



Building Secure Sun Fire™ Link Interconnect Networks Using Midframe Servers

*Joe Higgins, Enterprise Server Products, High End
Software*

Sun BluePrints™ OnLine—February 2003



<http://www.sun.com/blueprints>

Sun Microsystems, Inc.
4150 Network Circle
Santa Clara, CA 95045 U.S.A.
650 960-1300

Part No. 817-1656-10
Revision A, 1/10/03
Edition: February 2003

Copyright 2003 Sun Microsystems, Inc. 4150 Network Circle, Santa Clara, California 95045 U.S.A. All rights reserved.

Sun Microsystems, Inc. has intellectual property rights relating to technology embodied in the product that is described in this document. In particular, and without limitation, these intellectual property rights may include one or more of the U.S. patents listed at <http://www.sun.com/patents> and one or more additional patents or pending patent applications in the U.S. and in other countries.

This product or document is protected by copyright and distributed under licenses restricting its use, copying, distribution, and decompilation. No part of this product or document may be reproduced in any form by any means without prior written authorization of Sun and its licensors, if any. Third-party software, including font technology, is copyrighted and licensed from Sun suppliers.

Parts of the product may be derived from Berkeley BSD systems, licensed from the University of California. UNIX is a registered trademark in the United States and other countries, exclusively licensed through X/Open Company, Ltd.

Sun, Sun Microsystems, the Sun logo, Sun BluePrints, Sun Fire, Sun Enterprise, Sun HPC ClusterTools, Java, and Solaris are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. in the US and other countries. Products bearing SPARC trademarks are based upon an architecture developed by Sun Microsystems, Inc.

The OPEN LOOK and Sun™ Graphical User Interface was developed by Sun Microsystems, Inc. for its users and licensees. Sun acknowledges the pioneering efforts of Xerox in researching and developing the concept of visual or graphical user interfaces for the computer industry. Sun holds a non-exclusive license from Xerox to the Xerox Graphical User Interface, which license also covers Sun's licensees who implement OPEN LOOK GUIs and otherwise comply with Sun's written license agreements.

U.S. Government Rights —Commercial use. Government users are subject to the Sun Microsystems, Inc. standard license agreement and applicable provisions of the FAR and its supplements.

DOCUMENTATION IS PROVIDED "AS IS" AND ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE DISCLAIMED, EXCEPT TO THE EXTENT THAT SUCH DISCLAIMERS ARE HELD TO BE LEGALLY INVALID.

Copyright 2003 Sun Microsystems, Inc., 4150 Network Circle, Santa Clara, Californie 95045 Etats-Unis. Tous droits réservés.

Sun Microsystems, Inc. a les droits de propriété intellectuels relatants à la technologie incorporée dans le produit qui est décrit dans ce document. En particulier, et sans la limitation, ces droits de propriété intellectuels peuvent inclure un ou plus des brevets américains énumérés à <http://www.sun.com/patents> et un ou les brevets plus supplémentaires ou les applications de brevet en attente dans les États-Unis et dans les autres pays.

Ce produit ou document est protégé par un copyright et distribué avec des licences qui en restreignent l'utilisation, la copie, la distribution, et la décompilation. Aucune partie de ce produit ou document ne peut être reproduite sous aucune forme, par quelque moyen que ce soit, sans l'autorisation préalable et écrite de Sun et de ses bailleurs de licence, s'il y en a. Le logiciel détenu par des tiers, et qui comprend la technologie relative aux polices de caractères, est protégé par un copyright et licencié par des fournisseurs de Sun.

Des parties de ce produit pourront être dérivées des systèmes Berkeley BSD licenciés par l'Université de Californie. UNIX est une marque enregistrée aux États-Unis et dans d'autres pays et licenciée exclusivement par X/Open Company Ltd.

Sun, Sun Microsystems, the Sun logo, Sun BluePrints, Sun Fire, Sun Enterprise, Sun HPC ClusterTools, Java, et Solaris sont des marques de fabrique ou des marques déposées, ou marques de service, de Sun Microsystems, Inc. aux États-Unis et dans d'autres pays. Toutes les marques SPARC sont utilisées sous licence et sont des marques de fabrique ou des marques déposées de SPARC International, Inc. aux États-Unis et dans d'autres pays. Les produits portant les marques SPARC sont basés sur une architecture développée par Sun Microsystems, Inc.

L'interface d'utilisation graphique OPEN LOOK et Sun™ a été développée par Sun Microsystems, Inc. pour ses utilisateurs et licenciés. Sun reconnaît les efforts de pionniers de Xerox pour la recherche et le développement du concept des interfaces d'utilisation visuelle ou graphique pour l'industrie de l'informatique. Sun détient une licence non exclusive de Xerox sur l'interface d'utilisation graphique Xerox, cette licence couvrant également les licenciés de Sun qui mettent en place l'interface d'utilisation graphique OPEN LOOK et qui en outre se conforment aux licences écrites de Sun.

LA DOCUMENTATION EST FOURNIE "EN L'ÉTAT" ET TOUTES AUTRES CONDITIONS, DECLARATIONS ET GARANTIES EXPRESSES OU TACITES SONT FORMELLEMENT EXCLUES, DANS LA MESURE AUTORISÉE PAR LA LOI APPLICABLE, Y COMPRIS NOTAMMENT TOUTE GARANTIE IMPLICITE RELATIVE A LA QUALITE MARCHANDE, A L'APTITUDE A UNE UTILISATION PARTICULIERE OU A L'ABSENCE DE CONTREFAÇON.



Please
Recycle



Adobe PostScript

Building Secure Sun Fire™ Link Interconnect Networks Using Midframe Servers

In a distributed computer system, data is sent from one computer over a network to another computer. When the data is transmitted over the network, the data is sensitive to privacy, authenticity, and point of origin attacks. Deploying a secure distributed computer system can be difficult. Because the Sun Fire™ Link software is a distributed computer system, it is vulnerable to these types of attacks and must be protected.

Keeping the Sun Fire Link secure requires an ongoing effort. This article describes how to install and deploy the Sun Fire Link product so that it can be securely managed and operated. The article presents the software architecture and the steps for securing the Sun Fire Link interconnect. The commands used in configuration steps are either Sun Fire Link Manager (FM) or Solaris™ Operating Environment (Solaris OE) tools. The article requires a general knowledge of Solaris OE system administration.

The article also contains a section on how to create and configure a Sun Fire Link fabric, which is a collection of remote shared memory (RSM) partitions, each of which is made up of compute nodes and switch nodes.

The major elements of the recommendations are:

- Configuring the Administration Software to use Secure Sockets Layer (SSL)
- Configuring the tools to use role-based access control (RBAC)
- Setting up a private management subnet
- Setting up a midframe service processor (MSP)
- Enabling all security features of the software
- Securing the Sun Fire Link switch

This article builds upon the Sun BluePrints™ article “Securing the Sun Fire™ Midframe System Controller” Version 1.3, 10/23/02. For the Sun Fire Link Cluster to be secure you must follow all of the recommendations in this article.

This article is written for system installers and administrators who must install and deploy the Sun Fire Link product so that it can be securely managed and operated.

This article covers the following topics:

- *Sun Fire Link Overview*
- *Sun Fire Software Overview*
- *SSL Primer*
- *Securing the Midframe Sun Fire Link Cluster*

Sun Fire Link Overview

The Sun Fire Link product is a high-bandwidth, low-latency cluster interconnect used with Sun Fire™ 6800 and Sun Fire™ 15K servers to expand high-end Sun Fire series system capabilities beyond the chassis. A Sun Fire Link cluster consists of up to eight Sun Fire 6800 and/or Sun Fire 15K nodes connected to each other by a Sun Fire Link optical network. Each node has a separate instance of Solaris OE software running under a layer of clustering software, which can be either Sun™ Cluster or Sun HPC ClusterTool™ software. This separate hardware will include Sun Fire Link switches too. A Sun Fire Link cluster also requires an Ethernet network to carry cluster administration traffic. It connects all cluster components that exchange control and status/error information. We recommend a dedicated server for running the required management software. The Sun BluePrints article “Securing the Sun Fire Midframe System Controller” discusses the Midframe Service Processor (MSP). The MSP is a dedicated server that restricts access to the private System Controller (SC) network. The MSP is a workstation. Its only function is to serve as a firewall and to run the FM software. The goal of the MSP is to serve as a barrier between the private management network and the open network.

Sun Cluster and Sun HPC ClusterToo software use the remote shared memory (RSM) interface for internode communication across a Fire Link network. RSM is a Sun messaging interface that is highly efficient for remote memory operations. For Sun Fire Link clusters of two or three nodes, the network connections can be point-to-point (direct-connect topology) or through Sun Fire Link switches. For larger clusters (four to eight nodes), Sun Fire Link switches are required. The servers interface to the Sun Fire Link network is provided by an I/O subsystem specific to Sun Fire Links, called the Sun Fire Link assembly. These assemblies are installed in standard server I/O slots. Each Sun Fire Link assembly contains two optical

transceiver modules called Sun Fire Link optical modules. Each optical module supports a full-duplex optical link. The Sun Fire Link assemblies are installed in pairs to enhance availability and to support message striping for higher bandwidth.

.FIGURE 1 shows direct connect topology. Figure 2 shows switched connect topology.

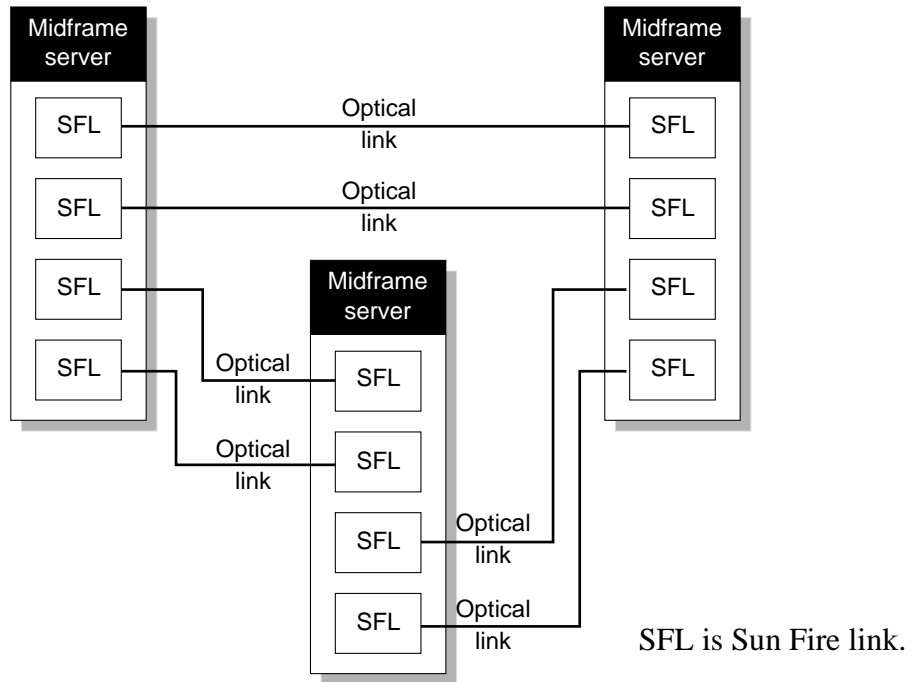


FIGURE 1 Direct Connect Topology

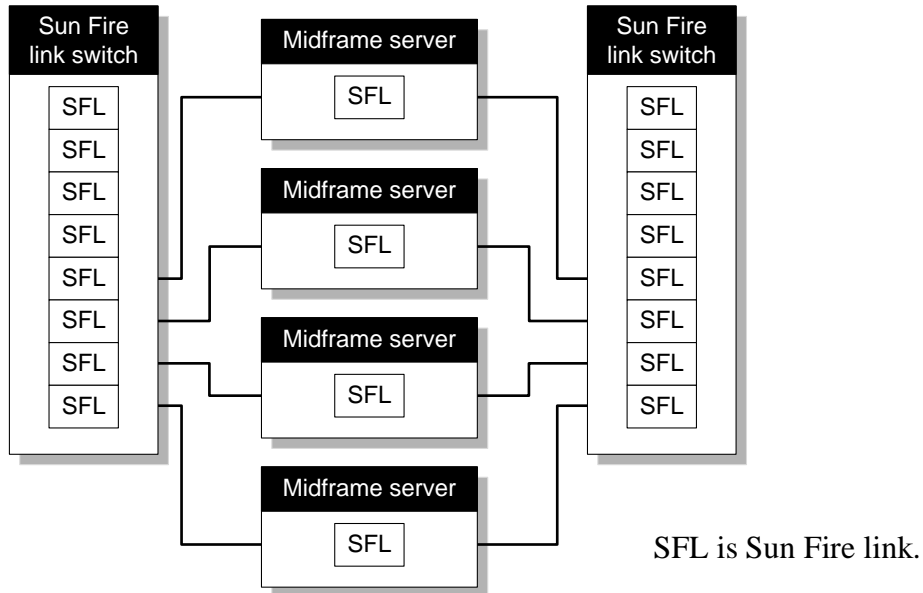


FIGURE 2 Switched Topology

Sun Fire Software Overview

The following are the elements of Sun Fire Link Interconnect software stack:

- Sun Fire Link RSM driver software
- Sun Fire Link switch software
- Midframe SC software
- Sun Fire Link Administration software

Sun Fire Link RSM Driver Software

The Sun Fire Link interface is managed by a device driver that implements the RSM interface. RSM supports operations on remote memory as if it were local. The driver software runs on a domain (compute node) as root. The RSM driver is responsible for interfacing to the Sun Fire Link hardware.

Sun Fire Link Switch and Switch Software

The Sun Fire Link Switch is part of the Sun Fire Link clustering technology. The switch manages the network in a hub and spoke fashion using eight optical ports. For direct-connect topologies, the switch is not part of the partition.; it is only part of a switched topology. The switch consists of an enclosure that includes a backplane and two fan trays.

The Sun Fire Link switch system controller module runs an embedded, real time operating system (RTOS). The switch has limited processing and memory resources and no local storage other than erasable programmable read only memories (EPROMS). The switch system controller (SSC) application software runs on the switch. The *Sun Fire Link Switch Installation and Service Manual* contains more detailed information on the switch and also contains information on the command line interface (CLI) provided by the Sun Fire Link Switch.

The SSC software configures and manages the hardware resources of the switch. For example, the software configures various links and ports to participate in a RSM partition. The switch stores the partition configuration, which is computed by the Sun Fire Link Manager (FM). Network discovery and fabric configuration services are exported to the FM by a private Java remote method invocation (RMI) interface. The FM can remotely execute the functions of the switch. RMI allows client applications to locate remote server objects and execute methods on those objects as if they were local objects. RMI is the object equivalent of remote procedure calls (RPC). RMI also allows code sharing between the client and server application. This shared code is downloaded dynamically between the Java virtual machines (VMs). The switch provides an embedded HTTP server to allow this class code to be downloaded. Currently, the switch SC does not support encrypted or strongly authenticated access methods. All traffic to the switch SC uses non-encrypted transport mechanisms such as Telnet, FTP, HTTP and SNMP. SSL is not used for the RMI so these RMI calls are also insecure. These insecure protocols should not be transmitted across general purpose intranets. These limitations require that the switch be placed on a private subnet. The MSP serves as a gateway system that can communicate between the private subnet and the general access intranet or management network. The switch SC can be accessed only though a serial port on the back of it. Any terminal server used should support privacy and encryption. Running a secure shell on any terminal server is recommended

Midframe System Controller Software

The system controller (SC) of a Sun Fire midframe system controls the assignment of resources within the midframe Sun Fire platform. This includes turning domains on and off and associating components such as CPUs, I/O cards, and memory with domains. All of the server's configuration is stored on the SC. Network discovery and fabric configuration services are exported to the FM software through a private

Java remote method invocation (RMI) interface. These RMI interface methods do not use SSL. To mitigate this risk as well as other SC limitations, the midframe SC should be placed on a private subnet. The blueprint article “Securing the Sun Fire Midframe System Controller” contains a detailed description of this.

Sun Fire Link Administration Software

Sun Fire Link software includes tools for administrating Sun Fire Link networks. The administration of Sun Fire Link networks includes the following tasks:

- Configuring and reconfiguring Fire Link partitions
- Dynamically adding nodes to and removing nodes from partitions
- Bringing up and taking down optical links
- Enforcing domain topology constraints
- Monitoring a configured cluster for faults, such as link failure

The following are the major components of the Sun Fire Link administration software.

- Sun Fire Link Manager
- FM proxies

The Sun™ Management Center (Sun MC) agent and Sun MC console are not described in this article and are not used.

FIGURE 3 illustrates where the software is located and how it communicates.

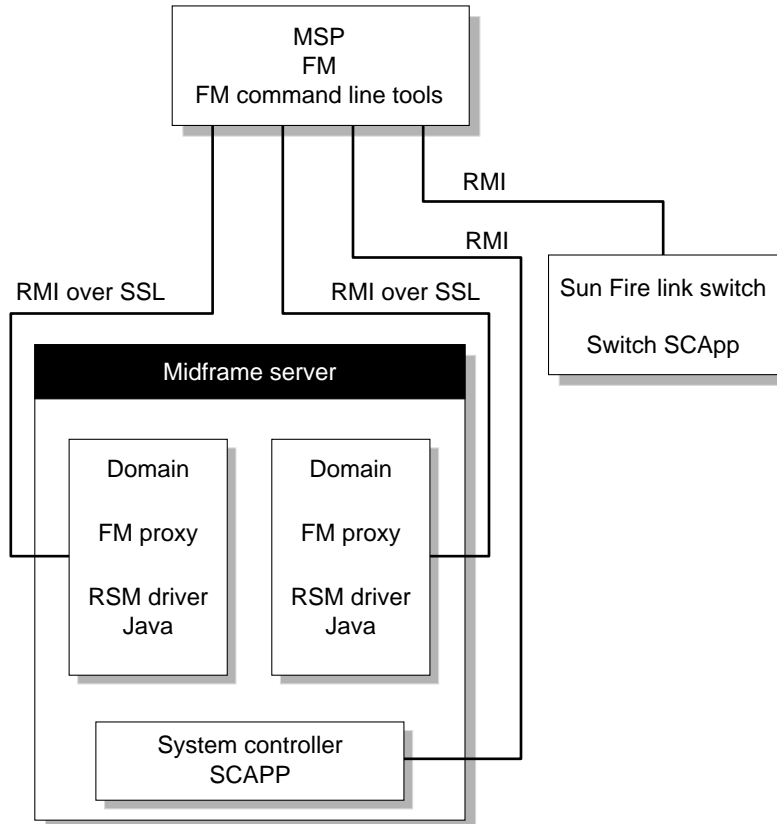


FIGURE 3 Sun Fire Link Administration Software

Sun Fire Link Manager

The Sun Fire Link Manager (FM) is installed on a host that is external to the Fire Link cluster. The FM must be installed on the MSP. The FM is a Java application that communicates to the managed entities (compute nodes and switches) through RMI. It is responsible for managing and configuring the Sun Fire Link fabric. The fabric is a collection of RSM partitions, compute nodes, and switch nodes.

Major functions of the FM are:

- Creating fabrics
- Creating switched and direct connect topologies
- Adding nodes to and removing nodes from a partition
- Modifying the striping level of a partition

Given the requested topology, stripe level, and node membership, the FM computes configuration information for each node. The FM then distributes the configuration information to every node of the fabric. This configuration information contains items such as striping level (how many links between each node) and the cluster ID. The configuration data is stored in the FM configuration file. The FM configuration file is the persistent form of the FM. If the FM is stopped and restarted the FM configuration file restores the memory resident data structures. This file contains the nodes in the fabrics, which partitions exist and what links are used in what partitions. Another file that the FM manages is the password file. This file contains password information for the domains, switch SCs, and midframe SCs. This data is very sensitive and should always be guarded. The FM has a set of command line tools that are used in the example in this article. These tools allow you to access the FM functionally.

This article discusses how to use role-based access control (RBAC) to control access to these files. These files should never be copied to an insecure location. It is important to continue to treat these files as sensitive data when they are backed up. The FM exposes its functionality through an RMI interface. This RMI interface is called by the FM command line tools. This interface is the access point that Sun MC uses to execute the FM functionality. Access to the RMI interface is protected by a community password. The FM RMI interface refuses connections from systems other than MSP. All FM command line interface (CLI) tools must be executed on the MSP.

FIGURE 4 is a diagram of the fabric, partition, and nodes.

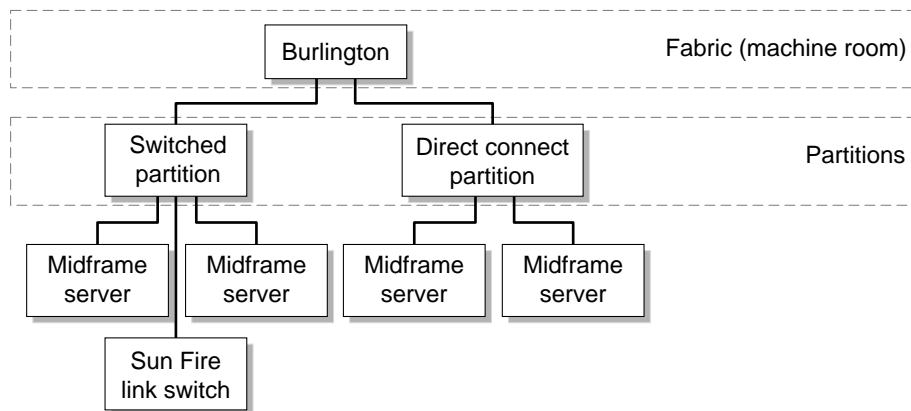


FIGURE 4 Fabric, Partition, and Node Diagram

FM Proxy

The FM proxy is a Solaris OE process that runs on every midframe node that participates in a Sun Fire Link cluster. It is a Java daemon that runs continually. The FM proxy provides configuration and data retrieval methods for the FM. The FM proxy downloads and fetches configuration information from the RSM driver. The FM proxy executes the FM proxy methods through a private Java RMI interface. The RSM driver then communicates this information to the Sun Fire Link hardware. The FM proxy's RMI traffic is validated and protected by using SSL because it is running on a domain. After following the procedures in this article, SSL will be used between the FM proxy and the FM. SSL will restrict access and secure communication to the FM proxy.

SSL Primer

For this article it is helpful to understand the background of public key cryptography. Data that travels across a network can easily be snooped, or its point of origin can be faked. When the data contains sensitive information, such as domain passwords, steps must be taken to protect the data from unauthorized parties. The Secure Sockets Layer (SSL) and Transport Layer Security (TLS) protocols were designed to help protect privacy and integrity of data while it is transferred.

SSL must be installed and enabled to configure secure Sun Fire Link software. We provide the instructions for how to configure Java™ Secure Socket Extension (JSSE). This is a reference implementation for a Java version of the SSL and TLS protocols and includes functionality for message integrity and data encryption.

Encryption is the process of using a complex algorithm to convert cleartext to an encoded message ciphertext. The ciphertext is meaningless without the algorithm and key. Decrypting is the inverse of encrypting. It produces a cleartext message from a ciphertext message. There are two types of methods of encryption; public/private and secret key cryptography. Public key cryptography uses an encryption algorithm in which two keys are produced. One key is made public while the other key is kept private. The public key and private key are cryptographic inverses; what one key encrypts only the other key can decrypt. It is also computationally infeasible to compute the private key from the public key. Secret Key Cryptograph uses an encryption algorithm in which the same key is used both to encrypt and decrypt the data. A digital signature is a computed value which can be used to validate the origin and integrity of a message. The originator of a message uses a key to compute a unique signature for the message. Both the message and the digital signature is sent to the recipient. The recipient uses a verification key to verify that the message and the signature match. If the signature and the message are verified then we can be confident about the source and the content of the message. SSL uses public key

cryptograph to provide authentication, secret key cryptograph, and digital signatures to provide for privacy and data integrity. To use SSL you must create a *keystore*. A keystore is a database of public and private keys. Typically there are two categories of keystore information: key entries and trusted certificate entries. A key entry consist of an identity and its private key. The private key is very sensitive data and must be protected. In contrast, a trusted certificate entry contains only a public key in addition to the entity's identity. Therefore a trusted certificate entry cannot be used where a private key is required.

Security Issues

In the Sun Fire Link system there are a few security risks:

- Insecure system controllers.
- Network traffic between the FM and the system controllers
- Network traffic between the FM and the FM proxy
- Cleartext password information
- Rogue FM clients

The procedures in this article resolve these issues by deploying the MSP and using SSL.

The insecure system controllers are placed on a private management network. Access is restricted by the MSP. Since these system controllers are protected by the MSP, they are no longer a security risk.

The network traffic between the FM and the system controllers is only on the private management network behind the MSP. This traffic is protected by the MSP.

The network traffic between the FM and FM Proxy will be protected by SSL. SSL is used to authenticate the endpoints and protect against privacy attacks. SSL is a widely used and understood security protocol. The cleartext password information is controlled by using SSL and the FM password.

The FM stores passwords for both system controllers and domains. When SSL is deployed the domain passwords are no longer needed; the authentication is by public key cryptography. When you install the FM password, the cleartext password is encrypted in a file.

Another potential issue is the RMI access to the FM. This access is restricted to the local host with root access because the machine on which the FM is running is the MSP. The FM security is protected by the security of the MSP.

Securing the Midframe Sun Fire Link Cluster

To configure the Sun Fire Link, the first step is to build a private subnet and configure the midframe service processor (MSP). This step is detailed in the “Securing the Sun Fire Midframe System Controller” blueprint article. All steps in that article must be followed to secure the midframe SC. For direct connect topologies (Sun Fire Link clusters that do not use a switch) you can ignore the Sun Fire Link switch-specific instructions. The MSP, private SC, and switch SC subnet provide a containment of the insecure elements. This containment is critical to protect these insecure components.

FIGURE 5 shows how the MSP, the switch SC, and midframe SC interrelate.

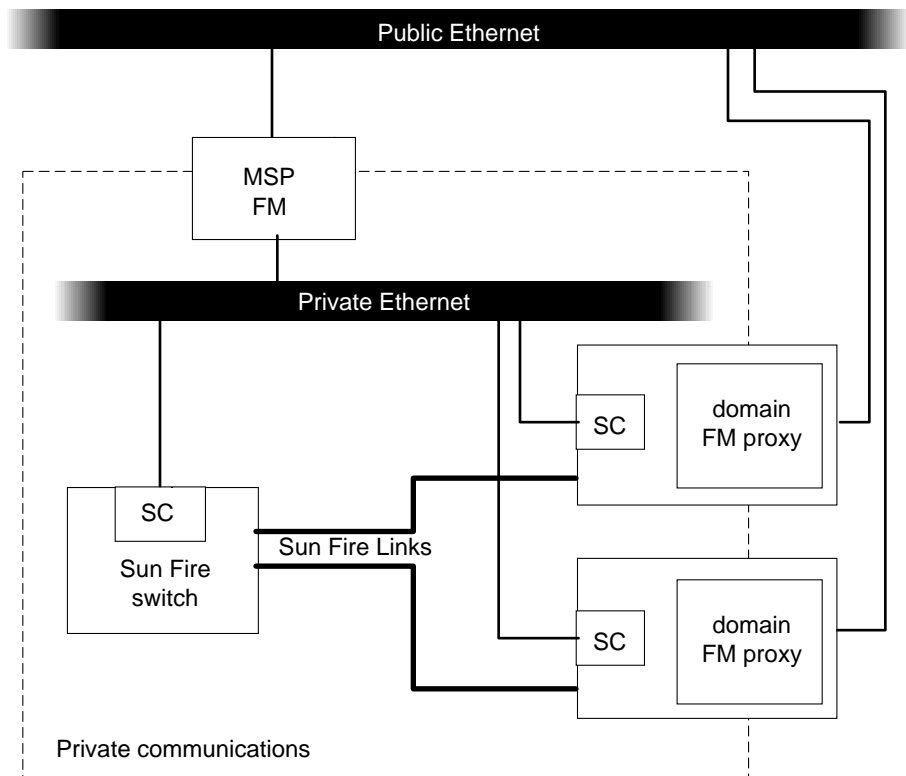


FIGURE 5 Sample SC Network Topology

After the MSP and the private subnet are set up, the following procedures are required:

1. Creating the keystores
2. Securing the FM proxies
3. Securing the FM
4. Securing the switch
5. Securing the RSM driver
6. Configuring the fabric

Creating the Keystores

This article uses `keytool` to administer the keystores. Keytool is a key and certificate management utility. You can choose other tools to create the keystores and keys, but the tool must create a single keystore that contains public and private keys.

The `keytool` command is part of the Java 1.2 environment. For additional information on `keytool` go to: <http://java.sun.com/products/jdk/1.2/docs/tooldocs/solaris/keytool.html>

You must create two keystores. The two keystores are `FMKeyStore` and `ProxyKeyStore`. The `FMKeyStore` is the keystore that is used by the FM. This keystore contains the Private key that the FM uses. The `ProxyKeyStore` is the keystore used by the FM Proxy. This contains the private key that the FM proxy uses and the public key for the FM.

FIGURE 6 shows where the certificate (public key) and private key will be distributed.

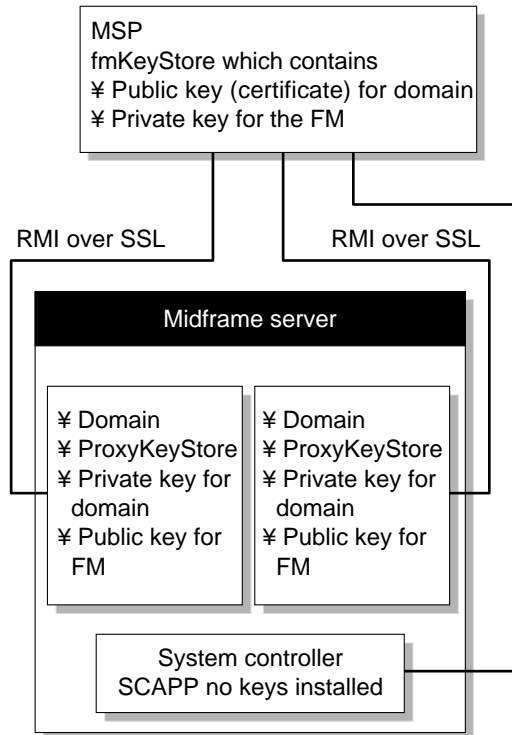


FIGURE 6 Public and Private Key Locations

▼ To Create the Keystore and Generate the Key Pair

1. Type the following command on the MSP in a safe private directory. Press return for the **fmKey** password.

```
# /usr/bin/keytool -genkey -dname "cn=JohnSmith, ou=Purchasing,
o=ABCSystems s="CA c=US" -alias fmKey -keystore fmKeyStore -validity 180
Enter keystore password: YourPassword
Enter key password for <fmKey>
(RETURN if same as keystore password):
```

Note – This command must be typed on the MSP as a single line. Multiple lines are used in the examples for legibility purposes.)

All of the italicized items represent values. X.500 Distinguished Names are used to identify entities, such as those that are named by the subject and issuer.

TABLE 1 lists the keywords, X.500 distinguished names, and examples.

TABLE 1 Keywords, X.500 Distinguished Names, and Examples

KeyWord	X.500 Distinguished Names	Example
CN	Common name (Name of person)	John Smith
OU	Organization unit (department)	Purchasing
O	Organization name (company)	ABCSystems
l=	Locality name (city name)	Burlington
S	State name (state)	MA
C	Country	US

This command creates the keystore named `fmKeyStore` in the working directory, and assigns it the password *YourPassword*. Substitute a password for *YourPassword* which must be kept secret. This password is referred to as the keystore password. You must remember this password because it is used in other steps. Keytool generates a public/private key pair for the entity named `fmKey`. The generated key expires in 180 days. Every 180 days you must generate a new private/public key. You must then replace the private key on the proxy and public key on the FM.

- 2. Verify that the keystore was created correctly and the key entry is contained in the keystore. Substitute *YourPassword* for the keystore password you specified in the previous command.**

```
#keytool -list -keystore fmKeyStore
Enter keystore password: YourPassword

Keystore type: jks
Keystore provider: SUN

Your keystore contains 1 entry:

fmkey, Fri Apr 12 12:11:44 EDT 2002, keyEntry,
Certificate fingerprint (MD5):
F1:11:FF:90:B0:D8:C6:DE:23:CE:36:3F:81:B2:30:36
```

Note – The message digest 5 (MDF) fingerprint and date will be different than that shown in the example. MD5 is an algorithm for computing digital signatures.

3. **Generate the keystore for the FM Proxy node. Select a password for this keystore that is different than the fmKeyStore password.**

```
# /usr/bin/keytool -genkey -dname "cn=JohnSmith, ou=Purchasing, o=ABCSystems,
s="MA, c=US" -alias proxyKey -keystore proxyKeyStore -validity 180
Enter keystore password: YourPassword
Enter key password for <fmKey>
(RETURN if same as keystore password):
```

```
# keytool -list -keystore proxyKeyStore
Enter keystore password: YourPassword

Keystore type: jks
Keystore provider: SUN

Your keystore contains 1 entry:

proxykey, Fri Apr 12 12:15:44 EDT 2002, keyEntry,
Certificate fingerprint (MD5):
F2:11:FF:90:B0:D8:C6:DE:23:CE:36:3F:81:B2:30:36
```

The date and fingerprint of the key will be different then it was before.

4. **Using your favorite text editor, generate a file that contains the certificate (public key) for the fmKey pair. This public key will be installed in the proxyKeyStore.**

```
# /usr/bin/keytool -export -alias fmKey -keystore fmKeyStore > fmCert
Enter keystore password: YourPassword
```

5. Verify that the public key was created. The strings for the CommonName and so forth are what you entered in the keytool -genkey step (Step1).

```
# /usr/bin/keytool -printcert -file fmCert
Owner: CN=JohnSmith, OU=Purchasing, O=ABCSystems, ST="MA c=US"
Issuer: CN=CommonName, -OU=OrganizationName, O=CompanyName, ST="CA
c=US"
Serial number: 3cb70740
Valid from: Fri Apr 12 12:11:44 EDT 2002 until: Wed Oct 09 12:11:44
EDT 2002
Certificate s:
MD5: F1:11:FF:90:B0:D8:C6:DE:23:CE:36:3F:81:B2:30:36
SHA1: 6C:76:5D:E9:64:84:08:E2:95:0B:64:95:70:6D:3F:E9:F5:D5:87:7E
```

The fingerprints and date should be the same as those displayed when the fmKeyStore was listed in Step 3.

6. Generate a file that contains the certificate (public key) for the proxyKey. This public key will be installed in the keystore fmKeyStore.

```
# /usr/bin/keytool -export -alias proxyKey -keystore proxyKeyStore >
proxyCert
Enter keystore password: YourPassword
```

7. Verify that the public key was created. The values for the CommonName and so forth are what you entered in the keytool -genkey step (Step 1).

```
# /usr/bin/keytool -printcert -file proxyCert
Owner: CN=JohnSmith, OU=Purchasing, O=ABCSystems, ST="CA c=US"
Issuer: CN=JohnSmith, OU=Purchasing, O=ACBCSystems, ST="CA c=US"
Serial number: 3d98d307
Valid from: Fri Apr 12 12:15:44 EDT 2002 until: Wed Oct 09 12:11:44
EDT 2002
Certificate fingerprints:
MD5: F2:11:FF:90:B0:D8:C6:DE:23:CE:36:3F:81:B2:30:36
SHA1: 6C:76:5D:E9:64:84:08:E2:95:0B:64:95:70:6D:3F:E9:F5:D5:87:7E
```

8. Import the FM's public key into the proxyKeyStore.

```
# /usr/bin/keytool -import -alias fmKey -file fmCert -keystore proxyKeyStore
Enter keystore password: YourPassword
Owner: CN=JohnSmith, OU=Purchasing, O=ABCSystems, ST="CA c=US"
Serial number: 3cb70740
Issuer: CN=JohnSmith, OU=Purchasing, O=ACBCSystems, ST="CA c=US"
Valid from: Fri Apr 12 12:11:44 EDT 2002 until: Wed Oct 09 12:11:44
EDT 2002
Certificate fingerprints:
MD5: F1:11:FF:90:B0:D8:C6:DE:23:CE:36:3F:81:B2:30:36
SHA1:6C:76:5D:E9:64:84:08:E2:95:0B:64:95:70:6D:3F:E9:F5:D5:87:7

Trust this certificate? [no]: yes

Certificate was added to keystore
```

9. Validate that the proxyKeyStore contains the private key for the proxyKey and public key for the FM.

```
# /usr/bin/keytool -list -keystore proxyKeyStore
keystore password: YourPassword
Keystore type: jks
Keystore provider: SUN

Your keystore contains 2 entries:
proxyKey, Fri Apr 12 12:15:44 EDT 2002
Certificate fingerprints(MD5)
F2:11:FF:90:B0:D8:C6:DE:23:CE:36:3F:81:B2:30:36
fmkey, Fri Apr 12 12:11:44 EDT 2002 trustedCertEntry,
Certificate fingerprint (MD5):
F1:11:FF:90:B0:D8:C6:DE:23:CE:36:3F:81:B2:30:36
```

10. Import the proxy public key into the fmKeyStore.

```
# /usr/bin/keytool -import -alias fmKey -file proxyCert -keystore fmKeyStore

Enter keystore password: YourPassword
Owner: CN=JohnSmith, OU=Purchasing, O=ABCSystems, ST="CA c=US"
Serial number: 3d98d307
Issuer: CN=JohnSmith, OU=Purchasing, O=ACBCSystems, ST="CA c=US"
Valid from: Fri Apr 12 12:15:44 EDT 2002 until: Wed Oct 09 12:11:44
EDT 2002
Certificate fingerprints:
MD5: F2:11:FF:90:B0:D8:C6:DE:23:CE:36:3F:81:B2:30:36
SHA1:6C:76:5D:E9:64:84:08:E2:95:0B:64:95:70:6D:3F:E9:F5:D5:87:7

Trust this certificate? [no]: yes

Certificate was added to keystore
```

11. Validate that the fmKeyStore contains the private key for the FM and public key for the Proxy.

```
# /usr/bin/keytool -list -keystore fmKeyStore

keystore password: YourPassword

Keystore type: jks

Keystore provider: SUN

Your keystore contains 2 entries:
fmKey, Fri Apr 12 12:11:44 EDT 2002
Certificate fingerprints (MD5):
F1:11:FF:90:B0:D8:C6:DE:23:CE:36:3F:81:B2:30:36
proxyKey, Fri Apr 12 12:15:44 EDT 2002 trustedCertEntry,
Certificate fingerprint (MD5):
F2:11:FF:90:B0:D8:C6:DE:23:CE:36:3F:81:B2:30:36
```

These two keystores will be used in other steps. If desired, you could generate a unique public/private key for each domain. You would then install each public key in the fmKeyStore. This article explains how to create a single keystore, which is then copied to each domain.

Securing the FM Proxy

These steps must be performed on every Solaris domain where the FM Proxy is running. Failure to do so will leave your system susceptible to attack. The required procedures are:

1. Configuring the proxy to use SSL
2. Installing the keystore
3. Creating the file `ssl.info`
4. Stopping and restarting the proxy.

▼ To Configure the Proxy to Use SSL

1. Download JSSE 1.0.2.

You can download the file anywhere on your local disk. Note that JSSE 1.0.2 requires that you have Java 1.2.1 or greater already installed. you can download the file from [HTTP://java.sun.com/products](http://java.sun.com/products). You should get the file `jsse-1_0_2-do.zip`.

2. Uncompress and extract the downloaded file. The following command unzips the download.

```
# unzip jsse-1_0_2-do.zip
Archive:  jsse-1_0_2-do.zip
  inflating: jsse1.0.2/BUGS.html
  inflating: jsse1.0.2/CHANGES.txt
  inflating: jsse1.0.2/COPYRIGHT.ht
... Many more files are listed
```

This creates a directory named `jsse1.0.2`, with two subdirectories named `doc` and `lib`.

3. Install the JSSE .jar files. in your extension directory. The JSSE Lib subdirectory contains the extension files `jsse.jar`, `jcert.jar` and `jnet.jar`. Determine which JVM the proxy is using by typing the following command.

```
# grep -v "#" /opt/SUNWwrsmp/jre_home.cfg
JAVA_BIN=/usr/java1.2/jre/bin/
```

The result of this command is that the directory contains the version of Java that the proxy is using. To get the Java *JRE* directory, remove the `/bin`. In the preceding example the directory is `/usr/java1.2/jre`. For this article substitute your Java directory for the string *\$JRE* in the commands.

4. Copy the following files to the *\$JRE/lib/ext* (installed extension) directory.

```
# cp lib/jsse.jar $JRE/lib/ext/jsse.jar
# cp lib/jcert.jar $JRE/lib/ext/jcert.jar
# cp lib/jnet.jar $JRE/lib/ext/jnet.jar
```

5. Verify that the files exist and they are owned by root.

```
# ls -l $JRE/lib/ext
-rw-r--r--  1 root      root 7637 Feb 20 10:17 jcert.jar
-rw-r--r--  1 root      root 3098 Feb 20 10:17 jnet.jar
-rw-r--r--  1 root      root 463471 Feb 20 10:17 jsse.jar
```

6. Register Sun JSSE provider.

JSSE comes standard with a cryptographic service provider, or provider for short, named SunJSSE. Although the SunJSSE provider must be configured explicitly, this provider should be registered statically. Static registration is done by editing the security Properties file located at *\$JRE/lib/security/java.security*. One of the types of properties contained in the *java.security* files is of the following form:\.

```
security.provider.n=providerClassName
```

This line declares security provider and its preference. You should add a new line to that section and install the standard provider shipped with the JRE. The entries should look like the following:

```
security.provider.1=sun.security.provider.Sun
security.provider.2=com.sun.net.ssl.internal.ssl.Provider
```

▼ To Install the proxyKeyStore

You must now distribute the keystore that was created previously, `proxyKeyStore`, to the RSM domain.

1. To move the key, use the secure copy `scp` command to copy the file onto your system. Execute the following command, substituting the machine name of `$MSP` for the server that is acting as your `$MSP`.

```
# scp $MSP:/privatedir/proxyKeyStore /opt/SUNWwrsmp/  
proxyKeyStore
```

2. Verify that this file is root read only. It is important that the private key is protected. The size of your file might be different.

```
# ls -l /opt/SUNWwrsmp/proxyKeyStore  
-rw----- 1 root root 7637 Feb 20 10:17
```

The proxy has a `java.policy` file. It is located in `/opt/SUNWwrsmp/java.policy`. This file should be edited so that the proxy has access to the file `proxyKeyStore`.

3. Add the following line above the `};` in the file `/opt/SUNWwrsmp:`

```
# vi /opt/SUNWwrsmp/java.policy  
add the following line  
permission java.io.FilePermission "/opt/SUNWwrsmp/proxyKeyStore",  
"read";
```

▼ To Create the File `ssl.info`

This file is the configuration data that the FM proxies needs to use the private key.

1. Create a file called `ssl.info` containing the following lines, but substitute the `keystore password` for `YourPassword`.

```
KEY_STORE_PASSPHRASE=YourPassword  
KEY_STORE_LOCATION=/opt/SUNWwrsmp/proxyKeyStore
```

This information is sensitive so verify that the file is root read only by entering the following command to verify the file access.

```
# ls -l /opt/SUNWwrsmp/ssl.info  
-rw----- 1 root root 7637 Feb 20 10:17
```

▼ To Stop and Restart the Proxy

1. To have the proxy use SSL and the enhanced security settings, you must restart the proxy. The following command will stop the proxy.

```
# /etc/init.d/wrsm_proxy stop
```

2. To start the proxy type the following command.

```
# /etc/init.d/wrsm_proxy start
```

3. Verify that the proxy started with enhanced security. This example only lists the critical lines of output. For readability, the rest of the lines are ignored.

```
# tail /var/opt/SUNWwrsmp/log/wrmsp_nohup.out
...
using SSL.
...
RP Bound To Registry
```

4. Repeat all of the preceding steps for each domain.

Securing the FM

The FM should be installed on the MSP. For the Sun Fire Link to be secure the appropriate action in the “MSP Security” blueprints must have been followed. If you have not done so, follow those instructions before you proceed. After the FM is installed correctly the following procedures are required:

1. Setting up the FM password file
2. Using RBAC on the FM
3. Setting up the FM’s JVM to use SSL
4. Setting up the keystore for the FM
5. Securing the account `sfluser`
6. Copying `wcrmp.jar`

Setting Up the FM Password File

The FM can delete and reconfigure RSM clusters, so access to the FM RMI interface must be protected. A shared secret controls access to the FM. This shared secret is known to valid FM client programs and the FM. The shared secret is contained in a file whose access is restricted to root. If validation of the shared secret fails, the FM rejects the client request. The file must be located in the FM installed directory. This article assumes that you installed the FM using the default data directory, `/var/opt/SUNWwcfm`. If you did not use the default directory, you must modify the commands to reflect the directory in which the FM was installed.

▼ To Set Up the FM Password File

1. Execute the following commands.

```
# echo "PASSWORD=YourPassword" >> /var/opt/SUNWwcfm/config/security.info
# echo "KEY_STORE_PASSPHRASE=KeyStorePassword" >> /var/opt/SUNWwcfm/config/security.info
# echo "KEY_STORE_LOCATION=/opt/SUNWwcfm/classes/fmKeyStore" >> /var/opt/SUNWwcfm/config/security.info
```

These commands create a file and set *YourPassword* as the password that the FM uses. The *keyStorePassword* should be the password that you specified for the FM keystore. These commands also specify the keystore location.

2. Check that the file was created correctly.

```
# more /var/opt/SUNWwcfm/config/security.info
PASSWORD=yourpassword
KEY_STORE_PASSPHRASE=KeyStorePassword
KEY_STORE_LOCATION=/opt/SUNWwcfm/classes/fmKeyStore
```

3. Set the access on the file. Execute the following command to only allow access by root.

```
# chmod 400 /var/opt/SUNWwcfm/security.info
```

4. Verify that the access is restricted on the file.

```
# ls -lst /var/opt/SUNWwcfm/security.info
-r----- 1 root other 21 Apr 10 11:58 /opt/SUNWwcfm/classes/security.info
```

The password file is now set up.

Using RBAC on the FM

RBAC is an alternative to the all-or-nothing superuser model. RBAC is in keeping with the security principle of least privilege, which states that no user should be given more privilege than required for performing the job. RBAC enables roles for assignment to specific individuals according to their job needs. This enables a variety of security policies. This section provides the steps to create a special purpose role 'fmadmin' and then configure the FM commands so that this role can access it. RBAC is available on the Solaris 8 OE and above. If the MSP is running an earlier version of the Solaris OE, RBAC should not be used.

Note – The “Securing the Solaris OE: Updated for Solaris 9,” blueprint article has a section on RBAC and Solaris OE in addition to examples to which you can refer.

RBAC applies the following concepts:

- **Roles.** These are persona that a specific user can take on. For example, users can be assigned the role fmadmin. They can then `su` to that role and execute those commands. They cannot execute other commands.
- **Authorization.** A permission can be assigned to a role to perform a specific class of actions.
- **Profile.** A package of rights can be assigned to a role or a user.

Configuring the FM commands to use RBAC requires the following:

- 1. Creating the role**
- 2. Creating the profile**
- 3. Adding the profile to the role**
- 4. Verifying that the changes have been made in the `/etc/user_attr` file**
- 5. Creating the user adding new role**
- 6. Assigning commands to the profile**

▼ To Configure the FM Commands to Use RBAC

1. Create the role as follows.

This example uses a role named *fmadmin* and a sample UID. You must use a UID that is valid for your installation.

```
# roleadd -u 123456 -g 101 -d /export/home/fmadmin -m fmadmin
6 blocks
# passwd fmadmin
New password:
Re-enter new password:
passwd (SYSTEM): passwd successfully changed for fmadmin
# pwconv
```

2. Verify that the role has been added correctly.

```
# grep -i fmadmin /etc/passwd
fmadmin:x:123456:101::/export/home/fmadmin:/bin/pfsh
```

The number UID of 123456 will be different on your system. Note that the shell is *pfsh*, which is the profile shell.

3. Create a profile with a comment for the *fmadmin*.

```
# echo "fm-profile:::Ability to issue fm commands:" >>
/etc/security/prof_attr
```

4. Verify that the profile was added.

```
# grep fm-profile /etc/security/prof_attr
fm-profile:::Ability to issue fm commands:
```

5. Add the profile to the *fmadmin* role. The *All* allows the *fmadmin* role to execute normal commands.

```
# rolemod -P fm-profile,All fmadmin
```

6. Verify the changes.

```
# grep fm-profile /etc/user_attr
fmadmin::::type=role;profiles=fm-profile,All
# profiles fmadmin
fm-profile
All
Basic Solaris User
```

7. Assign the role to a new or existing usermod -R account.

You can use any normal administrator account for this step. This procedure adds the role to the account `sfluser`. `sfluser` was created when you followed the normal FM installation notes.

```
# usermod -R fmadmin sfluser
```

8. Verify that the roles are assigned to the user

```
# roles sfluser
fmadmin
```

9. Assign commands to the profile. These commands assume that the FM was installed in

`/opt/SUNWwcfm`.

```

# echo "fm-profile:suser:cmd::/opt/SUNWwcfm/bin/
listfabrics:uid=0" >> /etc/security/exec_attr
# echo "fm-profile:suser:cmd::/opt/SUNWwcfm/bin/
createfabric:uid=0" >> /etc/security/exec_attr
# echo "fm-profile:suser:cmd::/opt/SUNWwcfm/bin/
deletefabric:uid=0" >> /etc/security/exec_attr
# echo "fm-profile:suser:cmd::/opt/SUNWwcfm/bin/
killfabrics:uid=0" >> /etc/security/exec_attr
# echo "fm-profile:suser:cmd::/opt/SUNWwcfm/bin/
startfabric:uid=0" >> /etc/security/exec_attr
# echo "fm-profile:suser:cmd::/opt/SUNWwcfm/bin/
stopfabric:uid=0" >> /etc/security/exec_attr
# echo "fm-profile:suser:cmd::/opt/SUNWwcfm/bin/wcfmconf:uid=0"
>> /etc/security/exec_attr
# echo "fm-profile:suser:cmd::/opt/SUNWwcfm/bin/wcfmstat:uid=0"
>> /etc/security/exec_attr
# echo "fm-profile:suser:cmd::/opt/SUNWwcfm/bin/wcfmver:uid=0"
>> /etc/security/exec_attr

```

▼ To Set Up the FM's JVM to Use SSL

These instructions are very similar to those used when SSL was enabled in the proxy's JVM.

1. Obtain and unzip JSSE 1.0.3.

You can download the file anywhere on your local disk. Note that JSSE 1.0.3 requires that you have Java 1.2.1 or greater already installed. You can download the file from [HTTP://java.sun.com/products](http://java.sun.com/products). You should get the file `jsse-1_0_3-do.zip`.

2. Uncompress and extract the downloaded file.

This will create a directory named `jsse1.0.3`, with two subdirectories named `doc` and `lib`. The following command unzips the download.

```

# unzip jsse-1_0_3-do.zip
Archive:  jsse-1_0_3-do.zip
  inflating:  jsse1.0.3/BUGS.html
  inflating:  jsse1.0.3/CHANGES.txt
  inflating:  jsse1.0.3/COPYRIGHT.ht
... Many more files are listed

```

3. Install the JSSE .jar files in your extension directory.

The `JSSE lib` subdirectory contains the extension files `jsse.jar`, `jcert.jar`, and `jnet.jar`. Determine the Java VM that the FM is using and type the following command. If the FM is installed in a different location you must modify the command.

```
# grep -v "#" /opt/SUNWwcfm/config/jre_home.cfg
JAVA_BIN=/usr/java1.2/jre/bin/
```

The line that is output is the Java bin directory the FM uses. To get the Java `$JRE` directory remove the `/bin`. In the preceding example, the directory is `/usr/java1.2/jre`. For this article substitute your Java directory for the string `$JRE` in the commands.

4. Copy these files in the `$JRE/lib/ext` (installed extension) directory.

```
# cp lib/jsse.jar $JRE/lib/ext/jsse.jar
# cp lib/jcert.jar $JRE/lib/ext/jcert.jar
# cp lib/jnet.jar $JRE/lib/ext/jnet.jar
```

5. Verify that the files exist and that they are owned by root.

```
ls -l $JRE/lib/ext
-rw-r--r--  1 root      root 7637 Feb 20 10:17 jcert.jar
-rw-r--r--  1 root      root 3098 Feb 20 10:17 jnet.jar
-rw-r--r--  1 root      root 463471 Feb 20 10:17 jsse.jar
```

6. Register the Sun JSSE provider.

JSSE comes standard with a cryptographic service provider named *SunJSSE*. Although the SunJSSE provider must be configured explicitly, it should be registered statically. Static registration is done by editing the security properties file which is located at:

```
$JRE/lib/security/java.security
```

One of the types of properties contained in the `java.security` files is of the following form:

```
security.provider.n=providerClassName
```

This line declares security provider and its preference. You should add a new line to that section and install the standard provider shipped with the JRE, the entries should now look like:

```
security.provider.1=sun.security.provider.Sun
security.provider.2=com.sun.net.ssl.internal.ssl.Provider
```

▼ To Set Up the Keystore for the FM

The FM keystore must be copied into the location specified earlier.

1. **Copy the keystore to the location** `/opt/SUNWwcfm/classes/fmKeyStore` **and verify that it is owned by root and read-only to root.**
2. **Modify the FM policy file so it can access the keyStore. It is located in** `/var/opt/SUNWwcfm/config/YourFabricName/cfg/YourFabricName.policy`, **where** *YourFabricName* **is the fabric name that you supplied to the** `createfabric` **command.**
3. **Edit this file so that the proxy has access to the file** `fmKeyStore`. **Add the following line above the** `};` **in the file** `/var/opt/SUNWwcfm/config/YourFabricName/cfg/YourFabricName.policy`.

```
# vi /var/opt/SUNWwcfm/config/YourFabricName/cfg/
YourFabricName.policy
add the following line
permission java.io.FilePermission "/opt/SUNWwcfm/classes/
fmKeyStore", "read";
```

The string *YourFabricName* should be equal to the fabric name you created. If you have not created a fabric you should create a fabric using the following:

```
# /opt/SUNWwcfm/bin/createFabric your_fabric_name
```

Securing the Fire Link Switch

Placing the switch behind the private network protects the switch. Providing additional security requires the following:

1. Setting the switch password

2. Setting the RMI password
3. Using the MSP as the flash server
4. Installing the write protect jumper

▼ To Set the Switch Password

The password for the switch must be set. The Switch SC interactively allows the user to set a password for console access. The user must provide the old password (if applicable) to be able to set a new password. The password is used to get to the CLI on the console. The switch uses a shared account. The password controls access to this account. The password will be prompted for when the switch is first powered on, after a reboot, or if the logout command has been issued. It can also be prompted for if the console timeout has been set and the console has timed out and closed. Setting a blank password clears the current password.

- **Set the password on the switch.**

```
# password
Enter Password>OldPassword
Enter New Password>NewPassword
password was changed.
```

If you forget the password, the command `resetpassword` will reset the password. To use this command you need physical access to the switch to move a jumper. For instructions, consult the *Sun Fire Link Installation* manual.

There is also a command called `logout` that allows the user to log out

▼ To Set the RMI Password

The RMI password is a string that is sent when the FM requests an operation of the switch. For example when the FM sends a configuration to the switch this password is sent to authorize the action.

- **Set the RMI password.**

```
# rmi=password
No RMI password is assigned. Do you want to set one? (y/[n]) >y
Enter password>
```

▼ To Use the MSP As the Flashserver

The `flashupdate` feature is used to update the firmware running on the switch. The update is initiated by using the `flashupdate` command on the switch. The source flash image may be on a server. This article refers to performing a `flashupdate` using the MSP as the flash server.

▼ To set the Write Protect Jumper

The switch has a jumper called *flash write enable jumper*. This jumper has two positions, write-protect and write-enabled. The factory setting for this jumper is the write-enabled position. For details, refer to the *Sun Fire Link Installation and Service manual*.

Some organizations may have security policies that require a high degree of protection against the risk of improper access to the real time operating system (RTOS). When such a requirement exists, the write-protect jumper can be used to provide this protection.

Securing the RSM Driver

With the RSM driver all commands that modify the configuration are restricted to the superuser, including items like `initial` and `replace`. All other operations (`dump config`, `info`, `topology`, and so forth) are available to any user.

▼ To Restrict Access to the Information

1. Change the protection on the wci device.

```
# chmod 0700 /devices/wrsm*
```

Ongoing Security Efforts

After these security steps are done, the keys that are used by SSL will have to be replaced before they expire. Repeat the steps in “Creating the Keystores” and replace the keystores on the FM and the FM proxies.

Configuring the Fabric

The following is an example of configuring the fabric SFL with the FM command line tools. The fabric SFL was created in "To Set Up the Keystore for the FM". We will use a switch topology and two compute nodes and four switch nodes. The key step is the creation of the cluster1.xml file. The example file uses password for the string you should use to substitute the correct password. The names of the compute nodes are: Traviata and Lucia. The switch nodes are greatsandy, greatbasin, dryvalley, and saltflats. The following steps and configuration file create a single partition called part1.

FIGURE 7 shows the fabric cabling.

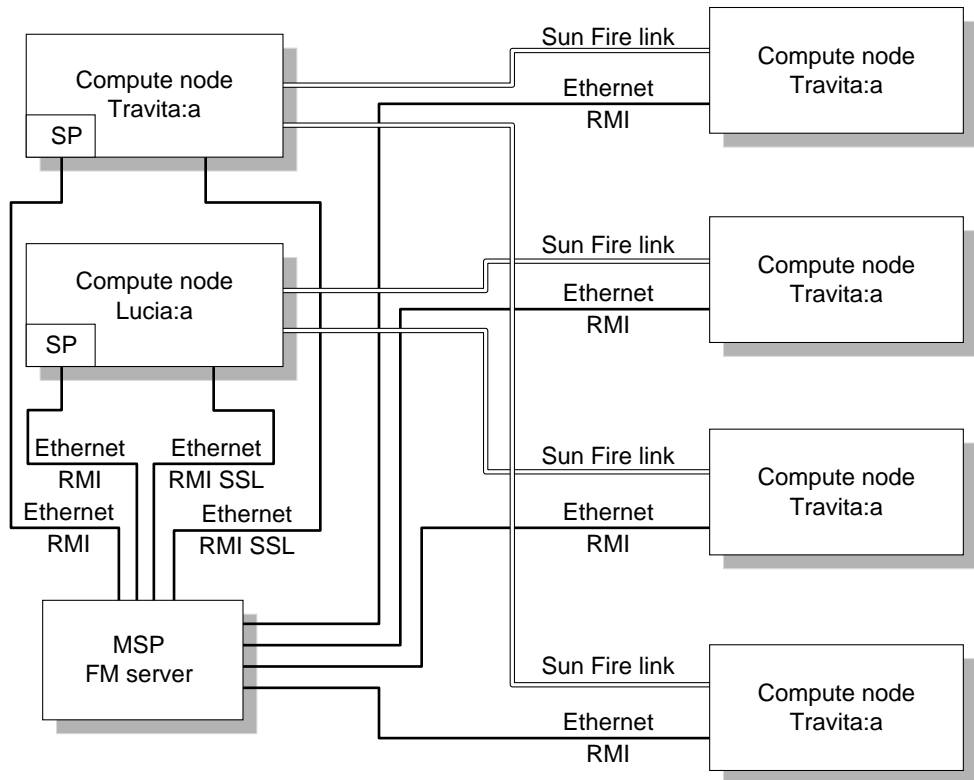


FIGURE 7 Fabric Cabling Diagram

▼ To Configure the Fabric

1. On the MSP, su to the role fmadmin. If you are not using RBAC, ignore this command.

```
# su fmadmin
```

2. Create the cluster1.xml file.

CODE EXAMPLE 1 lists the contents of cluster1.xml file for this topology.

CODE EXAMPLE 1 cluster1.xml File

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE fabric SYSTEM "fabric.dtd">
<fabric>
  <last_date_time>Mon Nov 12 14:21:11 EST 2001</
last_date_time>
  <fname>fmcdc</fname>
  <config_file>
  </config_file>
  <members>
    <switch_node>
      <node>
        <sc_name>greatsandy</sc_name>
        <sc_user_name>sfluser</sc_user_name>
        <sc_password>password</sc_password>
        <chassis_type>WCIX_SWITCH</chassis_type>
      </node>
    </switch_node>
    <switch_node>
      <node>
        <sc_name>greatbasin</sc_name>
        <sc_user_name>sfluser</sc_user_name>
        <sc_password>password</sc_password>
        <chassis_type>WCIX_SWITCH</chassis_type>
      </node>
    </switch_node>
    <switch_node>
      <node>
        <sc_name>dryvalley</sc_name>
        <sc_user_name>sfluser</sc_user_name>
        <sc_password>password</sc_password>
        <chassis_type>WCIX_SWITCH</chassis_type>
      </node>
    </switch_node>
  </members>
</fabric>
```

CODE EXAMPLE 1 cluster1.xml File (Continued)

```
<switch_node>
  <node>
    <sc_name>saltflats</sc_name>
    <sc_user_name>sfluser</sc_user_name>
    <sc_password>password</sc_password>
    <chassis_type>WCIX_SWITCH</chassis_type>
  </node>
</switch_node>
<rsm_node>
  <node>
    <sc_name>traviata</sc_name>
    <sc_user_name>sfluser</sc_user_name>
    <sc_password>password</sc_password>
    <chassis_type>S8</chassis_type>
  </node>
  <domain_name>a</domain_name>
  <hostname>traviata-a</hostname>
  <host_user>sfluser</host_user>
  <host_password>password</host_password>
</rsm_node>
<rsm_node>
  <node>
    <sc_name>brother</sc_name>
    <sc_user_name>sfluser</sc_user_name>
    <sc_password>password</sc_password>
    <chassis_type>S8</chassis_type>
  </node>
  <domain_name>a</domain_name>
  <hostname>brother-a</hostname>
  <host_user>sfluser</host_user>
  <host_password>password</host_password>
</rsm_node>
<rsm_node>
  <node>
    <sc_name>lucia</sc_name>
    <sc_user_name>sfluser</sc_user_name>
    <sc_password>password</sc_password>
    <chassis_type>S8</chassis_type>
  </node>
  <domain_name>a</domain_name>
  <hostname>lucia-a</hostname>
  <host_user>sfluser</host_user>
  <host_password>password</host_password>
</rsm_node>
</members>
</partitions>
```

CODE EXAMPLE 1 cluster1.xml File (Continued)

```
<partition type="RSM" topology="WcixSwitch">
  <pname>part1</pname>
  <stripping_level>4 </stripping_level>
  <partition_members>
    <node_partition_member>
      <sc_name>lucia</sc_name>
      <domain_name>a</domain_name>
    </node_partition_member>
    <node_partition_member>
      <sc_name>traviata</sc_name>
      <domain_name>a</domain_name>
    </node_partition_member>
    <switch_partition_member>
      <sc_name>greatsandy</sc_name>
    </switch_partition_member>
    <switch_partition_member>
      <sc_name>greatbasin</sc_name>
    </switch_partition_member>
    <switch_partition_member>
      <sc_name>dryvalley</sc_name>
    </switch_partition_member>
    <switch_partition_member>
      <sc_name>saltflats</sc_name>
    </switch_partition_member>
  </partition_members>
</partition>
</partitions>
</fabric>
```

Note – The cluster1.xml file contains password strings and should only be placed in secure directories. You must be very careful with this file.

The file cluster1.xml (which was created in the previous step) is used as an argument to the command line tools.

```
# startfabric sfl
# wcfmconf cluster1.xml
Found FM at [//localhost:1099/fmdc]
Configuration file processed successfully.
```

Conclusions

After you follow the steps in this article, the Sun Fire Link can be administered in a secure manner.

About the Author

Joe Higgins has worked in R&D for over 15 years. He is a staff engineer in Enterprise Server Products working on system management and security. Past areas of experience include software architectures, UI Design, and CAD technologies.

His current areas of expertise are network security, system management, and Solaris OE security. Current research activities include topics in graph theory and security of system management software.

Prior to working at Sun, Joe worked at Autodesk, 3D/EYE, Boeing, and Sandia National Laboratories.

Acknowledgements

The author would like to recognize the following individuals for their contributions to this article:

Alex Noordergraaf for answering my questions and making security a reality at ESP. Louise Hua for taking the time to do an excellent QE effort. Steve Bonczar for his support and confidence and Peter Wagener for providing helpful edits and comments.

References

“RBAC in the Solaris™ Operating Environment,” Whitepapers, Sun Microsystems, Inc.

To access this document go to:

<http://www.sun.com/software/whitepapers/wp-rbac/#overview>

“Java™ Secure Socket Extension (JSSE) 1.0.3_01.” Java product document, Sun Microsystems, Inc.

To access this document go to:

<http://java.sun.com/products/jsse/index-103.html>

“keytool - Key and Certificate Management Tool,” Java product document, sun Microsystems, Inc. To access this document go to:

<http://java.sun.com/products/jdk/1.2/docs/tooldocs/solaris/keytool.html>

The following articles are published as part of the Sun security blueprints:

Alex Noordergraaf “Securing Sun Enterprise™ 10000 System Service Processors,” *Sun BluePrints OnLine*, March 2002

Alex Noordergraaf and Dina K. Nimeh “Securing Sun Fire™ 12K and Sun Fire™ 15K System Controllers: updated for SMS 1.2,” *Sun BluePrints OnLine*, July 2002

Joel Weise “Public Key Infrastructure Overview,” *Sun BluePrints OnLine*, August 2001

To access these articles go to: <http://www.sun.com/solutions/blueprints/pubs.html>

Ordering Sun Documents

The SunDocsSM program provides more than 250 manuals from Sun Microsystems, Inc. If you live in the United States, Canada, Europe, or Japan, you can purchase documentation sets or individual manuals through this program.

Accessing Sun Documentation Online

The `docs.sun.com` web site enables you to access Sun technical documentation online. You can browse the `docs.sun.com` archive or search for a specific book title or subject. The URL is `http://docs.sun.com/`

To reference Sun BluePrints OnLine articles, visit the Sun BluePrints OnLine Web site at: `http://www.sun.com/blueprints/online.html`