

```
$telnet scalix1 25
Trying...
Connected to scalix1.pwd.scalix.com.
Escape character is ’^]’.
220 scalix1.pwd.scalix.com Hello from the Scalix Lab
Connection closed by foreign host.
root@scalix1[] telnet scalix1 25
Trying...
Connected to scalix1.pwd.scalix.com.
Escape character is ’^]’.
220 scalix1.pwd.scalix.com Hello from the Scalix Lab
ehlo scalix1
250-scalix1.pwd.scalix.com Hello scalix1.pwd.scalix.com
[15.145.204.43], pleased to meet you
250-AUTH LOGIN PLAIN DIGEST-MD5 ANONYMOUS CRAM-MD5
250-AUTH=LOGIN
250-DSN
250 8BITMIME
auth login
334 VXNlcm5hbWU6
bXIgYWRtaW4=
334 UGFzc3dvcmQ6
YWRtaW4=
235 Authentication successful
mail from: bob@scalix.com
530 Authentication mismatch
```

Scalix found the mismatch and the connection rejected with a SMTP 530 error code.

When you try sending the message by performing a telnet from scalix2 to scalix1, the following occurs:

```
$telnet scalix1 25
Trying...
Connected to scalix1.pwd.scalix.com.
Escape character is ’^]’.
220 scalix1.pwd.scalix.com Hello from the Scalix Lab
ehlo scalix2
250-scalix1.pwd.scalix.com Hello scalix2.pwd.scalix.com
[15.145.204.249], pleased to meet you
250-AUTH LOGIN PLAIN DIGEST-MD5 ANONYMOUS CRAM-MD5
250-AUTH=LOGIN
250-DSN
250 8BITMIME
auth login
334 VXNlcm5hbWU6
bXIgYWRtaW4=
334 UGFzc3dvcmQ6
YWRtaW4=
235 Authentication successful
mail from: bob@scalix.com
250 bob@scalix.com... Sender ok
rcpt to: kelly@scalix1
250 Ok
data
354 Enter mail, end with "." on a line by itself (relay)
subject: an important meeting
Please attend the boardroom meeting tomorrow, 9am
bob
.
250 Ok
```

The message is accepted, but there is a header highlighting the discrepancy:

```
Return-Path: <bob@scalix.com>
Received: from scalix2 (scalix2.pwd.scalix.com 15.145.204.249)
by scalix1.pwd.scalix.com (Scalix SMTP Relay B.07.00.00)
via ESMTP; Wed, 18 Apr 2001 17:51:59 +0100 (BST)
Date: Wed, 18 Apr 2001 17:51:59 +0100
From: bob@scalix.com
Sender: bob@scalix.com
Message-ID: <4752.987612719.scalix1.pwd.scalix.com@MHS>
Subject: an important meeting
X-Scalix-Authentication-Mismatch: originator bob@scalix.com
authenticated as tom at 15.145.204.249
MIME-Version: 1.0
Content-Type: text/plain;
charset="US-ASCII"
Content-Transfer-Encoding: quoted-printable
Content-Disposition: inline
```

```
Please attend the boardroom meeting tomorrow, 9am
bob
```

The IP address is that of scalix2.

Restricting Who Can Use This Server To Relay Mail

It is very important to prevent outside hosts using your server to relay large quantities of unwanted mail to other internal and external hosts. You can use configuration entries for RELAY events to tell the SMTP Relay what to do when a host, which matches the pattern, attempts to send a message through the SMTP Relay to a Sendmail recipient on another server.

Normally, all local hosts should be included so that they will be allowed to send messages to any external host. You can also list here any external hosts which are allowed to use this server to relay.

For example, note the two entries in the default file. These are automatically inserted by Scalix, including the domain of this host:

```
RELAY Accept domain

RELAY Reject ALL
```

This default setup means that any attempt to use this server to relay a message will only be successful if the sending server is in the same domain as this server. As the SMTP Relay always inserts a standard Received: header into the message, the Header action does not make sense for a RELAY event and will be ignored.

Note that SMTP Relay cannot block relay attempts through the Internet Gateway (the message goes into Scalix and is relayed using Scalix routes). See “Internet Mail Gateway” on page 29 for more information.

Blocking Mail From Certain Hosts

Configuration entries for SUBMIT events describe what the SMTP Relay does when a host matching one of the given patterns attempts to submit a message. You can use lines like the examples below to block/log message submission from hosts which are known or suspected of sending large amounts of unwanted mail (spam):

```
SUBMIT Reject known.spammer.net

SUBMIT Log_Reject another.spammer.net

SUBMIT Header possible.spammer.net

SUBMIT Accept ALL
```

the SMTP Relay looks at the address in the MAIL FROM: line of the message and looks through the SUBMIT lines in the configuration file to see if the address matches any of those specified. If the address is known.spammer.net, an SMTP 500 code is returned and the message is not accepted.

If the address is another.spammer.net, an SMTP 500 code is returned, the message is not accepted and, if debug logging is enabled, this action is logged in the log ~/tmp/smtpd.log file.

If the address is possible.spammer.net, the message will be accepted but the following header will be added to the RFC 822 message header to indicate that the message was from a suspect host:

```
X-Scalix-Suspicious-Host: IP address
```

If the message is submitted by any host other than those identified in the lines above, it is accepted.

For example, you determine that a host (scalixopp) might be sending unwanted mail, and you want the SMTP Relay to add a header to any message from that host instructing the recipient that the sender is not trusted. Also, you are certain that scalix2 is sending SPAM and you want to reject connections from scalix2 and have the rejection logged in the SMTP Relay's log file.

Add the following lines to scalix1's Relay configuration file:

```
SUBMIT Header scalixopp.pwd.scalix.com

SUBMIT Log_Reject scalix2.pwd.scalix.com
```

However, you know that messages from scalix2 will be relayed to scalix1 by scalixopp. The following shows what happens when you send a message from Tom on scalix2 to Fred on scalix1:

```
Return-Path: <tom@scalix2.pwd.scalix.com>
Received: from scalixopp.pwd.scalix.com (scalixopp.pwd.scalix.com
15.145.205.23)
by scalix1.pwd.scalix.com (Scalix SMTP Relay B.07.00.00)
via ESMTP; Thu, 19 Apr 2001 09:27:29 +0100 (BST)
Received: from scalixopp.pwd.scalix.com (root@localhost)
by scalixopp.pwd.scalix.com (8.9.3 (PHNE_18546)/8.9.3 SMKit7.01
Scalix) with
ESMTP id JAA20532
for <fred@scalix1.pwd.scalix.com>; Thu, 19 Apr 2001 09:27:28 +0100
(BST)
Received: from scalix2.pwd.scalix.com (scalix2.pwd.scalix.com
15.145.204.249)
by scalixopp.pwd.scalix.com (Scalix SMTP Relay B.07.00.00)
via ESMTP; Thu, 19 Apr 2001 09:27:28 +0100 (BST)
Received: from scalix2.pwd.scalix.com (root@localhost)
by scalix2.pwd.scalix.com (8.8.6 (PHNE_17190)/8.8.6) with ESMTP id
JAA03131
for <fred@scalix1>; Thu, 19 Apr 2001 09:27:26 +0100 (BST)
Received: from joyford3 (scalixpwdl86.pwd.scalix.com
15.145.205.60)
by scalix2.pwd.scalix.com (Scalix SMTP Relay B.07.00.00)
via ESMTP; Thu, 19 Apr 2001 09:27:27 +0100 (BST)
Date: Thu, 19 Apr 2001 09:27:26 +0100
From: "tom" <tom@scalix2.pwd.scalix.com>
Sender: "tom" <tom@scalix2.pwd.scalix.com>
To: fred@scalix1.pwd.scalix.com
Message-ID: <001401c0c8aa$90ca33a0$62cc910f@pwd.scalix.com>
Subject: a message
X-MSMail-Priority: Normal
```

```
X-Priority: 3
X-Scalix-Suspicious-Host: 15.145.205.23
X-Mailer: Microsoft Outlook Express 5.50.4133.2400
X-MimeOLE: Produced By Microsoft MimeOLE V5.50.4133.2400
MIME-Version: 1.0
Content-Type: multipart/alternative;
boundary="----=_NextPart_000_0011_01C0C8B2.F2455DA0"
...
```

A header is inserted to show that this message was sent (or relayed) to Fred by scalixopp (15.145.205.23), one of the suspect hosts.

If you try sending a message directly from scalix2 to scalix1:

```
$telnet scalix1 25
Trying...
Connected to scalix1.pwd.scalix.com.
Escape character is ’^]’.
550 Denied
Connection closed by foreign host.
```

The connection is refused immediately. Note the rejection logged in the smtpd.log file on scalix1.

```
Rejected connection from 15.145.204.249
```

where 15.145.204.249 is the IP address of scalix2.

Blocking Mail From Specific Senders

The ORIGINATOR event describes an attempt to send a message from a user whose e-mail address matches a pattern. We can use this event to block mail coming from known spammers.

Here are some example lines:

```
ORIGINATOR Log_Reject spam@advert.com

ORIGINATOR Discard spam@blast.net

ORIGINATOR Defer spam*@*

ORIGINATOR Accept ALL
```

The first line rejects any message from spam@advert.com and logs the rejection in smtpd.log. The sending host will get a SMTP 500 response. The second line accepts messages form spam@blast.net but discards immediately discards them.

The third line defers the delivery of any messages from addresses matching spam*@*. The sending hosts receives an SMTP 400 response and the messages is stored on the sending host. The submission is attempted until the sending host rejects the messages.

If you set the header action, the inserted header takes the following form:

```
X-Scalix-Suspicious-Originator: email address
```

To flag any messages as suspicious from users with the scalix2.pwd.scalix.com domain in their address, add the following line to smtpd.cfg on scalix1:

```
ORIGINATOR Header *@scalix2.pwd.scalix.com
```

When you use an Internet client to send a message from tom on scalix2 to Fred on scalix1, the message has the extra header underlined below:

```
Return-Path: <tom@scalix2.pwd.scalix.com>
Received: from scalixopp.pwd.scalix.com (scalixopp.pwd.scalix.com
15.145.205.23)
by scalix1.pwd.scalix.com (Scalix SMTP Relay B.07.00.00)
via ESMTP; Thu, 19 Apr 2001 14:30:16 +0100 (BST)
Received: from scalixopp.pwd.scalix.com (root@localhost)
by scalixopp.pwd.scalix.com (8.9.3 (PHNE_18546)/8.9.3 SMKit7.01
Scalix) with
ESMTP id OAA18566
for <tom@scalix1.pwd.scalix.com>; Thu, 19 Apr 2001 14:30:15 +0100
(BST)
Received: from scalix2.pwd.scalix.com (scalix2.pwd.scalix.com
15.145.204.249)
by scalixopp.pwd.scalix.com (Scalix SMTP Relay B.07.00.00)
via ESMTP; Thu, 19 Apr 2001 14:30:15 +0100 (BST)
Received: from scalix2.pwd.scalix.com (root@localhost)
by scalix2.pwd.scalix.com (8.8.6 (PHNE_17190)/8.8.6) with ESMTP id
OAA03265
for <tom@scalix1>; Thu, 19 Apr 2001 14:30:14 +0100 (BST)
Received: from joyford3 (scalixpwdl86.pwd.scalix.com
15.145.205.60)
by scalix2.pwd.scalix.com (Scalix SMTP Relay B.07.00.00)
via ESMTP; Thu, 19 Apr 2001 14:30:15 +0100 (BST)
Date: Thu, 19 Apr 2001 14:30:14 +0100
From: "tom" <tom@scalix2.pwd.scalix.com>
Sender: "tom" <tom@scalix2.pwd.scalix.com>
To: tom@scalix1.pwd.scalix.com
Message-ID: <001401c0c8d4$ddc0b5b0$62cc910f@pwd.scalix.com>
Subject: see the headers
X-MSMail-Priority: Normal
X-Priority: 3
X-Scalix-Suspicious-Originator: <tom@scalix2.pwd.scalix.com> at
X-Mailer: Microsoft Outlook Express 5.50.4133.2400
X-MimeOLE: Produced By Microsoft MimeOLE V5.50.4133.2400
MIME-Version: 1.0
Content-Type: multipart/alternative;
boundary="----=_NextPart_000_0011_01C0C8DD.3F55A940"
```

Blocking Mail For Specific Recipients

Use the RECIPIENT event to manage messages sent to addresses that do not exist. This option is useful if an individual is no longer part of an organization and you want to stop mail from being delivered to that account without actually removing (deleting) the account.

You can configure RECIPIENT events as follows:

```
RECIPIENT Log_Reject Unknown.User@pwd.scalix.com
```

```
RECIPIENT Reject User.HasLeftTheCompany@pwd.scalix.com
RECIPIENT Accept ALL
```

The first line rejects incoming messages for the user Unknown.User@pwd.scalix.com and logs the rejection in smtpd.log.

The second line rejects mail for User.HasLeftTheCompany@pwd.scalix.com.

Using a DNS Block List

If needed, you can create a DNS block list or "blacklist" (DNSBL) in which you identify a list of IP addresses to be avoided. This can be useful as a means to block known spammers.

To create a DNS Block List:

1. Go to the file ~/sys/smtpd.cfg
2. Add the following lines.

   ```
   # Reject and log submission from addresses listed in bl.spamcop.net:

   SUBMIT log_reject DNSBL,bl.spamcop.net,ALL
   ```

3. Restart the smtpd service.

   ```
   omoff -d0 -w smtpd

   omon smtpd
   ```

Authentication

This chapter explains how Scalix's native authentication system works, and how to integrate with external authentication systems such as LDAP, Kerberos or Windows NT Domain, if desired.

If you plan to use the authentication system that is native to the product, you can skip this chapter.

Contents:

This chapter includes the following information:

Authentication Overview

Many different components of Scalix require authentication. Because there are so many different parts and protocols in the system, it requires more than one authentication mechanism.

The Scalix components that require some form of authentication include:

- All UAL remote clients (including Scalix Connect for Outlook and IMAP)
- POP3
- IMAP
- SMTP relay for SMTP authentication
- LDAP server
- Scalix admin commands
- Special Scalix diagnostic commands

Note	The admin tools use authentication in a very specific and different way.

Likewise, there are several different ways you can do authentication on a Scalix system: You can rely entirely on Scalix's native authentication setup, or work with your own such as an existing LDAP server, the Kerberos method, or the Microsoft Windows NT domain, which function alongside the native authentication method.

For maximum flexibility, all authentication configurations are based on the Linux standard Pluggable Authentication Modules (PAM).

The way authentication works in Scalix is:

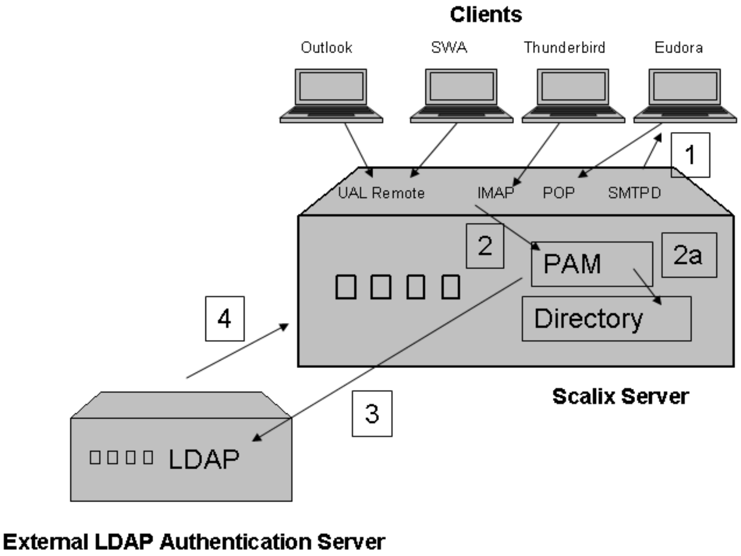
1 The clients communicate via their own access language (UAL, IMAP, or POP3) to request authentication from the Scalix server, where they encounter the PAM modules.

2 On the server, the PAM modules get the user's information from the directory.

- If you are using the native authentication system, the PAM modules verify the username and password in the directory, then send confirmation back to the client via UAL, IMAP POP or SMTPD.

3 If you are using external authentication such as LDAP or Windows NT, it grabs the authentication ID from the directory and sends that to the LDAP server for authentication.

4 The external authentication server sends a success/failure message back to the PAM modules, which pass the message back to the clients via UAL, IMAP, POP or SMTPD.



Details

Various components of the Scalix system incorporate authentication in very different ways; and not every implementation is fully logical. Fortunately there is some level of convergence; most email clients ultimately end up using a UAL remote connection type as their point of entry into the server. This is true for Scalix Connect for Outlook, SWA, IMAP clients such as Mozilla Thunderbird and even the command line mail tools omsend, omlogon, etc.

The exception to this is the POP and SMTP server, which talks to the message store directly for performance reasons. Therefore, the POP service needs its own PAM configuration file (which is quite easy to miss when setting up external authentication.)

The Scalix admin commands also use the PAM infrastructure; in this case it is used to check if a user executing an admin command actually has (1) admin rights or (2) his or her effective user ID equals 0, identifying the user as a root-level user.

Two very specific commands, omqdump and omcontain require another, extra layer of protection; they use the so-called *diag* authentication, which is based on a secret password only known by those properly initiated through Scalix training.

An Overview of PAM

Now a standard library in LInux, Pluggable Authentication Modules (PAM) connects applications that require authentication with shared library modules that interface with external authentication mechanisms. This includes such authentication mechanisms as Unix-passwd, SMB/NT Domains, LDAP, Kerberos, RADIUS and more. PAM is highly configurable.

Scalix is installed with a collection of PAM modules, but you can configure them for freeware or commercial modules as long as they follow the PAM spec.

What is PAM?

PAM is a standard API, with a full set of libraries incorporated in modules. These modules are used in the Linux and Unix world to make applications independent of authentication mechanisms. Various applications such as the Unix login, network services such as http, ftp or ssh can be set up to use the PAM infrastructure.

On the authentication module side, PAM modules are available to help a system authenticate against a variety of systems such as the Linux passwd/shadow files, LDAP servers, Kerberos Ticket systems or legacy Windows NT domains. Open network authentication mechanisms such as RADIUS or TACACS are also supported by PAM modules, as are a variety of biometric devices such as fingerprint or iris scanners.

PAM Configuration File Syntax

The PAM configuration files are rule-based ASCII files that follow a common syntax. When modifying them to set up external systems, remember this syntax:

```
module-type  control-flag  module-path  args
```

Where:

- **module-type** refers to one of the four types (the three used by Scalix)
- **module-path** is a relative or absolute path to the PAM shared library module with or without its `.so` suffix.

  In Scalix-PAM, pathnames are interpreted relative to */opt/scalix/lib/security* while in standard Linux PAM they are relative to */lib/security.*

- **args** is a list of arguments passed to the PAM module. Every PAM module supports at least a few generic arguments.

The following example illustrates a simple PAM configuration file:

```
# ~/sys/pam.d/ual.remote example
auth required om_auth nullok
account required om_auth
session required om_auth
password required om_auth nullok


# ~/sys/pam.d/pop3 example
auth required /lib/security/pam_smb debug nolocal
account required om_auth
session required om_auth
password required /lib/security/pam_smb debug nolocal
```

PAM Module Stacking

If one application has multiple PAM entries, as would be the case with an external authentication system, they execute in the order in which they appear in the configuration file. So if one method of authentication fails, it fails over to the other.

For example: If you are using an existing LDAP authentication system that fails, your users can provide their Scalix username and password to get in. In this scenario, the ual.remote file has these entries:

```
auth sufficient om_ldap

auth sufficient om_auth

auth required   pam_deny

account required om_auth

password required om_auth

session required om_auth
```

Where “sufficient” means that the external LDAP authentication is not “required,” so can fail over to the internal Scalix authentication.

The *control-flag* parameter defines how every single stacked module contributes to the overall success or failure of the operation. There are four different values for this field:

- Required: This module is required for authentication to succeed
- Optional: The result of this module is used only if there are no other modules returning results.
- Sufficient: If the module succeeds, stop processing modules and return sucess.
- Requisite: If the module fails, stop processing modules and return failure

So, in the example code above, the system first tries to authenticate through "om_auth". If that succeeds, everything is fine and the user is signed on. If om_auth fails, it next tries om_ldap. If that one succeeds, the user is signed on. If it, too, fails, the system tries pam_deny, which always fails.

The practical upshot is that the user has two chances to authnticate: Through their local password or their LDAP password.

Examples of Module Stacking

The following examples apply to module stacking for the auth module type only. Note that these configurations can be adapted for the other module types as well.

Example 1: Try to authenticate using ldap first, if user is unknown, try Scalix next

```
auth sufficient om_ldap user_unknown=ignore
auth sufficient om_auth use_first_pass
auth required pam_deny
```

Example 2: Try to authenticate using a Kerberos password, and if the user is “admin” and Kerberos fails, try a second time, this time authentication against the Scalix internal password.

```
auth sufficient om_krb5
auth required om_admin
auth required om_auth use_first_pass nullok
```

Note that the user MUST exist in Kerberos, even if the password does not match.

PAM Config Files for Scalix Applications

If you are using an external authentication system, the following config files must be modified so that Scalix knows it should authenticate against an external source. They are located in ~/sys/pam.d

- ual.remote: Allows Outlook and SWA users to authenticate against an external authentication server.
- omslapdeng: Allows SWA personal contacts to be searched.
- smtpd.auth: Allows users coming in through SMTPD to authenticate against an external authentication server.
- POP3: Allows POP3 users to authenticate against an external authentication server.

The file names are arbitrarily chosen and hard-coded into the respective programs; the LDAP server and SMTP Relay actually have two files each, one for admin access to the server process (i.e. for starting/stopping it), the other for actual network access through the respective protocol. The config file names listed above refer to the network functionality as PAM files for commands rarely change.

Default PAM Configuration

The default configuration of all Scalix components uses the om_auth module, which includes full functionality, so it can be used for all four module types. If you do not intend to use external authentication, leave these as is.

The default PAM configuration for ual.remote is as follows:

```
auth     required om_auth nullok
account  required om_auth
password required om_auth nullok
```

The nullok argument will allow empty passwords to be used.

The module will get a canonicalized username in positional format provided by the Scalix-PAM library.

Authentication Against Scalix's Internal Userlist Directory

The om_auth PAM module provides authentication functionality against the Scalix internal USERLIST directory, in this manner:

- The authentication function of the module asks the user for a password, encrypts it and checks it against the UL-PWD attribute in USERLIST.
- The account function checks if the password is not expired, the account is not locked and no other restrictions exist.
- The password function actually enables the user to change the password in USERLIST. It updates the UL-PWD and UL-SASL-PWD attributes. (The latter is needed by SMTP, IMAP and LDAP authentication when using the more secure SASL authentication.) For SMTP, this is the only supported authentication method. Plain text authentication is not allowed.
- The UAL layer canonicalizes the user name to *Positional Format*. So the username the PAM module works on contains at least the X.400 Personal Name, Mail-node and Common Name.

Commonly-Used Generic Options

The following generic options are the most commonly used in PAM configuration. They are located in the *ual.remote* file and can be commented out if desired.

- Debug: Write debugging information to syslog. This option makes most PAM modules log more information that might be useful for troubleshooting. In Linux PAM environments, logging goes to syslog, whereas in Scalix environments, it is redirected to the Scalix logging mechanism.
- Use_first_pass: Do not prompt the user for a password. Instead, use the password retrieved by a previous module in the stack. All modules handling authentication should actually implement the use_first_pass argument, preventing a secondary module on the stack prompting for a password again and instead taking it from the existing PAM environment created for and modified by the module(s) stacked on top of it.

PAM Modules that Scalix Supplies

While any PAM module can be used, Scalix supplies a few original PAM modules. These differ from Linux PAM modules in their logging behavior (syslog vs. Scalix log), their configuration file location (/etc vs. ~/logs) and the type of username they can accept.

If you want to use the native authentication sytsem, leave these modules as they are. If you plan to use external authentication, they will be modified. For more on modifying the Scalix PAM modules for use with an external authentication system such as LDAP, see "PAM Integration with LDAP" on page 59.

The following list categorizes the Scalix-supplied PAM modules:

Full authentication modules

- om_auth
- om_ldap, om_krb5

Generic PAM helper modules

- pam_permit, pam_deny
- pam_if, pam_listfile

Admin Scalix PAM modules

- om_admin
- om_unix2om
- om_diag

Special Integration PAM module

- om_om2authid

Unsupported Modules

- pam_radius_auth, pam_smb, pam_unix

This is a full descriptive list of the same set of Scalix-supplied modules:

- om_auth: The main Scalix module. Authenticating against the Scalix USERLIST directory.
- om_ldap: A Scalix version of the standard Linux pam_ldap module.
- om_krb5: A Scalix version of the standard module for Kerberos 5.
- pam_permit: Always allow. (Useful if stacking is required.)
- pam_deny: Always deny. (Useful if stacking is required.)
- pam_if: Conditionally stack.
- pam_listfile: Allows if user is part of a textfile based simple list.
- om_admin: Allows if Scalix user associated with callers Unix user ID is either root or a Scalix admin user.
- om_unix2om: When this is stacked with om_admin, it converts a Unix UID into a Scalix User ID.
- om_diag: Support special authentication modes for omqdump/omcontain.
- om_om2authid: Convert a Scalix User ID into an authentication identifier. (Useful if stacking is required.)

PAM Modules that the Operating System Supplies

You can use any OS-supplied PAM module with Scalix. When customizing the Scalix PAM configuration file, be sure to enter the absolute pathname to the correct module.

The module might need additional configuration; refer to the module‘s documentation/ manpage for details.

The Scalix username handed over by the Scalix PAM library must first be converted into the Authentication Identifier. This is done using the special `om_om2authid` PAM module in the stack.

When using an OS-supplied PAM module, you should be aware of the following requirements:

- The module name should be specified as a fully-qualified path in the PAM config files.
- The module will almost certainly require configuration files outside the Scalix directories; therefore it might not be possible to configure multiple instances of the module differently in a multi-instance Scalix environment.
- Scalix PAM modules use a username in positional format; this cannot be interpreted by a OS-supplied PAM module; the username must first be converted into the user's authid, which is used to *link* to the external authentication provider. The om_om2authid module is stacked before the actual PAM module and does the conversion in the PAM environment.

PAM Integration with LDAP

When integrating with an LDAP server, the PAM configuration for the ual.remote file and other services typcially looks like this:

```
auth  sufficient  om_ldap  user_unknown=ignore

auth  sufficient  om_auth  use_first_pass nullok

auth  required    pam_deny
```

This allows a user to be successfully authenticated through LDAP but still accepts accounts defined locally to Scalix only. For example: admin accounts.

Configuring Scalix for LDAP Authentication

If you choose not to use the PAM authenticaton that is native to Scalix, the system also supports authentication against a non-Scalix directory such as OpenLDAP, which is an Open Source package available for RedHat and SuSE Linux. It also provides failover capabilities so that if one directory is not available, a secondary directory is automatically used.

This section describes how to allow users to authenticate with their LDAP passwords against an LDAP-compliant directory when accessing their Scalix mailboxes.

Configuring LDAP Authentication for Clients

To set up communication between the various clients, the PAM modules and LDAP, you have to create or configure several files on the Scalix server.

To configure for the different clients:

1. Go to the directory *~/sys/pam.d.*
2. Inside that directory, find the following files:
   - For UAL clients such as Outlook and SWA, modify the file *ual.remote*.
   - For Mozilla Mail, which uses AuthSMTP, create or modify the file *smtpd.auth*.

- For POP3 clients such as Eudora or Mozilla Mail, create or modify the file *pop3*.
- For SWA personal contacts, create or modify the file *omslapdeng*.

3 *A*dd the following lines to each file for which you have active clients.

```
auth sufficient om_ldap

auth sufficient om_auth

auth required   pam_deny

account required om_auth

password required om_auth

session required om_auth
```

Note

Of these six lines, the first three, auth, test the username and password that the user supplies. The fourth line, account, decide whether or not you have access to the system. The fifth, password, change your password if needed. And the six, session, currently are not in use but must be present for proper system function.

Note

The second line "auth sufficient om_auth" provides a secondary opportunity to access the mailbox if invalid credentials are passed to the LDAP source. The second line compares the credentials to the Scalix directory. Remove this line if you do not wish to provide this.

Note

The Scalix Admin Console (SAC) also uses the file *omslapdeng*. So if you plan to use SAC, you must keep the second line, "auth sufficient om_auth" in order for SAC to authenticate the sxadmin and sxqueryadmin users.

Final Configuration

The final configuration for LDAP is to the general LDAP configuration file, *om_ldap.conf.*

While the PAM config file connects the LDAP PAM module to the Scalix system, the module itself must be configured to know how to connect to the LDAP server. This is done through the *~/sys/om_ldap.conf* config file.

To configure the general LDAP file:

1 Go to the directory *~/sys*.

2 Create (or modify) the file *om_ldap.conf*.

3 Add the following lines.

```
host=ldaphost.acme.com

search=subtree

base=dc=acme,dc=com

filter=uid=%s
```

Parameters in this file are defined as `name=value`, with a full list of parameters including the following:

- **host:** Specifies hostname and optionally port number for the LDAP server to use.
- **base**: Specifies the search base for the initial search operation as a DN; only set for search values of one or subtree.
- **search:** Can be one of none, one, subtree to specify the depth of search; should be set to none if using a one-level tree to avoid the search operation altogether.
- **filter:** LDAP filter string used for the search in one and subtree modes.
- **dn**: Set to the dn of the user with search=none.
- **tls**: Set to on or off, to enable the LDAP module to negotiate use of TLS; the LDAP server must also be configured to support this.

Verifying LDAP Authentication

To determine if the functionality is working properly, choose a user who has a different LDAP password than their Scalix password. Use the following sequence of steps.

To verify LDAP authentication:

1. Log in to Outlook.
2. In the Scalix logon screen, enter the user's LDAP password.

   The user should be able to log in successfully.
3. Log in again to Outlook.
4. In the Scalix logon screen, enter the user's Scalix password.

   The user should be able to log in successfully.
5. Log in to Outlook.
6. In the Scalix logon screen enter an incorrect password.

   The login should fail.

Choosing Between the Native and Operating System LDAP Modules

Before integrating with an LDAP authentication server, you must decide whether to use the native PAM LDAP modules (om_ldap) or the regular Linux PAM LDAP modules (pam_ldap).

The Scalix-supplied module is more efficiently integrated into the Scalix environment by means of placement of log and config files. But it has several limitations:

- If a search operation is required, the LDAP server must allow for anonymous access; pre-binding is not supported.
- The module does not support LDAP password changes.
- When using TLS, the module uses part of the OS configuration for open LDAP clients. This could interfere with a configuration of the operating system to use LDAP for authentication purposes.

By contrast, most Linux vendors deliver much newer and more advanced versions of PAM modules for LDAP authentication.

The advantages of the newer, more advanced versions often outweigh any additional problems. So consider deprecating the om_ldap and relying solely on pam_ldap.

Configuring Scalix for Windows NT Authentication

If needed, a third method of authenticating is the existing Windows NT 4 Domain authentication system.

To integrate with the Windows NT authentication system:

1. Put the hostname of your primary and secondary Windows domain controller in your DNS and/or */etc/hosts* file. As these are NETBIOS names, typing the names in ALL CAPS is important.
2. Create the */etc/pam_smb.conf* config file with a text editor. The file must have exactly three lines, as shown here with placeholder texts:

```
NTDOMAINNAME

NETBIOS_NAME_OF_PDC

NETBIOS_NAME_OF_BDC
```

Alert — If you have only one DC, put its name in two times.

Using the pam_smb_auth Module

The pam_smb_auth PAM module that comes with all supported Linux distributions is a perfect example of the benefits of and issues with OS-supplied PAM modules.

On one hand, the module makes it possible to authenticate against a Windows NT domain controller or other SMB compliant system; this is currently not possible with Scalix-supplied PAM modules.

On the other hand, this specific module provides very limited debugging information; config file syntax must be followed rigidly; capitalization and empty lines will matter. The quality of the code in the module is not controlled by Scalix's strict quality management and can only be supported in a limited way.

Set up your ual.remote Scalix PAM configuration file as follows:

```
auth required om_om2authid

auth sufficient /lib/security/pam_smb_auth debug nolocal

auth sufficient om_auth use_first_pass

auth required pam_deny
```

Configuring Scalix for Kerberos Authentication

A fourth method of authentication that works with Scalix is Kerberos, which can take several forms. The different approaches to Kerberos authentication and how to implement each one are explained below.

About Kerberos Authentication

Kerberos is a network authentication protocol designed to provide strong authentication for client/server applications by using secret-key cryptography. Kerberos uses authentication tickets to authenticate users and/or services.

A Kerberos client can perform secure communications with a Kerberos service if both the client and the service authenticate against a Kerberos Distribution Center (the KDC - the Kerberos Server) and obtain a Ticket Granting Ticket (TGT). The client then requests a service ticket for a specific service.

Therefore, a triangular relationship exists between the client and the KDC, the service and the KDC, and between the client and the service. A Kerberos principal is either a client identity or a service identity operating in the Kerberos realm.

In the Scalix environment, you can use secure Kerberos communication with the following Scalix Services:

- Remote Execution Service
- Scalix Management Console Service
- Scalix UAL Service
- Scalix IMAP Service

You can also configure single sign-on authentication with a KDC on the master domain controller that uses Microsoft Active Directory. This allows Scalix users to automatically authenticate with the Scalix server when they log in to their Windows domain. See “Single Sign-on Kerberos Authentication” on page 63 for more information.

Single Sign-on Kerberos Authentication

Single sign-on authentication allows MS Outlook users to access their e-mail using the Kerberos security protocol. This authentication mechanism allows users to log in to their local domain in a Microsoft Active Directory® environment and access their e-mail without any further authentication.

The Active Directory service is a core component of the Windows operating system. It provides a directory service supporting LDAP, and a Kerberos Key Distribution Center (KDC) to authenticate users. It allows organizations to share and manage information about network resources and users and provides a single sign-on environment that integrates with the standard Windows desktop login. In addition, it acts as a single point of management for Windows-based user accounts, clients, servers and applications.

Kerberos is a network authentication protocol designed to provide strong authentication for client/server applications by using secret-key cryptography. Kerberos uses authentication tickets to authenticate users and/or services.

A Kerberos client can perform secure communications with a Kerberos service if both the client and the service authenticate against a Kerberos Distribution Center (the KDC - the

Kerberos Server) and obtain a Ticket Granting Ticket (TGT). The client then requests a service ticket for a specific service.

Therefore, a triangular relationship exists between the client and the KDC, the service and the KDC, and between the client and the service. A Kerberos principal is either a client identity or a service identity operating in the Kerberos realm.

When a user logs in to the domain, a request is made for a ticket, and once authenticated, the user can use that ticket for as long as it remains valid. Server-side tickets are stored in the **keytab** file. Client-side tickets are stored in a temporary file.

When you launch MS Outlook, the Scalix Connect connector uses the user's Kerberos credentials to access the Server.

To implement the single sign-on environment, you must have:

- Active Directory
- The latest version of Scalix Connect for Microsoft Outlook
- The latest version of Scalix Server
- The ktpass utility from the Microsoft Developer Network support web site

Creating Keytab Files

The ktpass utility creates the keytab files used by Linux Kerberos-based systems to define Key Distribution Center (KDC) hosts and user/service mappings.

ktpass is available from the Windows 2000 resource kit and the Windows Server 2003 installation CD under \Support\Tools\Support.cab.

For more information on this tool, go to the following URL:

http://www.microsoft.com/windows2000/techinfo/planning/security/kerbsteps.asp

You must install the ktpass utility on the Windows domain controller server.

Configuring Single Sign-on

Single sign-on configuration requires you to make configuration changes to DNS and create an Active Directory user. This user, for single sign-on purposes, is actually the Scalix Service. You must create an Active Directory "user" for the UAL Scalix service. When this is complete, you then convert this user/service into a Kerberos Service Principal.

Note	You can also create an Active Directory user for the Scalix IMAP service if single sign-on users in your network use the Evolution e-mail client.

To configure single sign-on authentication.

1. On the domain controller, go to **Start > Programs > Administrative tools > DNS**.
2. Make sure you have created Forward Lookup Zones for your domains and created host records for all Scalix servers in the appropriate zone.
3. Under **Forward Lookup Zones,** select a Scalix server single sign-on domain and go to **Action > New Alias**.
4. In the **Alias name** field, enter `scalix-default-mail`.

5 In the **Fully qualified name for target host** field, enter the fully-qualified name of the Scalix server with which you are using single sign-on (for example, `scalixserver.acme.net`).

6 Click **OK**.

7 Select **Reverse Lookup Zones** and make sure you have created zones for your domain subnets.

8 In the subnet in which the single sign-on Scalix server resides, select **Action > New Pointer.**

9 Enter the last two or three digits of the Scalix Server IP address and fully-qualified hostname of the Scalix Server (for example, `scalixserver.acme.net`).

10 Click **OK**.

11 Close the DNS window.

12 Go to **Start > Programs > Administrative Tools > Active Directory Users and Computers**.

13 If it does not already exist, right-click the root domain controller and select **New > Organizational Unit** and name the new unit `Scalix Services`.

This creates a separate organizational unit to contain Scalix server data.

14 Select the new Scalix Services organizational unit and select **Action > New > User**.

15 In the **First Name** field Enter `scalix-ual`.

You can also enter the name of the single sign-on Scalix server in the **Last Name** field. This allows you to identify the keytabs you generate for multiple Scalix servers.

16 Do not modify the selection (domain) in the pull-down menu.

17 In the **User logon name** field, enter `scalix-ual` .

18 Click **Next**.

The Password window displays.

19 Enter and confirm a password for the user. Make sure that the password you enter is sufficiently complex and that;

- The **User must change password at next logon** field is not selected
- The **User cannot change password** field is not selected
- the **Password never expires** field is selected

20 Click **Next**.

If Microsoft Exchange is installed on the server, the Exchange Mailbox window displays.

21 Clear the **Create an Exchange Mailbox** field.

22 Click **Next**

23 Click **Finish**.

This completes the creation of an Active Directory user that represents the Scalix UAL Service for the Scalix Server.

24. On the domain controller, open a DOS window and change the directory (cd) to the directory that contains ktpass (typically, `c:\Program Files\Support Tools`). See “Creating Keytab Files” on page 64 for more information.

25. To change the Scalix Service account to a Kerberos Service account and generate a keytab, enter:

```
ktpass –princ scalix-ual/scalixservername.domain@REALM –mapuser
<domain>\scalix-ual -pass password –out path\filename -kvno 3
```

For example:

```
ktpass -princ scalix-ual/scalixserver.acme.net@ACME.NET -mapuser
scalix-ual -pass <password> -out scalix-ual.keytab -kvno 3
```

> **Note** The -kvno option prevents potential key version mismatches that cause SSO to fail. Setting this value to 3 ensures that keytab version is the same for existing and future users in Active Directory.

26. If you used the Last name field and entered `scalixserver1`, enter:

```
ktpass -princ scalix-ual/scalixserver.acme.net@ACME.NET -mapuser
scalix-ual-scalixserver1 -pass password -out scalix-ual-
scalixserver1.keytab
```

You should see the following information indicating that the keytab was successfully created:

```
Successfully mapped scalix-ual/scalixserver.acme.net to scalix-
ual.
Key created.
Output keytab to scalix-ual.keytab:
Keytab version: 0x502
keysize 68 scalix-ual/scalixserver.acme.net@ACME.NET ptype 1
(KRB5_NT_PRINCIPAL)
vno 3 etype 0x1 (DES-CBC-CRC) keylength 8 (0xe6fb762ad01f8a9b)
Account has been set for DES-only encryption.
```

27. Securely copy the keytab to the home directory of the single sign-on Scalix server. You can use a floppy disk or the `scp` command to transfer the keytab.

28. On the Scalix server, log in using your Linux account and then change to `root` user.

29. Merge the keytab you created with the Kerberos system keytab file. Enter:

```
ommergekeys /path/filename.keytab
```

30. Modify the `/etc/`**krb5.conf** file. Enter:

```
omkrbconf -r REALM -s servername.domain -d domain
```

Where

- `-r` specifies the realm that the Kerberos database controls. For example, if your domain name is acme.com, your realm is ACME or ACME.NET.
- `-s` specifies the fully qualified host name of the Kerberos KDC machine. For Single Sign-on, the KDC is the Domain Controller with Active Directory installed.

- (optional) `-d` specifies the domain name in which the Kerberos Realm operates. If you do not specify a value, the domain name is determined from the current domain.

31 In order for single sign-on to operate, the authentication ID for a Scalix server mailbox must match the domain identity (the ID in Active Directory) for the user. For example, if `jsmith@acme.net` is the User Logon ID for a user in Active Directory, enter the following on the Scalix Server:

```
ommodu –o jsmith -–authid jsmith@ACME.NET
```

The `REALM` information MUST be uppercase.

32 To view the `authid` value (`-o`) for a user, enter:

```
omshowu “Joe Smith/mailnode”
```

This user can now use single sign-authentication. After the user logs in to the Windows domain, the user no longer must enter username or password information during MS Outlook profile creation or login.

If Active Directory is unavailable at any point after setting up single sign-on, the Scalix server prompts users for their regular domain password for authentication.

Non-SSO Kerberos Authentication

Non-SSO Kerberos authentication differs from SSO authentication in that the user is required to enter a password to log in to MS Outlook, Scalix Web Access, or IMAP clients. However, the password the user enters is their Kerberos password instead of their Scalix password.

To configure non-SSO Kerberos authentication:

1 Make sure that Kerberos Server, Workstation, and Libraries are installed on a Scalix Server in your network. To verify that the required Kerberos rpms are installed on the system, enter:

```
rpm -qa | grep krb
```

You can obtain any missing Kerberos rpms from the Linux operating system installation CDs. See http://web.mit.edu/kerberos/www/ (Red Hat and Fedora) or http://www.pdc.kth.se/heimdal/ (SuSE) for more information.

2 Initialize the KDC. Enter:

```
omkrbinstall -r realm -s servername.domain -a username -p password
```

Where

- `-r` specifies the realm that the KDC manages.
- `-s` specifies the name of the Scalix Server that hosts the KDC.
- `-a` specifies the principal fully qualified hostname of the KDC administrator.
- `-p` specifies the KDC administrator password.

3 The following prompt displays:

```
Checking MIT Kerberos installation...done. Initializing database '/
var//krb5kdc/principal' for realm 'REALM', master key name 'K/
M@REALM' You will be prompted for the database Master Password. It
is important that you NOT FORGET this password.
```

```
Enter KDC database master key:
```

4 Enter a password for the KDC database.

```
Reenter KDC database master key to verify:
```

5 Reenter the password for verification.

The following information displays:

```
Success! Kerberos database created, configured and started.
```

6 Create Scalix Service principal keytabs. Enter:

```
omaddprincs -s all -h server.domain -o filename.keytab
```

- Specifying all for the -s option creates one keytab file for all Scalix Services (Remote Execution Service, UAL, IMAP, and the Scalix Management Console Service). However, you can also create individual keytabs for specific services. See the omaddprincs man page for more information.
- -h specifies the fully qualified domain hostname of the Scalix Server on which the Scalix Services are installed.
- -o specifies the keytab file name.

The following information displays if you use -s all to create a keytab:

Creating new Scalix principals in Kerberos database and keytab filename.keytab:

```
scalix-ual/server.domain@REALM
imap/server.domain@REALM
ubermanager/server.domain@REALM
res/server.domain@REALM
```

7 If necessary, manually copy the keytab file(s) to the Scalix Servers on which you want to use Kerberos authentication. See the *Scalix Installation Guide* for information about Scalix Management Console-specific keytab deployment.

8 Modify the file ~/sys/pam.d/**ual.remote** so that it appears as follows:

Note: Bold text indicates the lines that need to be modified.

```
# Standard Scalix Authentication
#
# Comment this out if you want to use one of the alternative
authentication
# schemes below.
# auth  required om_auth nullok

#
# Kerberos authentication 1
#
# With this scheme we attempt local authentication first and, if
that
# fails, we try Kerberos authentication. Note that if we do it the
other
# way around we run the risk of the KDC locking a principal account
for
# users that are known to both Kerberos and Scalix. See om_krb5(8)
for more
```

```
# information.
#
# auth  sufficient om_auth nullok
auth  sufficient om_krb5 use_first_pass
auth  required pam_deny
# Kerberos authentication 2
```

When you modify and save the ual.remote file, client sessions are initiated using UAL_INIT/UAL_SIGNON, and include the principal name and password. After the user is found (verified) in the USERLIST Directory, the authid value from their Directory entry is used to authenticate them through PAM (Pluggable Authentication Module). The om_krb5 module (with the authid value and password) is used to contact the KDC through the Kerberos client libraries.

9 For non-SSO Kerberos POP access, modify the file ~/sys/pam.d/pop3 so that it appears as follows:

Bold text indicates the lines that need to be modified.

```
#auth     required om_auth
account  required om_auth
password required om_auth
auth  sufficient om_krb5 use_first_pass
auth  required pam_deny
```

10 Scalix users will now authenticate against the KDC using their Kerberos password. If users experience problems while logging in to Scalix, make sure they are in the KDC.

```
kinit username
```

11 If the user is not in the system, enter:

```
kadmin.local
```

```
addprinc -pwd password username
```

12 This adds a user principle. To verify that the user was successfully added, enter:

```
listprincs
```

You should see a user principal for the user you created.

Make sure the user’s authid value is set to username@DOMAIN.NET.

Using the Domain Password

If you want to use Kerberos authentication and have users use their Windows (Active Directory) domain password when logging into Scalix, complete all the steps in <Xref_Color>“Single Sign-on Kerberos Authentication”, and then edit the ~/sys/pam.d/ual.remote file as described in Step <Xref_Color>8 of <Xref_Color>“Non-SSO Kerberos Authentication”.

Some Kerberos Behaviors to Note

If your system performs Kerberos-based SSO with Scalix Connect for Outlook, you should know that the behavior of username mapping has changed since the 10.0.0 release:

- For all implementations, the Authentication ID of the user must be modified to convert the existing username to all-lowercase letters. For example, if a users Authentication

ID has been "SmithJane@SCALIX.REALM", it must be converted to "smithjane@SCALIX.REALM".

- A new omldapsync option, detailed below, is provided to automate this mapping when synchronizing the username from Active Directory or other LDAP-based directories.
- For Active Directory based implementations, where the AD KDC treats the username as case-insensitive, this will result in some issues with capitalizations of usernames during Sign-on to be resolved.
- For MIT-Kerberos based Kerberos KDC implementations, where the MIT KDC treats the username as case-sensitive, Kerberos-based SSO will now only work with lowercase usernames on the Kerberos Server side. This is in line with most implementations of MIT Kerberos.

Also, if you make use of Active Directory as the master for omldapsync agreement type 11, and are using Kerberos for Single Sign-on, then the principal username (from AD) will need to be converted to lowercase@UPPERCASE in Scalix. This can be done by editing the following mapping line in the agreement config file:

Convert this line:

```
userPrincipalName|UL-AUTHID|*,1,256|!TOUPPER=@|
```

To this:

```
userPrincipalName|UL-AUTHID|*,1,256|!CUSTOM=TO_CANONICAL_PRINCIPAL
```

This should be completed when creating the new sync agreement, before initiating omldapsync. If you are upgrading from an older version of Scalix, the existing sync agreement must be edited as shown above. You must then run omldapsync with the -M option (as detailed in the omldapsync man page). This will force all the existing records in Scalix to be updated.

For More Information:

For more information on PAM and Scalix, see the following:

Technote (on Scalix Support site:)

url/Scalix_Pluggable_Authentication_Modules_(OM-PAM).html

Acrobat files (on Scalix Support site:)

Securing Scalix

This chapter covers the many different ways to secure a Scalix system, ranging from internal security measures to the use of an Apache Web server, stunnel or VPNs.

Contents:

Overview

As a program, Scalix is only as secure as its operating system. Anyone with root permissions has unlimited access. For this reason, you must institute careful network security to ensure the safety and integrity of the system and its data.

Scalix servers are typically kept behind a corporate firewall such as inside an Intranet. In these cases, using VPN technology for client access is a useful tool to secure remote access and no additional security provisions are needed on the Scalix side.

However, you may also wish to use an Apache proxy server or an stunnel server in a DMZ to secure access to the system.

Tip	To implement even higher security, implement the Kerberos authentication protocol as described in the chapter titled, "Authentication" on page 52.

Internal Security Precautions

As with any mission-critical application, there are certain internal security precautions you should take. These are listed below and some are explained in greater depth elsewhere in this manual.

Data Security

To prevent unauthorized access to data, make sure other users are not assigned to the group named *scalix* because it is a special Linux user group that owns Scalix data. All Scalix data is owned by a special Linux user named *scalix* in the group named *scalix*. This user and group are created when you install the software.

Individual user data is password protected. Users access their data by being registered with Scalix. All users must enter their passwords before accessing their Scalix data.

Administrator Capabilities

Only administrators or users with root permissions can add, delete, or modify users, or modify the Scalix system.

The omcheck command enables the administrator to verify that ownerships and permissions are set correctly for Scalix system files and directories.

Restricted User Access

You can control access to public folders using access control lists. In addition, individual users can contol access to their mail, calendar and contact folders using delegate permissions.

Message Security

There are instances when a Scalix administrator can read messages addressed to other people. For example, if a non-delivery report comes through the administrator’s email box, or when using some Scalix diagnostic tools, the administrator may see the content of an individual message. However, messages marked as "Personal", "Private", or "Company Confidential" cannot be read by even the administrator.

Monitoring Usage

The Audit Log, which records user activity, can identify unusual usage patterns. In addition, in case of break-ins or inappropriate activity, it can provide evidence of when individual users were on the system.

Virus Protection

Scalix integrates with several third-party anti-virus programs. The service router interfaces with virus-scanning applications from Trend Micro, McAfee and ClamAV. When you activate virus scanning, the service router scans all Scalix message attachments. Depending on how you configure virus scanning, the Scalix server can attempt to repair infected files, return infected messages to the sender, or discard the message. For more on virus protection, see “Virus Protection” on page 23.

Spam Protection

Scalix also allows you to configure anti-spam measures on the SMTP Relay to prevent abuse of the Scalix system by external entities. It integrates with SpamAssasin, among other anti-spam programs. For more on Spam protection, see “Spam Protection” on page 32.

Outlook Client Security

Outlook E-mail security parameters provide protection against software viruses that users might receive in their Inbox as an attachment file.

In addition, access to Scalix Connect from outside a network should only be done through a secure VPN unless stunnel is configured.

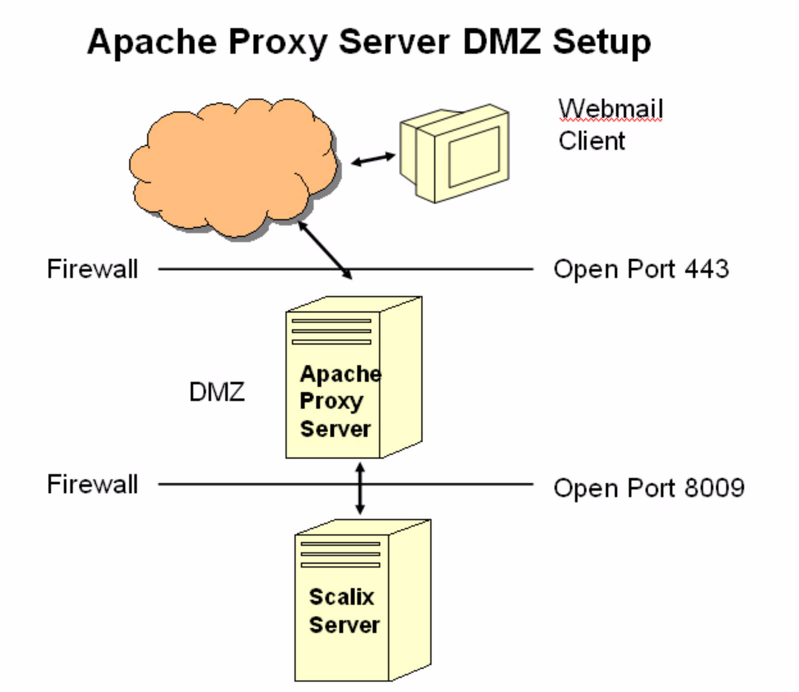
Using a VPN

The most efficient way to secure communication between many of the different clients that work with Scalix (Outlook, Outlook Express, Thunderbird, Apple Mail, etc) and the Scalix Server is through use of a VPN.

Because the setting up of a VPN is outside the scope of a Scalix system, instructions are not covered in this manual.

Using an Apache Web Server

If the majority of your users are on SWA, another good method to secure client access to an internal Scalix system is through an Apache Web Server in the DMZ, which activates HTTPS and serves as a proxy or gateway to the system. This setup would look like:



There are four distinct steps to setting up and integrating an Apache proxy server with Scalix:

- Install Scalix as documented in the Scalix Installation Guide.
- Install and configure the Apache Web server in the DMZ.
- Install the Scalix Tomcat Connector rpm on the Apache Web server to facilitate communications between Tomcat on the Scalix server and Apache in the DMZ.
- Copy the scalix-tomcat files from the Scalix server to the Apache server and replace the virtual host entry.

Because you already have installed Scalix at this point, and because Apache is bundled with any Linux server, these instructions begin with Apache configuration.

Configuring the Apache Web Server for use with Scalix

Because Scalix Web Access (SWA) and the Management Console (SAC) exchange data and credentials with the Scalix Server without encryption, Scalix Corporation recommends activating *Secure Socket Layer* (SSL) security.

IMPORTANT: Name-Based Virtual Hosts and SSL

It is not possible to run multiple SSL-enabled virtual hosts on a server with only one IP address. Users connecting to such a setup will receive a warning message stating that the certificate does not match the server name every time they visit the URL. A separate IP address or port is necessary for every SSL-enabled domain to achieve communication based on a valid SSL certificate. Despite the warning message, you still get the same level of encryption that you would have on any valid SSL site. This means that as long as the warning message is acceptable, communications between the Web server and client are still secure. The concept of uniquely knowing the server's identity, which is guaranteed by a valid SSL certificate, is forfeited.

Setting up SSL for SUSE Linux

The process of setting up SSL for SUSE Linux starts by activating mod_ssl by means of Yast.

To create a key and self-signed certificate:

1. Log in to Scalix as root.
2. Start Yast.
3. Navigate to **Network Services | HTTP Server.**
4. Verify that **Disabled** is selected. (Apache2 will need to be started manually.)
5. Select **Modules** and click **Edit**.
6. Select **ssl** and click **Toggle Status**.
7. Click **OK**, then click **Finish**.
8. To create a test SSL certificate, enter these commands:

```
$ cd /usr/share/doc/packages/apache2

$./certificate.sh
```

9 Follow the on-screen instructions to build the SSL certificate. The resulting certificate files reside in the directories /etc/apache2/ssl*.

Completing the process

To make a copy of the vhost-ssl.template:

1 Log in to [name] as root.

2 Run these commands:

```
# cd /etc/apache2/vhosts.d/

# cp vhost-ssl.template vhost.conf
```

You now need to configure Apache to start with SSL by adding a flag directive to the Apache sysconfig file.

To configure Apache to start with SSL:

1 Log in to Scalix as root.

2 Use your preferred editor and open this file:

```
/etc/sysconfig/apache2 +/APACHE_SERVER_FLAGS
```

3 Edit this line of code:

```
APACHE_SERVER_FLAGS=""
```

to match this example:

```
APACHE_SERVER_FLAGS="SSL"
```

4 Restart Apache:

```
# rcapache2 restart
```

Tip	If you have enabled SuSEfirewall2, do not forget to open the firewall for SSL-enabled Apache on port 443. This can be done via YaST by navigating to Security and Users > Firewall > Allowed Services. Add HTTPS Server to the list of Allowed Services.

Setting up SSL for Red Hat Linux

To create a key and self-signed certificate:

1 Log in to Scalix as root.

2 Run the following command to create your key:

```
# openssl genrsa 1024 > /etc/httpd/conf/ssl.key/server.key
```

3 Run the following command to make sure the permissions are set correctly for the key file:

```
# chmod go-rwx /etc/httpd/conf/ssl.key/server.key

#umask 77 ; \

/usr/bin/openssl req -new -key /etc/httpd/conf/ssl.key/server.key \

-x509 -days 365 -out /etc/httpd/conf/ssl.crt/server.crt
```

4 After entering the passphrase, you are asked for more information. (You do not see a prompt if you created a key without a passphrase.)

5 After providing the correct information, a self-signed certificate is created in /etc/httpd/conf/ssl.crt/server.crt. Restart the secure server after generating the certificate.

```
# service httpd restart
```

6 You can get a real certificate with global validity from vendors such as Thawte (http://www.thawte.com/) or Verisign (www.verisign.com). Instructions are provided for installing the certificate on Apache.

Installing Scalix Tomcat Connector

Because Tomcat handles communications with the Apache Web server, you must install the Scalix Tomcat connector on the Apache server.

To set up an Apache proxy server in a DMZ:

1 Install the Scalix Apache/Tomcat Connector RPM on the Apache server.

2 Copy the contents of the following directory from the Scalix Server to the Apache server.

- For SUSE Linux Enterprise Server 9 or Red Hat Enterprise Linux 3 or 4:

```
~/tomcat/jk/instance-$instance.conf

~/tomcat/jk/app-$instance.webmail.conf

~/tomcat/jk/workers.conf
```

- For SUSE Linux Enterprise Server 10 or Fedora Core 5:

```
~/tomcat/ajp/instance-$instance.conf

~/tomcat/ajp/app-$instance.webmail.conf
```

3 On the Apache server, edit the following file to replace the VirtualHost entry with the hostname of the external server.

- For SUSE Linux Enterprise Server 9 or Red Hat Enterprise Linux 3 or 4:

```
~/tomcat/connector/jk/instance-$instance.conf
```

- For SUSE Linux Enterprise Server 10 or Fedora Core 5:

```
~/tomcat/connector/ajp/instances-$instance.conf
```

4 Restart Apache.

- For Red Hat Enterprise Linux

```
service httpd restart
```

- For SUSE Linux Enterprise Server

```
/etc init.d/apache2 restart
```

Opening Ports

The default port for SSL and TLS requests on the Web server side is 443. There is no conflict between a regular Apache listening on port 80 and an SSL/TLS-enabled Apache listening on port 443. In fact, HTTP and HTTPS can be run with the same Apache instance. Usually one virtual host is used to dispatch requests to port 80 and port 443 to separate virtual servers.

To open ports:

1 Open the following ports in the firewall between the Internet and the DMZ.

- 443 - HTTPS
- 80 - HTTPS (only if you want to tolerate the risk)

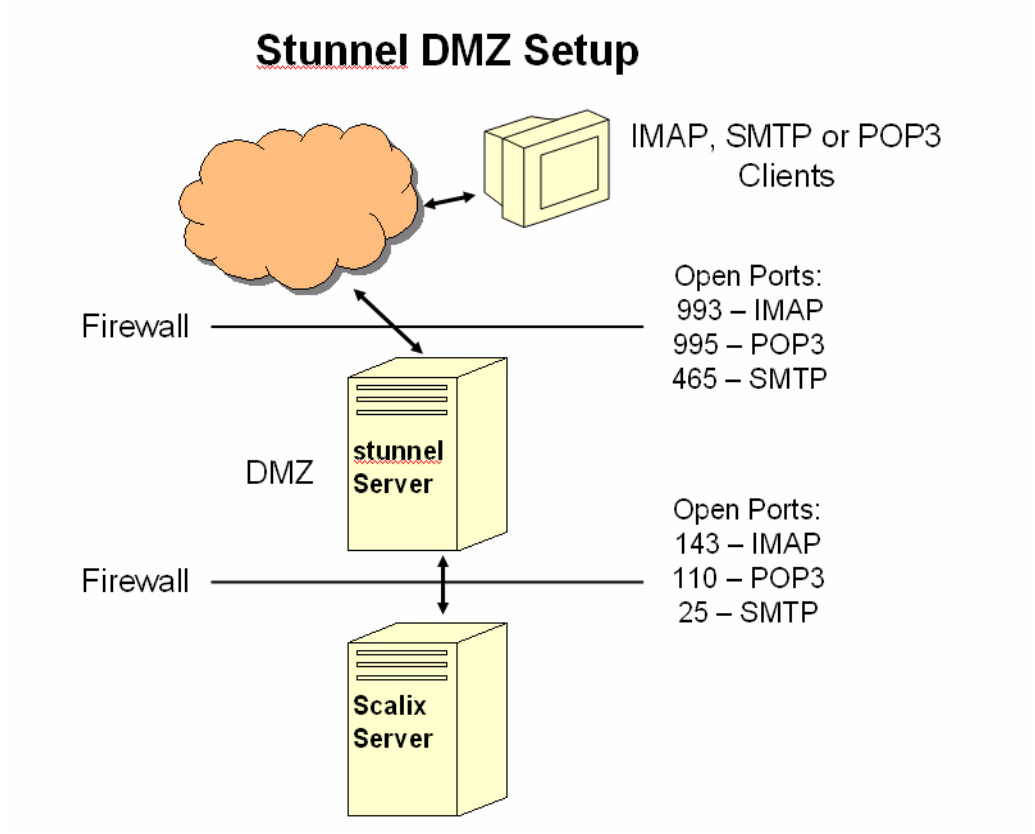
Alert	Scalix recommends closing port 80 to external traffic. Port 80, however, must remain open internally as some Scalix services require it

2 Open the following ports in the firewall between the DMZ and the internal network.

- 8009 - AJP (the Apache/Tomcat connector)

Using stunnel

If VPNs do not work in your unique circumstances, you can use stunnel as a last resort for securing IMAP, POP3 or SMTP communications. That setup would look like:



Overview of stunnel

The stunnel program is a generic open-source SSL encryption wrapper that works on both client and server sides. It can be used to add SSL functionality to commonly used POP3 and IMAP clients such as Outlook Express, Mozilla or Apple Mail without any changes in the programs' code. It supports standard SSL encryption with three levels of authentication.

The stunnel program protects against interception or manipulation of data by intermediate hosts. If compiled with libwrap support, it also protects against IP source routing, where a host can pretend that an IP packet comes from another, trusted host; or DNS spoofing, where an attacker forges name server records.

It does not protect against anything that compromises a host's security. Once an attacker gains root access to a machine, he can subvert stunnel, too.

It is supported on both Unix/Linux and Windows, and can be installed as a Windows Service.

On the server side, it can wrap any application based on port-forwarding, inetd integration or command execution.

On the client side, it generally uses port-forwarding.

Running stunnel on Linux

The stunnel program is available as an RPM that is distributed with RedHat. It requires OpenSSL for encryption handling.

To install stunnel:

1 Install the RPM file.

2 Generate Certificate(s).

3 Configure application Integration.

4 Ensure stunnel is started at boot time.

Security Note: Do NOT use the Certificate provided with stunnel/RedHat installation. This will NOT be secure.

Configuring stunnel for POP3, IMAP and LDAP

Because POP, IMAP and LDAP are implemented as a Scalix service or daemon, wrapping based on command line or inetd is not supported. Instead, use port forwarding mode.

This is done by putting a “service” section into stunnel.conf, e.g. for POP3:

```
[pop3s] # freely chosen service name

accept  = 995 # standard POP3S port number

connect = 110 # standard POP3 port number
```

The drawback to this is that the standard ports remain available.

Securing SMTP

SMTP can be secured using an stunnel wrapper just as with any other service. The standard port number for SMTPS is 465.

However, normally the same SMTP server (sendmail or Scalix SMTP Relay) is used both for incoming traffic from other domains and for Standard Client Mail submission.

As MTA-MTA traffic will never run over SSL, non-secure SMTP still has to be allowed.

As Sendmail supports TLS, both secure and unsecure traffic can be handled over the same port in a well-defined way.

Other forms of Security

You can employ other means of securing your system, including “hardening” it or installing SSH tunnel. For more on those methds and others, see the Scalix Wiki at http://www.scalix.com/wiki.

Advanced Setup and Configuration

About This Section

The remaining chapters in this guide involve more advanced setup and configuration tasks. Those include such procedures as configuring routing between servers, integrating with external directories such as Active Directory or LDAP, setting up multiple server environments and more.

This Section's Contents Include:

Included in this section are the following topics:

Integrating with Active Directory

This chapter covers ways to integrate Scalix with Microsoft Active Directory.

Contents

This chapter includes the following information:

Integrating with Active Directory

If you want to manage some or all of your Scalix accounts with Microsoft Active Directory, you can do so after installing a series of special Scalix schema and GUI extensions.

There are several distinct tasks to execute when integrating Scalix with Active Directory. They are:

- Install and run the application known as Scalix*ForestPrep* to add schema extensions to Active Directory.
- Install the Scalix Active Directory GUI extensions on every administrative workstation running Active Directory.
- Create and test a synchronization agreement between Scalix and Active Directory, then schedule a regularly-occurring synchronization. For this, you use the omldapsync command.
- [Optional] Activate authentication between Scalix, Active Directory and your Kerberos-based security system.

Each of these tasks is detailed separately in the following sections. Once these are finished, you can use Active Directory to create and manage Scalix users.

Installing the Schema Extensions

The first step in integrating Scalix and Active Directory is installing the Scalix schema extensions to Active Directory. This extends the Active Directory schema with new Scalix-specific object classes and attributes that allow you to remotely manage your Scalix-based users and groups with Active Directory.

As part of this procedure, you install the ScalixForestPrep application, then run the application to install the extensions.

Errors, if any, are logged into the Event Viewer, providing you with a permanent record in case of Scalix/Active Directory problems that you suspect are related to the extensions.

ScalixForestPrep finds which domain controller is functioning as the Schema Master. It then attempts to apply all extensions to this domain controller.

For descriptions of these extensions, see “Scalix Active Directory Extensions ” on page 92.

Alert	Remember that adding extensions to Active Directory is irreversible.

To install the ScalixForestPrep exe file and then the AD extensions:

1 Using an administrator account with schema administrator rights, log in to the host computer on which the schema master is stored. Or log in to a workstation with access to the schema master host.

2 Start the Scalix Active Directory Extensions installer. It’s located in the tarball under the directory, software/scalix_ade.

```
./Scalix AD Schema Extensions.msi
```

3 Work through the installation wizard. When installation is complete, ScalixForestPrep is located in the directory:

```
c:\Program Files\Scalix\Administration\
```

4 Run the ScalixForestPrep application to install the extensions:

```
ScalixForestPrep.exe --install
```

5 If the installation is successful, an “update successful” message appears in the window.

6 Exit the terminal window.

Note	You can run the application ScalixForestPrep.exe with several parameters for different results. With --install, it installs the extensions. With --register, it registers the GUI extensions. With --force, it forces the reinstallation of the schema changes. If you run it without any parameters, it only gives you the status.

Alert	The new schema extensions do not become active until the mandatory waiting period expires. The length of that waiting period depends on your unique setup, but is always at least five minutes. This interval is maintained so that all additions or changes will not upset current processes. Active Directory may take a long time to disseminate the new Scalix extensions through the system. Key factors include the number of domain controllers, the number of Active Directory servers and connection speeds between network resources. Additionally, the older the Windows OS underneath AD, the slower the full update will be; Windows 2000-based systems will require a complete Active Directory database resynchronization while Windows 2003-based systems will take less time to propagate changes. Your particular Active Directory system may be updated in minutes—or may take a weekend.

Installing the ADUC GUI Extensions

Now you must update all Active Directory workstations with the Scalix ADUC GUI enhancements, which install several special Scalix tabs in the Active Directory Properties dialog box. These options are relevant for users or groups on the Scalix system.

There are several ways to do this:

- Use Active Directory itself (via GPO) to propagate the GUI enhancements.
- Distribute the Scalix installer for individual per-station installation, network-accessible file sharing or Web FTP.
- Use a third-party utility to script a mass installation that installs the extensions when an administrator logs into the Active Directory server.

If you use the Scalix installer option:

1. For a first-time installation of the Scalix GUI enhancements on an ADUC workstation, log in using a Windows administrator with local admin rights.
2. If it's not already present, copy the file "Scalix AD Extensions.msi" to the workstation desktop.
3. Start Scalix AD Extensions.msi.
4. Work through the wizard.
5. Click **Finish** when the process is complete.
6. ADUC is now ready for Scalix account management.

Setting Up Synchronization Agreements

Before the Scalix system can communicate with the Active Directory server, a custom synchronization agreement must be created and configured using the omldapsync command. Once this agreement has been tested successfully and run at least once, you can implement a cron job to automatically run synchronization on a regular basis.

Requirements for running synchronization agreements between Scalix and AD:

- Log in to the Scalix server as root.

- Have the domain names of the Active Directory and Scalix servers.
- Have the domain name of the server with the Scalix Administration Server installed.
- Have the authentication ID and related password for the Scalix administrator.
- Have the administrator ID and related password for the Windows/AD server.

To prepare and test a new synchronization agreement:

1. Log in to the Scalix Server as root.
2. To run synchronization in "interactive" mode, enter this command at the prompt:

   ```
   omldapsync -i <syncid>
   ```

   Where <syncid> is a unique name for your Active Directory-Scalix synchronization agreement. The name should be no more than six alphanumeric characters in length; for example, AD_SX1.

   After you press Enter, the synchronization "common tasks menu" appears, followed by a numbered list of interactive setup and administrative tasks.
3. When the omldapsync menu appears, enter "1" (one) at the prompt, and press **Enter**.

   The omldapsync command creates the subdirectory for the newly named synchronization agreement along with the <agreement_name>.cfg file.
4. At the next prompt, you are asked to select the synchronization agreement type. Enter "11" (eleven) at the prompt and press **Enter**.

   ```
   Select sync agreement type to create (00):
   ```
5. The first of a series of interactive configuration prompts now appears:

   ```
   INPUT: value for SCALIXHIDEUSERENTRY (scalixHideUserEntry):
   ```

   Press Enter to accept the default value for this prompt and for all of the following value queries, listed below:

   ```
   INPUT: value for SCALIXHIDEUSERENTRY (scalixHideUserEntry):

   INPUT: value for SCALIXMAILBOXCLASS (scalixMailboxClass):

   INPUT: value for SCALIXLIMITMAILBOXSIZE (scalixLimitMailboxSize):

   INPUT: value for SCALIXLIMITOUTBOUNDMAIL (scalixLimitOutbound-
   Mail):

   INPUT: value for SCALIXLIMITINBOUNDMAIL (scalixLimitInboundMail):

   INPUT: value for SCALIXLIMITNOTIFYUSER (scalixLimitNotifyUser):

   INPUT: value for EX_SCALIX_MAILBOX (scalixScalixObject):

   INPUT: value for EX_SCALIX_MAILNODE (scalixMailnode):

   INPUT: value for EX_SCALIX_MSGLANG (scalixServerLanguage):

   INPUT: value for EX_SCALIX_ADMIN (scalixAdministrator):

   INPUT: value for EX_SCALIX_MBOXADMIN (scalixMailboxAdministrator):
   ```
6. When this prompt appears:

```
Edit config file now y/n (n):
```

Press “Y” for Yes.

7 When this prompt appears:

```
Use vi to edit y/n (n):
```

Press “N” to be guided through an interactive session, in which you can efficiently enter the configuration settings. (The option is to press “Y”, and use VI to edit the configuration file manually—which is not documented in this guide.)

The rest of this procedure details the interactive sequence of queries.

8 The first configuration prompt (JAVA_HOME) asks for the location of the Java installation on the Scalix server. Enter the full pathway for the Java directory.

9 The next prompt (EX_HOST) asks for the remote LDAP server name. Enter the name of your Active Directory server.

10 The next prompt (EX_LOGON) asks for the Active Directory administrator account name. The format for your entry should be:

```
cn=administrator,dc=organization,dc=com
```

11 The next prompt (EX_PASS) asks for the related Active Directory Admin password. Be sure to enter “1” (one), so that the synchronization can be fully automated.

12 The next prompt (IM_HOST) asks for the fully qualified domain name (FQDN) of the Scalix server on which the directory will be stored. If you have one server, enter that domain name. If you have several servers in your Scalix system, enter the FQDN of the server on which Scalix Administration Server is running.

The format should be

```
server_name.domain.com
```

13 The next prompt (IM_CAA_URL) asks for the URL of the Scalix server on which Administration Server is running. If you have one server, enter that URL.

The format should be:

```
http://<your_scalix_mailserver_FQDN>/caa/
```

14 Be sure to end the URL in a slash, as shown above.

Note	If you are setting up synchronization on a Scalix server running v10 of Scalix, enter a URL without the 8080 port number: http://<your_scalix_mailserver_FQDN>/caa/

15 When the next prompt (IM_CAA_KEYSTORE) appears, press Enter to accept the default of no entry.

16 When the next prompt (IM_CAA_ID) appears, enter the authentication ID for a full Scalix administrator. The authentication ID is separate from the administrator’s mailing address or display name.

17 When the next prompt (IM_CAA_PASS) appears, type the password associated with the Scalix administrator authentication ID.

Note	Ideally, you will already have verified the usability of this authentication ID and password by logging in to Scalix with Scalix Administrative Console using this administrator account.

18 When the next prompt appears (EX_BASEn) [*with "n" being replaced by a number*], enter the container name and its full LDAP suffix, as shown here:

```
EX_BASE1:cn=users,dc=scalix,dc=com
```

- If needed, you can list up to nine sequentially numbered containers at this time, if used for Scalix users and groups on Active Directory.

19 When this prompt appears:

```
Compare old config to new y/n (?):
```

Type "Y" for Yes.

omldapsync displays a summary of this new configuration on-screen.

20 When this prompt appears:

```
Replace old config with new y/n (?):
```

Type "Y" for Yes.

A series of status messages now appear, noting that the updated file was "installed".

21 When this prompt appears:

```
Attempt to test data extraction now y/n (n):
```

Type "Y" for Yes.

Omldapsync now initiates a non-destructive test of the synchronization communication parameters. No user data will be downloaded from Active Directory to Scalix at this time.

- A series of status messages appears, as omldapsync contacts both servers and establishes the connection.

22 If the test is successful, this message appears:

```
[DATE TIME] STATUS: Configuration of [AGREEMENT_NAME] completed
```

23 If the test fails, you will want to edit the configuration file to correct the problem entry, then re-test the data extraction.

The "configuration completed" message is followed by the omldapsync interactive menu.

24 Press "2" (number two) to start loading all the Active Directory-specific users in a Scalix directory.

- After the synchronization is initiated, a series of status messages report the success of various synchronization actions: new users added, users deleted, new limits applied, etc. You should review this list for the "entries failed" counts in each category.

25 If the download is unsuccessful, you may see a direction to a log file, a SOAP failure report with details, or a prompt to run an omldap utility that will help you fix the problems—after which you can re-start the users download again.

26 When the loading is complete, another series of status messages concludes with:

```
LDAP dir sync export [AGREEMENT_NAME] completed
```

If the synchronization is successful, your Scalix server now hosts a set of users and groups managed by Active Directory.

27 You should now set up a cron job to run this omldapsync agreement at the regular time intervals of your choosing.

Alert!	This newly configured Active Directory/Scalix synchronization is uni-directional; Active Directory records are downloaded to Scalix. This means that you can use Scalix utilities to fully manage Scalix-generated user and group records, but you should only use Active Directory to manage all your Active Directory-generated/controlled records. Changes made with other utilities will be erased in the next synchronization.

Manually Running Synchronization Agreements

To manually run sync agreements:

1 To manually run a synchronization agreement at any time, log in to Scalix, then enter this command:

```
omldapsync -u [AGREEMENT_NAME]
```

2 The Active Directory directory downloads to Scalix, and when finished, a series of status messages ends with this line:

```
LDAP dir sync export [AGREEMENT_NAME] completed\
```

Using Active Directory to Manage Scalix Mailboxes and Groups

Once Scalix and Active Directory are integrated, you can manage your Scalix users and groups in the same way you would manage Microsoft users except for the following procedures, which are unique to the Scalix system and therefor handled through the Scalix extensions:

- Adding and removing Scalix mailboxes
- Setting mailbox types (premium vs standard users)
- Assigning mailnodes
- Setting mailbox limits
- Establishling message language
- Granting administrative access
- Hiding user entries

Alert	Remember that deletion of users and groups is final, and totally erases all records and associated data.

Alert

You can use the Scalix CLI to open and change Active Directory-specific records on the Scalix server, but any changes you make are overwritten in the next Active Directory/Scalix synchronization. Remember, you can use Scalix utilities to fully manage Scalix-generated user and group records, but you should use Active Directory to manage all your Active Directory-controlled records.

Managing Mailnodes and Email Domains in Active Directory

The first step after integrating Active Directory with Scalix is to identify your mailnodes and email domains.

Mailnodes, a unique Scalix feature, can be used to organize your mail user community into manageable groups; for example, by work group, department or employment status. Each Scalix server is associated with a single mailnode, which is created during installation, but they can be subdivided into multiple sub-nodes later if needed. For more on mailnodes, see “Managing Mailnodes” on page 40.

Email domains must be entered again now because Scalix has to verify their licensing, which is tracked by domain.

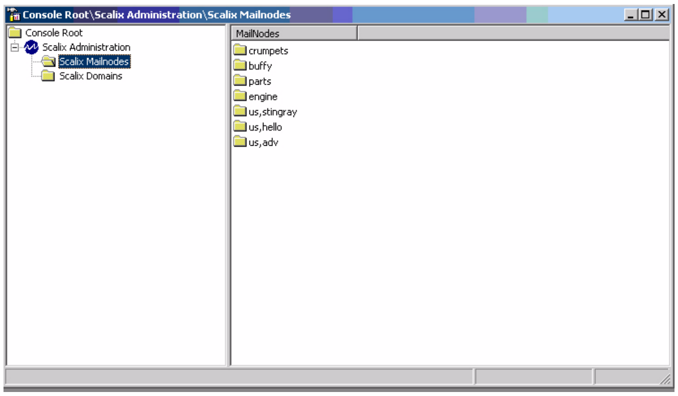
To manage Scalix mailnodes and domains in Active Directory:

1 Log in to the domain with administrative privileges.

2 Launch the Microsoft Management Console and using the Add/Remove Snap-in feature, add the Scalix Management Console snap-in.

3 You see a hierarchy on the left with two nodes:

- Scalix Mailnodes
- Scalix Domains



4 To add a mailnode, you have two options:

- To add a single mailnode: Right click **Scalix Mailnodes** and select **Add** from the menu. Type the name of the node in the dialog box.
- To add multiple mailnodes: Right click **Scalix Mailnodes** and select **Import** from the menu. Browse to the location of the file containing the list of mailnodes.

5 To add a domain, you have two options:

- To add a single mailnode: Right click **Scalix Domains** and select **Add** from the menu. Type the name of the node in the dialog box.
- To add multiple mailnodes: Right click **Scalix Domains** and select **Import** from the menu. Browse to the location of the file containing the list of mailnodes.

Creating New Scalix Mailboxes and Groups within Active Directory

Once you've integrated Active Directory and Scalix, the existing AD "New User" and "New Group" wizards offer additional screens that allow you create mailboxes that are enabled for use with Scalix.

To create a new user or group:

1. Using either the context menu or the menu bar icons, create a new user or group as you would normally do.
2. Advance through the New User or New Group wizard. When the Scalix screen appears, fill in the fields as defined below: In most cases, the screens pre-populate with the needed information.
   - **Create a Scalix Mailbox**: Should be checked by default
   - **Home Mailnode:** Default should prepopulate
   - **Email Address:** Select whether you want an auto-generated addressed or want to create addresses manually
   - **Mailbox Type:** The choices are Premium, Standard or Internet. For more on the difference between the two, see "About Scalix User Types" on page 13.
3. When satisified that all is correct, Click **Next**.
4. You get a summary page. Review the Scalix options you selected and click **Finish**.

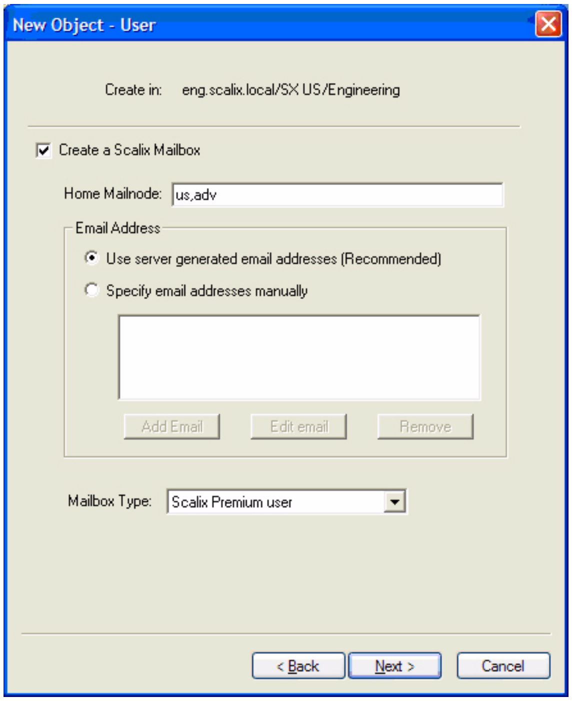
Adding a Mailbox to an Existing User

If the user already exists in the system but does not yet have a mailbox, you can give him or her one.

To give a mailbox to an existing user:

1. In Active Directory, select the user.
2. Right click the user and select **Create Scalix Mailbox**.

3 You get a dialog box with the same fields as outlined above. Fill in or change those fields as needed.



New Object - User

Create in: eng.scalix.local/SX US/Engineering

☑ Create a Scalix Mailbox

Home Mailnode: us,adv

Email Address

● Use server generated email addresses (Recommended)

○ Specify email addresses manually

Mailbox Type: Scalix Premium user

4 When finished, click **OK**.

Removing a Scalix Mailbox

You can delete a user's or group's Scalix mailbox and all its contents, while retaining the user or group record in Active Directory. The following procedure will result in an Active Directory account record that used to be associated with a Scalix server mailbox; on completion, there will be no mailbox for the user or group, and all Scalix data will be deleted.

For example, you may want to perform this task after migrating the Scalix mailbox to another, separate server.

In the next synchronization, the mailbox and its contents are deleted. The Active Directory account remains for other uses.

To remove a Scalix mailbox:

1 In Active Directory, select the user.

2 Right click the user and select **Remove Scalix Mailbox**.

3 At the warning, click **Yes**.

Modifying a Scalix Mailbox

Once created in Active Directory, you can modify users and groups if needed. The Active Directory Properties box now contains two new tabs, **Scalix General** and **Scalix Advanced**, that allow you to modify the following mailbox properties:

- Mailnode
- Email address
- Mailbox type (Premium vs Standard and Internet)

- Server language
- Mailbox size limits and warning settings
- Administrative access
- Hide User

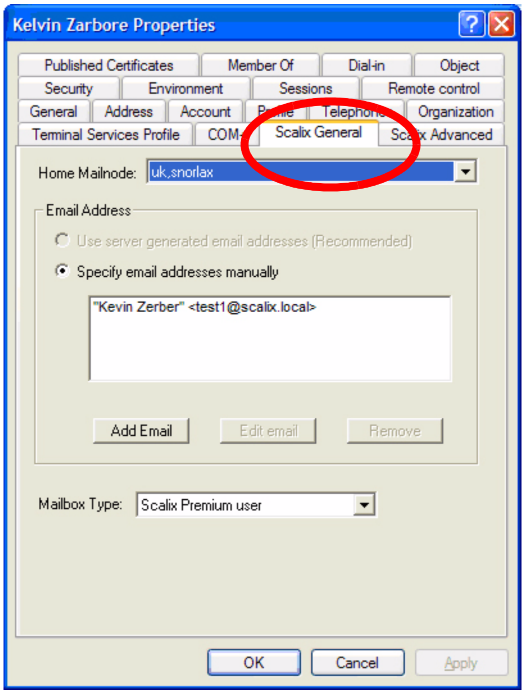
When working with users, both tabs appear in the Properties dialog box. When working with groups, only the Scalix General tab appears.

To modify a user's Scalix attributes:

1 In Active Directory, select the user.

2 Right click the user and select **Properties**.

3 To change the mailnode, address or mailbox type, select the **Scalix General** tab.



Home Mailnode: uk,snorlax

Email Address

○ Use server generated email addresses (Recommended)

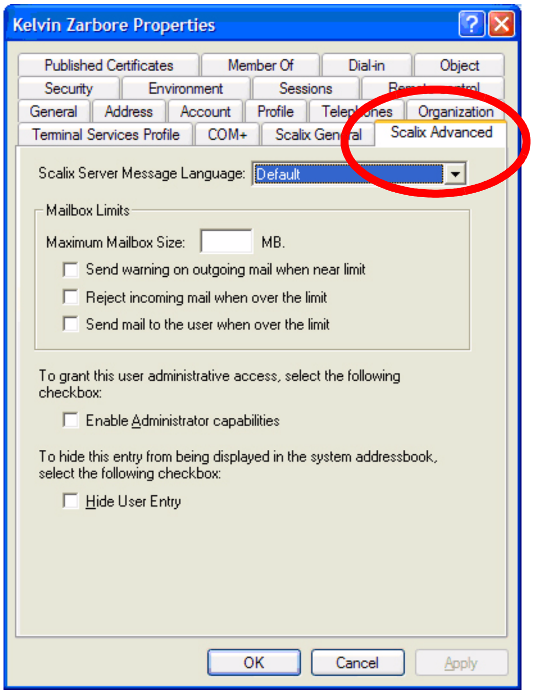
● Specify email addresses manually

"Kevin Zerber" <test1@scalix.local>

Mailbox Type: Scalix Premium user

4 Modify any of the information in the fields.

5 To change the language, mailbox size limits, administrative access level or display status, select the **Scalix Advanced** tabIn the **Scalix Server Message Language** field, select the desired option from the drop-down menu.



Scalix Server Message Language: Default

Mailbox Limits

Maximum Mailbox Size: MB.

□ Send warning on outgoing mail when near limit

□ Reject incoming mail when over the limit

□ Send mail to the user when over the limit

To grant this user administrative access, select the following checkbox:

□ Enable Administrator capabilities

To hide this entry from being displayed in the system addressbook, select the following checkbox:

□ Hide User Entry

6 Under **Mailbox Limits**, type in the maximum size (in MB) of the mailbox and check the boxes next to the action you want the server to take if the user nears or exceeds the maximum.

7 To grant the user administrator rights, click the checkbox next to **Enable Administrator Capabilities**.

8 To hide this entry from the system address book, click the check box next to **Hide User Entry.**

9 Click **OK**.

Scalix Active Directory Extensions

Scalix has the OID root of 1.3.6.1.4.1.19049, and all of the following extensions are appended to it. The first table shows the extensions that match the options in the Scalix Server tab (esp. users).

Table 1: Active Directory Extensions and their Definitions

Extension	Definition
[1.1.10] scalixScalixObject	True if this is an object managed by Scalix.
[1.1.11] scalixMailnode	The mailnode that is hosting this object.
[1.1.12] scalixAdministrator	True if this user has general admin capabilities.
[1.1.13] scalixMailboxAdministrator	True if this user has mailbox admin capabilities.

Table 1: Active Directory Extensions and their Definitions

[1.1.14] scalixServerLanguage	The language for server to client communications.
[1.1.15] scalixEmailAddress	A multivalued list of email addresses for this mailbox.
[1.1.16] scalixLimitMailboxSize	The maximum size of the mailbox in MB -- 0 to use server default.
[1.1.17] scalixLimitOutboundMail	True if Scalix will warn when near limit on outbound mail.
[1.1.18] scalixLimitInboundMail	TRUE if Scalix will reject inbound mail upon limit reached.
[1.1.19] scalixLimitNotifyUser	TRUE if Scalix will notify user when limit is reached.
[1.1.20] scalixHideUserEntry	TRUE if this directory entry is to be hidden from the CDA.
[1.1.21] scalixMailboxClass	Set to “full” or “limited” to control class, or leave it blank for default.

The following table shows the Scalix object classes that extend the Active Directory OpenLDAP schema.

Table 2: Object Classes and their Definitions

Directory Object Class	Definition
[1.2.10.23] scalixUserClass	Auxiliary class of attributes to extend the User class
[1.2.11.24] scalixGroupClass	Auxiliary class of attributes to extend the Group class

Integrating with an LDAP Directory

This chapter covers ways to integrate Scalix with an LDAP directory.

Contents

This chapter includes the following information:

About the LDAP Server and Directories

Server

The LDAP Server is a Scalix daemon process that provides an interface to enable LDAP clients to store and retrieve data from a Scalix directory without having any information about the operation of Scalix.

The LDAP directory service is based on a client-server model. The LDAP Server provides LDAP clients access to shared Scalix directories that do not have an associated password.

Scalix automatically enables search-only LDAP support. Consequently, there is minimal configuration required to enable LDAP client directory searches. The LDAP Server process (omslapd) starts when Scalix starts and runs until Scalix is shut down.

The LDAP directory is a hierarchical tree-like structure comprised of one Scalix directory containing structural information and one or more additional Scalix directories containing user and entity information. Using this structure, the LDAP Server provides a hierarchical view of a Scalix directory, enabling LDAP clients to access directory entries.

An entry is referenced by its Distinguished Name (DN), also known as a directory Distinguished Name (DDN), which is an unambiguous identifier for that entry. The DN is constructed from a Relative Distinguished Name (RDN).

Directories

The LDAP directory service model is based on entries. An entry is a collection of attributes that has a name, called a Distinguished Name (DN). The DN is used to refer to the entry unambiguously. Each of the entry's attributes has a type and one or more values. The types are strings, like "cn" for common name, or "mail" for e-mail address. The values depend on what type of attribute it is. For example, a mail attribute might contain the value `John-Doe@Acme.co.uk`.

While LDAP has a hierarchical structure like a tree, the Scalix directory has a flat structure like a telephone directory. The Scalix directory is made up of a series of entries identifying a user (or entity) by attributes. Attributes include O/R Address attributes, personal and employment related attributes, and e-mail address attributes among others. Traditionally, the Scalix directory is a single, flat database. Entries are not grouped into any hierarchal structure.

Configuring the LDAP Server

The behavior of the LDAP Server is controlled by a number of configuration files that allow you to customize the operation of the LDAP Server. They are:

Table 1: LDAP Server Configuration Files and their Descriptions

File Name	Description
ldap.attribs	LDAP attribute mapping file. This file defines the mapping between LDAP attribute names and Scalix internal attribute names.
slapd.conf	LDAP Server configuration file. This file sets options that control the runtime behavior of the LDAP Server.
dit.cfg	Directory configuration file. This file specifies the name of the Scalix directory and the default DN suffix used by some of the Scalix directory commands.

To change LDAP server configuration files:

1 Go to the appropriate configurations file located under:

```
~/sys/
```

2 Using an editor such as vi, open the file and change the value you want configured.

3 Restart the LDAP server.

```
omoff -a slapd
```

```
omon -a slapd
```

Starting and Stopping the LDAP Server

The LDAP Server process (`omslapd`) starts when Scalix starts and runs until Scalix is shut down. If required, you can stop the LDAP Server.

To stop the LDAP server:

1 Enter the following command.

```
omoff -d delay -a slapd
```

Where delay indicates the time in seconds to wait before stopping the daemon.

To start the LDAP daemon process:

1 Enter the following command.

```
omon -a slapd
```

LDAP and Scalix Attribute Type Mappings

All they really need to know is that there are mappings that are automatically put into the omldapsync command. If they go with the defaults, they're fine. If they want to do custom mappings, they can look on the Wiki or contact customer support.

LDAP Commands

The following table lists and describes LDAP commands:

Table 2: LDAP Commands and their Descriptions

Command	Description
omldapadd	Add one or more entries to an LDAP directory
omldapdelete	Delete one or more entries from an LDAP directory
omldapmodify	Modify an LDAP directory entry
omldapmoddn	Modify the DN of an LDAP entry
omldapsearch	Search an LDAP directory

Multiple Server Environments

This chapter introduces multi-server environments, including how to set up a high availability system with failover, distribute roles among servers, set up their mail routing, synchronize their directories and public folders, and designate their trust relationships.

Contents

This chapter includes the following information:

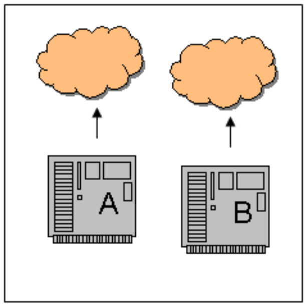
Distributed Architecture

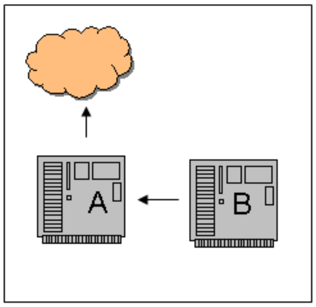
You can set up Scalix as a distributed system with multiple servers for backup, failover, scalability, geographical or performance reasons. Before beginning, though, you must decide what role each server will take, what their relationships will be and how they will interact. The key decisions you need to make include:

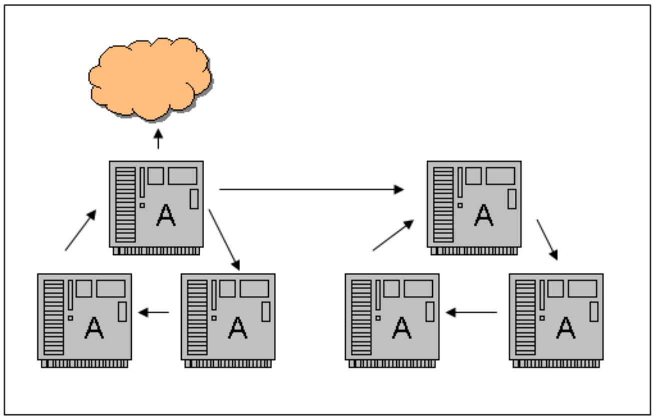
- Which server will host the Scalix Management Console (SAC). This machine also acts as the manager for all other servers.
- How to do directory and public folder synchronization. If the system only has two directories or two sets of public folders, they have an equal import/export relationship and sync with each other. But with three or more servers, you must appoint one directory as the master and have all others synch in to that one. This "master" directory can be the same machine as the "ubermanager" but it doesn't have to be.
- How you want mail routing done. The possibilities are numerous and include:
    - Each server routes to the Internet.
    - One server acts as the smarthost and all outbound messages go through it.
    - One server acts as a hub and all other servers route through it.

  - All servers route to each other.
  - Some servers route to each other and all others report in to those.
- Which server will act as the gateway into the system. The gateway is the first point of contact for incoming mail.

Some possible routing, smart host and gateway setups include:







Routing Mail

The first step in putting together a multiple-server environment is setting up mail routing between the different hosts.

There are many different routing possibilities that run the gamut from one extreme: Every server routs in to one hub, to the other extreme: Every server routes to all others.

On Scalix, routing works through the concept of mailnodes: All servers have mailnodes (and some have more than one). All messages have mailnode notations in their headers. Each server can read the mailnode information and those that are designated for routing, forward messages to their appropriate servers according to their mailnode header notations. For more on mailnodes, see the “Glossary” on page 111.

To add the routes between the servers:

Note	The following steps use two servers, Server A and Server B as examples. Server A is known as serverA.domain.com and Server B is serverB.domain.com. Substitute your own values for these. In cases of more than two servers, establish the routing between two first, then repeat the procedure to set up the routing between two others, then so on and so forth until all routes are configured.

1 Run the following commands on each of the two servers.

   On ServerA: `omaddrt -m serverB,mailnode -q SMINTFC -i scalix@serverB.domain.com`

   On ServerB: `omaddrt -m serverA,mailnode -q SMINTFC -i scalix@serverA.domain.com`

2 On both machines, run the following command to set up the Scalix-to-Scalix transport gateway for sending messages between Scalix servers.

```
omoff -d 0 -w router; omon route
```

3 If you are using CNAME DNS records as your hostname (serverA.domain.com is really called something else), you must make some changes to Sendmail and the Scalix SMTP relay configuration before this works. This is because one of the first things Sendmail does is to rewrite outbound addresses to be the A DNS record rather than any CNAME.

   To do this, edit the file ~/sys/smtpd.cfg and set the following:

```
DOMAIN_NAME=real.host.name

LOCAL_NAMES=cname1.domain.com,cname2.domain.com
```

   Where DOMAIN_NAME is the record namefor server A and LOCAL_NAMES is a comma-separated list of CNAME record names for the server.

4 Stop and restart the SMTP Relay.

```
omoff -d0 smtpd; -w;

omon smtpd
```

5 If you are using a smart host configuration, configure Sendmail to send directly to the other Scalix servers rather than going through the smart host. This is done by using the mailertable feature of Sendmail.

6 In */etc/mail*, edit the file *mailertable* and add the following line:

```
real.host.name<TAB>esmtp:[real.host.name]
```

   Where <TAB> is a tab character.

   This tells Sendmail that if any message is sent to @real.host.name, it should use the esmtp mailer to send it to real.host.name. The square brackets (the [] characters) surrounding the host name tell Sendmail NOT to use DNS to determine MX records.

7 To rebuild the mailertable lookup, go to */etc/mail* and run the following command.

```
make mailertable.db
```

8 To ensure that the Scalix rules are added back into the sendmail.cf file, run the following command

```
onsendin
```

9 Restart the sendmail service.

10 Repeat as needed to establish all other routes, substituting server names as required.

Synchronizing Directories

When running multiple directories on different servers, they must be synchronized.

About Directory Synchronization

Synchronization allows you to automatically maintain consistent directory entries across a network. It ensures that whenever you add, modify, or delete an entry at its primary location, the change is applied to other directories throughout the network. Directory synchronization ensures the following:

- Directory entries are always up-to-date throughout the network.
- Fewer messages are incorrectly addressed.
- A minimal amount of time is required to maintain directories.

You can synchronize directories with other Scalix directories or with directories in other mail systems.

In a synchronization agreement between two Scalix directories, the importing server requests updates from the exporting server. The exporting server extracts updates from the relevant directory change log and returns the updates to the importing server, where they are applied to the import directory. This process automatically repeats at set intervals, with the importing server always initiating the exchange.

Often, two synchronization agreements are created between a pair of directories so that a bidirectional link is created. In this scenario, each directory acts as both an import and an export directory.

In cases of three or more directories, one directory must be designated as the master and all others synchronize with that one.

The import and export directory must be consistent in terms of:

- The names of the directories to be synchronized
- The addresses of the importing and exporting servers
- The frequency of the updates

For every local import agreement there must be an associated agreement on the exporting system.

Also, you might have to update the routing table on each system to provide routes to the new O/R Addresses that are made available through synchronization.

Note	Synchronization is not possible with directories acting as X.500 Directory access points (marked as X500 in the omlistdirs command).

Scalix Corporation recommends that the Scalix network administrator research and design the optimal network topology to use for synchronization. For a more in-depth look at directory synchronization, see the *Scalix Administration Guide.*

Creating Directory Synchronization Agreements

The following steps use two servers, Server A and Server B as examples. Server A is known as serverA.domain.com and Server B is serverB.domain.com. Please substitute your own values for these.

For this to work, you need an import agreement on one side and an export agreement on the other.

To create directory synchronization agreements:

1 Set up an import agreement on one server and an export agreement on the other by running the following commands:

```
On ServerA: omaddds -i -m +DIRSYNC/mailnodeB -t "010000 00:00"

On ServerB: omaddds -e -m +DIRSYNC/mailnodeA
```

Where the -t option specifies when this agreement should come into effect. The date format is yymmdd hh:mm.

To do this the other way around:

```
On ServerA: omaddds -e -m +DIRSYNC/mailnodeB

On ServerB: omaddds -e -m +DIRSYNC/mailnodeA
```

2 Program the synchronization process to make the update requests as soon as the service starts, rather than waiting for a timeout. To do this, add the following line to the file *~/sys/general.cfg* on both machines.

```
DS_CUST_SEND_REQ_NOW=TRUE

DS_CUST_MSGQ_TIMEOUT=2
```

3 Restart the dirsync service and enable auditing to see the messages transfer between machines.

```
omconfaud dirsync 15

omoff -d 0 -w dirsync ; omon dirsync
```

4 To check that mail is flowing correctly, review the messages in the directory *~/logs/audit*.

5 Repeat as needed until all directories have synchronization agreements.

Sychronizing Public Folders

When using public folders on different servers, all folder hierarchies must be synchronized.

About Public Folder Synchronization

Public folder synchronization is the process of automatically updating public folders and their contents from one system to another. The process ensures that when you add an item to one public folder, the same item is also added to all equivalent public folders in the network.

Synchronization is managed using synchronization agreements. These define the rules of each exchange. Each agreement defines whether items are imported or exported, and the folders to which the agreement applies. All items within the hierarchy of a specified public folder are included in the agreement.

Synchronization is performed by exchanging mail messages between two public folder servers. Each message adds one item to a public folder. An item is anything that can be added to a public folder, such as messages, calendars, other public folders, or files. The “sending” server is the exporting BB server, the “receiving” server is the importing BB server.

The items on on Server A are master items, because they were originally created on Server A. The same items on on Server B are secondary items, because they are copies of the master. Whether an item is a master or a secondary is important when deleting or modifying items.

In cases of three or more public folder setups, one must be appointed as the master and all others must synch with that one.

Note	Deletion of public folders does not replicate with synchronization.

For a more in-depth look at public folders, see the *Scalix Administration Guide*.

Creating Folder Synchronization Agreements

Matching agreements must exist on the exporting and importing systems before items can be exchanged. Typically, a number of agreements are specified on each system, with each agreement specifying the exchange for several Public Folders.

Note	The basic synchronization procedures for directories and public folders are the same, but with different values

To run public folder synchronization agreements:

1 Set up an import agreement on one server and an export agreement on the other by running the following commands:

```
On Server A: omaddbbsa -i y -m "OMSYNC +BB/serverB,mailnode" -s "BB
subject" -t "010000 00:00"
```

```
On Server B: omaddbbsa -e y -m "OMSYNC +BB/serverA,mailnode" -s "BB
subject"
```

Where the -t option specifies when this agreement should come into effect. The date format is yymmdd hh:mm.

To do this the other way around:

```
On Server A: omaddbbsa -e y -m "OMSYNC +BB/serverA,mailnode" -s "BB
subject"
```

```
On Server B: omaddbbsa -i y -m "OMSYNC +BB/serverB,mailnode" -s "BB
subject" -t "010000 00:00"
```

2 In the file ~/sys/*general.cfg*, change the default interval from one hour to a smaller time period such as one minute (the example below shows one minute).

```
BBS_CUST_CHECK_TIME=1
```

3 Restart the synchronization service and enable auditing to see the messages transfer between machines.

```
omconfaud bbs 15

omoff -d 0 -w bbs; omon bbs
```

4 To check that mail is flowing correctly, review the messages in the directory ~/*logs/audit*.

5 Repeat as needed until all directories have synchronization agreements.

Configuring Outbound Internet Messages

If you want to use one server as a bridgehead to the Internet, configure the other server(s) to route all Internet mail to the first.

To configure Server B to use Server A as a bridgehead:

1 Run the following commands on Server B.

```
omoff -d 0 router

omdelrt -m internet

omdelrt -m internet,tnef

omaddrt -m internet -q SMINTFC -i scalix@serverA.domain.com

omaddrt -m internet,* -q SMINTFC -i scalix@serverA.domain.com

omon router
```

You need both the omaddrt commands because you have the standard MIME route and also the TNEF route.

2 If you don't want outbound mail to go through another Scalix server, leave your *sendmail.cf* configuration as is. If you have another edge (non-Scalix) server responsible for outbound routing, edit *sendmail.cf* as follows:

Replace:

```
DS
```

with

```
DSother.host.name
```

3 Restart the Sendmail service. This routes all non-local mail through to the named server.

4 If you are using *sendmail.mc* for your configuration, replace:

```
dnl define(`SMART_HOST',`smtp.your.provider')
```

with

```
define(`SMART_HOST',`other.host.name')
```

and run the following command in the directory */etc/mail*.

```
make
```

5 Restart the Sendmail service.

Alert

If you make any changes to the sendmail.mc file and run 'make', you *must* run the command omsendin to ensure that the Scalix rules are added back into the sendmail.cf file.

Server Trust Relationships

In multi-server setups, it is essential to establish server trust relationships to enable cross-server delegation, resource booking and more.

When setting up server trust relationships, all servers must be set up to use a Kerberos server. To do that, you must set up the following Kerberos identities:

- **For IMAP:** imap/<Server 2 FQDN>@KERB.DOMAIN <mailto:<Server 2 FQDN>@KERB.DOMAIN>
- **For UAL:** ual-scalix/<Server 1 FQDN>@KERB.DOMAIN <mailto:<Server 1 FQDN>@KERB.DOMAIN>

Where <Server 1 FQDN> and <Server 2 FQDN> take the form of an FQDN such as hostname.domain_name.com. And KERB.DOMAIN must be in upper case letters.

To enable cross server booking or delegation in the opposite direction, simply reverse these directions.

To set up the Kerberos identities for server trust relationships:

1 Add one of the following Kerberos principles to the keytab file on the first server.

For IMAP:

```
imap/<Server 2 FQDN>@KERB.DOMAIN <mailto:<Server 2
FQDN>@KERB.DOMAIN>
```

For UAL:

```
<Server 2 FQDN> ual-scalix/<Server 1 FQDN>@KERB.DOMAIN
<mailto:<Server 1 FQDN>@KERB.DOMAIN>
```

2 Set up the ~/sys/trust file on the first server to contain the following line.

```
imap/<Server 2 FQDN>@KERB.DOMAIN <mailto:<Server 2
FQDN>@KERB.DOMAIN> ASUSER
```

3 Then telnet into the second server and perform the following steps.

a Telnet to server 2.

```
<Server 2 FQDN> 143
```

b You see a system response that looks something like this.

```
* OK Scalix IMAP server 9.3.0.10-alpha ready on two.example.com
```

c Type in the following.

```
1 login delegate pass
```

d You see a system response that looks something like this.

```
1 OK LOGIN completed, now connected to two.example.com
```

e Typein the following.

```
2 namespace
```

f You see the following system response.

```
* NAMESPACE (("" "/")) (("Other Users/" "/")) (("Public Folders/"
"/"))

2 OK NAMESPACE completed
```

g Type in the following.

```
3 select "Other Users/principal@<Server 1 FQDN> <mailto:princi-
pal@<Server 1 FQDN>/INBOX"
```

h You see the following system response.

```
* 7 EXISTS

* 0 RECENT

* OK [UIDVALIDITY 1] UIDVALIDITY value

* FLAGS (\Answered \Flagged \Deleted \Seen \Draft $MdnSent)

* OK [PERMANENTFLAGS (\Answered \Flagged \Deleted \Seen \Draft
$MdnSent)] flags will stay set

3 OK [READ-WRITE] SELECT completed
```

Localizing Scalix

This chapter explains how to localize Scalix for use in other languages.

Contents

This chapter includes the following information:

Overview

Using UTF8 character encoding, Scalix provides full multi-byte language support, coupled with an open-source localization kit for channel partners and customers tofacilitate international deployments..

Note	You can only run one language at a time per Scalix installation.

Localizing Outlook

You can localize the MAPI Connector for any language that Outlook supports.

Tools

For this procedure, you need the following tools:

- Microsoft resource compiler Visual Studio 6.0, including the compiler RC.EXE and the linker LINK.EXE. For more on the compiler, see http://msdn.microsoft.com/library/default.asp?url=/library/en-us/tools/tools/resource_compiler.asp.

Procedures

To localize the MAPI Connector:

1. Use your preferred editing and translation tools to localize the English strings in the file SCLXRES\RESEN.RC and in all sub-files of the SCLXRES\EN subdirectory.

   Make sure you can save the file with the proper font files.

2. From the directory SCLXRES, run the resource compiler. The command is:

   ```
   rc /l<LCID Culture ID> /fo "<file name>\sclxres.res" /i
   "..\shared\include" /i "..\common" /d "THIS_LANG_<NAME OF LAN-
   GUAGE>" sclxres.rc
   ```

   Where:

   - **rc -** The compiler command
   - **<LCID Culture ID> -** Can be found at http://www.microsoft.co.ke/globaldev/nlsweb/default.mspx. This code number must be preceded by an “l” (a lowercase l, not an i) and must match the “THIS-LANG...” option
   - **l -** Specifies default language for compilation. For example, -l409 is equivalent to including the following statement at the top of the resource script file: LANGUAGE LANG_ENGLISH,SUBLANG_ENGLISH_US
   - **fo -** Renames the source file so that it comes out as a .res file
   - **i -** The "include" command so takes a directory as its variable
   - **d -** This variable changes according to the culture ID specified earlier

   Note that in this step, you are compiling the file SCLXRES.RC, which includes all subfiles and subfolders, and creates a new file named SCLXRES.RES to the same directory.

3. When the compiler completes, you see a new file named sclxres.res in the directory SCLXRES.

4. Using the resource compiler's LINK.EXE tool, link the .res file to produce a dll in the same directory that will be called SCLXRES.DLL. The command is:

   ```
   link /nologo /dll /pdb:none /machine:I386 /nodefaultlib /
   implib:"sclxres.lib" /NOENTRY sclxres.res
   ```

5. Copy the new file, SCLXRES.DLL to the MAPI Connector's installation directory (the default installation directory is C:\Program Files\Scalix\Connect).

Note	For more on the link command, see http://msdn.microsoft.com/library/default.asp?url=/library/en-us/vccore98/html/_core_linker_reference.asp.

Localizing SWA

Localization of SWA involves translating two xml files, putting them in the appropriate location, and then restarting Tomcat.

Note	Unless your localization file is part of a Scalix distribution, you MUST do a backup before the next upgrade. If you do not, your file will be lost.

To add a new locale to SWA version 11.0:

1 Translate the files strings_en.xml and strings_en_US.xml to create the files strings_xx.xml and strings_xx_XX.xml. Those strings are located at:

```
~/tomcat/webapps/webmail/WEB-INF/data
```

For example: For Nederlands Dutch, create files named string_nl.xml and strings_nl_NL.xml.

2 Stop Tomcat by running the shutdown script found at:

```
/etc/init.d/scalix-tomcat stop
```

3 If Tomcat has not already unpacked it, uncompress the war file.

4 Put the files strings_xx.xml and strings_xx_XX.xml in the WEB-INF/data directory.

5 Delete the Tomcat cache if needed by deleting the contents of the following directory.

```
~/tomcat/work/Catalina/ directory
```

6 Restart Tomcat.

```
/etc/init.d/scalix-tomcat restart
```

Localizing the Search and Index Service

The Scalix Search Index Service can be configured to process text for any language. To work with different languages, it uses stemming rules for that specific language, which break down words by removing suffixes and endings just as they do with the English language. For example, a search for the English word "singing" will match the word "sing".

To change the language assumed by SIS when it indexes text or builds search queries:

1 Stop Tomcat.

```
/etc/init.d/scalix-tomcat stop
```

2 On the server where SIS is installed, open the following file:

```
/etc/opt/scalix/sis/sis.properties
```

3 Edit the property, sis.language, to one of the following:

- Danish
- Dutch
- English
- Finnish

- French
- German
- Italian
- Norwegian
- Portuguese
- Russian
- Spanish
- Swedish

(English is default)

4 In the file /var/opt/scalix-sis, remove existing indexes and rebuild them using the command sxmkindex.

5 All subsequent index and search operations will use the stemming rules for the given language.

Note	This procedure uses the Snowball Analyzer written by Martin Porter. For more information, see the Lucene API at: http://lucene.apache.org/java/docs/api/net/sf/snowball/ext/package-summary.html.

Using Custom Analyzers

For additional languages, you can use other analyzers, all of which have built-in stemmers.

To use these additional stemmers:

1 In the file sis.properties, specify a custom class of your own.

```
index.content.analyzer.class==com.scalix.index.message.MyRussian-
Analyzer
```

2 The list of potential analyzers includes:

- BrazilianAnalyzer
- ChineseAnalyzer
- CJKAnalyzer
- CzechAnalyzer
- DutchAnalyzer
- FrenchAnalyzer
- GermanAnalyzer
- GreekAnalyzer

3 The class must be implemented like this:

```
import org.apache.lucene.analysis.Analyzer;

import org.apache.lucene.analysis.ru.RussianAnalyzer;

import com.scalix.index.message.MessageAnalyzer;

public class MyRussianAnalyser extends MessageAnalyzer {
    public Analyzer getAnalyzer() {

       return new RussianAnalyzer();

    }

}
```

4 Compile the class and put it in the following directory so that it can be picked up by the Web Apps classloader.

```
~/tomcat/webapps/sis/WEB-INF/classes
```

Glossary

Some terms and acronyms in this manual may be unfamiliar to users. Here are some terms and definitions that are specific to the Scalix product and the Linux platform.

Table 1: Terms and Definitions

Address Directories	In Scalix terminology, the address directories are databases that clients use to look up names and addresses. Scalix directories can hold addresses of both Scalix and non-Scalix users, and other information that an administrator can configure such as job titles and phone numbers. Directories can be searched by any number of attributes.
Management Console or SAC	The Scalix Management Console (SAC) is a browser-based application that enables most day-to-day system administration tasks on a Scalix messaging system through an easy-to-use GUI. It is a separate component of Scalix that users can access with any approved browser on either Microsoft Windows or Linux workstations. SAC provides efficient access to a wide range of Scalix server options, including user account management, starting and stopping server services, administering queues, public distribution list or group management, and changing low-level server configuration settings. It also provides system monitoring to assess the status of processes and resources.
ADUC	(Active Directory Users and Computers) ...
Authentication Identifier	The Scalix system has several ways of identifying users for different purposes: Display names, personal names, authentication IDs and Internet addresses. The display name (also known as a “common name”) is used in Outlook and other clients as the “displayed” address. It can serve authentication purposes and determines the sort order in the Outlook address book. Authentication IDs support the concept of a separate login name and allow for integration with external authentication systems that may have their own naming rules. Internet addresses are SMTP addresses of the form name@domain. Personal names are used for internal addressing of email and are sometimes referred to as “X.400 addresses,” “OpenMail addresses” or “ORN (originator Recipient name.”
Bulletin Board	In Scalix terminology, a bulletin board is a set of public folders where members can share files, ideas, documents and more. They are a shared area in the Scalix message store.
Clam AV	An open source freeware program that protects against viruses.
Community Edition	The free, single-server, unlimited-use version of the Scalix product. Does not include advanced groupware and collaboration functionality.

Table 1: Terms and Definitions

DDR	
Display Names vs User Names vs Personal Names vs authentication ID vs Internet address	The Scalix system has several ways of identifying users for different purposes: Display names, personal names, authentication IDs and Internet addresses. The display name (also known as a “common name”) is used in Outlook and other clients as the “displayed” address. It can be used for authentication purposes and determines the sort order in the Outlook address book. Authentication IDs support the concept of a separate login name and allow for integration with external authentication systems that may have their own naming rules. Internet addresses are SMTP addresses of the form name@domain. Personal names are used for internal addressing of email and are sometimes referred to as “X.400 addresses,” “OpenMail addresses” or “ORN (originator Recipient name.”
Enterprise Edition	The company’s flagship product, which includes multi-server support, unlimited number of Standard users, any number of Premium users, the full complement of Scalix advanced capabilities, and a wide variety of technical support options.
Gateway	Gateways are a way of passing messages out of the Scalix network to different mail environments. The gateway converts outgoing messages from a Scalix format to a format that external services can use to do send processes, and later to a format that target environments can receive such as an SMTP address. Scalix comes with a standard SMTP gateway that converts Scalix-formatted messages to SMTP and vice-versa. This SMTP gateway is called the Unix Mail Gateway or Internet Mail Gateway.
Groups and PDLs	In Scalix terminology, the terms “group” and “PDL” are used interchangeably to mean a group of people organized into a mailing list. PDLs can contain both local and remote users, and can contain nested PDLs.
IMAP	(Internet Message Access Protocol) A standard interface between an e-mail client program and the mail server. In Scalix, the iMAP4 server enables a client to: Access, list, read, and delete items from inboxes, filing cabinets and public folders; read parts of a message without downloading the entire thing, keep a record of which messages have been read, and update messages on the server from a client. IMAP extesnions also provide for calendaring and contact management.
Internet Domains vs mailnodes	Mailnodes have no direct relationship to Internet domains. However, you can set up rules so that when a user is created on a mailnode, Internet address generation kicks in and creates an Internet address for the user. You can map multiple mailnodes to the same Internet domain name.
LDAP	(Lightweight Directory Access Protocol) A protocol used to access a directory listing. In Scalix, the LDAP server is a daemon process based on a client/server model that provides an interface to enable LDAP clients to store and retreive data from a Scalix directory without any information about the operation of Scalix. It provides LDAP clients access to shared Scalix directories that do not have an associated password.
LVM	(Logical Volume Manager) Used for backing up Scalix directories.

Table 1: Terms and Definitions

Mail Nodes	A logical structure used to organize users into administrative groupings. For example, some companies organize their email users by work group whereas others break their users down by employment status. Each Scalix server is associated with a single mail node created during installation. Mailnode names, which also are created during installation, are often the same as host names and sometimes can contain both the hostname and domain name After installation, you can use the Management Console to create additional mail nodes on a server, including customizing any new mail nodes with a specific Internet address or domain name.
MAPI	(Mail API) A programming interface from Microsoft that enables a client application to send to and receive mail from Exchange Server or a Microsoft Mail (MS Mail) messaging system. Microsoft applications such as Outlook, the Exchange client and Microsoft Schedule use MAPI.
Message Store	The message store is a collection of flat Linux files held in file system directories on the Scalix server. It holds new messages received as well as messages in transit. For clients that use the message store (server-based clients), It also holds old messages that are files for reference in folders, copies of outgoing messages, draft messages, private distribution lists, personal information such as calendaring, tasks, bulletin boards, public folders and more.
Mx Records	Mail exchanger records inside DNS servers. These decide which server is responsible for dealing with mail or domain DNS actions.
OpenMail	The original technology, licensed from Hewlett Packard, upon which the Scalix system is based.
O/R or Originator/Recipient Address	An attribute list that distinguishes one user, or distribution list, from another and defines the user's point of access to the message handling system or the distribution list's location.
PAM	(Pluggable Authentication Modules). A standard library in Linux that connects applications that require authentication with shared library modules interfacing with authentication mechanisms.
PDL	In Scalix terminology, the terms “group” and “PDL” are used interchangeably to mean a group of people organized into a mailing list. PDLs can contain both local and remove users, and can contain nested PDLs.
Personal Name	The Scalix system has several ways of identifying users for different purposes: Display names, personal names, authentication IDs and Internet addresses. The display name (also known as a “common name”) is used in Outlook and other clients as the “displayed” address. It can be used for authentication purposes and determines the sort order in the Outlook address book. Authentication IDs support the concept of a separate login name and allow for integration with external authentication systems that may have their own naming rules. Internet addresses are SMTP addresses of the form name@domain. Personal names are used for internal addressing of email and are sometimes referred to as “X.400 addresses,” “OpenMail addresses” or “ORN (originator Recipient name.”
POP	(Post Office Protocol) A standard interface between an e-mail client program and the mail server. The Scalix POP3 server enables clients to list, read and delte items from the inbox area of the Scalix message store. The Scalix POP3 server does not provide access to any other areas of the message store such as public folders.

Table 1: Terms and Definitions

Premium Users	Scalix has two levels of access and usage: Premium and Standard. Premium users have access to the full benefits and functionality of the Scalix email and calendaring system. Standard users gain access to a subset of Scalix functionality including email, personal calendar and contacts through Scalix Web Access and Novell Evolution as well as email access using POP/IMAP clients.
Realm	
SAC	The Scalix Management Console (SAC) is a browser-based application that enables most day-to-day system administration tasks on a Scalix messaging system through an easy-to-use GUI. It is a separate component of Scalix that users can access with any approved browser on either Microsoft Windows or Linux workstations. SAC provides efficient access to a wide range of Scalix server options, including user account management, starting and stopping server services, administering queues, public distribution list or group management, and changing low-level server configuration settings. It also provides system monitoring to assess the status of processes and resources.
Scalix Connect	A MAPI application that enables the use of the Outlook client interface and all of its functionality.
Sendmail	An SMTP-based message transfer agent (MTA) that runs under Unix and Linux. It is the mail transfer process used inside the Scalix system.
SSL	
Small Business Edition	A version of the Scalix system that targets organizations getting started with a commercial version of Scalix that do not have the higher end requirements of Enterprise Edition. It is functionally equivalent to Enterprise Edition except that it allows only single-server installations
SmartHost	
Spam Assasin	An open source freeware program that filters spam.
Standard Users	Scalix has two levels of access and usage: Premium and Standard. Premium users have access to the full benefits and functionality of the Scalix email and calendaring system. Standard users gain access to a subset of Scalix functionality including email, personal calendar and contacts through Scalix Web Access and Novell Evolution as well as email access using POP/IMAP clients.
SWA	Scalix Web Access, the browser-based email, calendar, contacts and public folders client that comes with any Scalix installation.
Transports	Transports are services that Scalix uses to pass Scalix format messages to other Scalix services. Scalix uses Sendmail and SMTP formatted messages to send messages between servers in the Scalix network, but other connections can be written. The transport service on the Scalix server is called the Sendmail Interface.
UAL	(User Access Layer) A proprietary Scalix protocol that enables communication between clients and the Scalix server.
WAP	(Wireless Application Protocol) A standard for providing cellular phones, pagers and other handheld devices with secure access to e-mail and text-based Web pages.