IPsec—A Secure Deployment Option

Regunathan Rajaiah, Sun Software Services

Sun BluePrints™ OnLine—June 2004

http://www.sun.com/blueprints

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

Part No. 817-7289-10
Revision A, 6/4/04
Edition: June 2004

Please
Recycle

IPsec—A Secure Deployment Option

In today's IT environments, it is critical to protect data traffic between disparate host systems in multitier applications. Information security teams look for security vulnerabilities and try to manage the risk of data tampering, snooping, and eavesdropping. Plain text data flow of critical information like passwords, credit card numbers, and privacy information between systems is highly vulnerable to misuse. The operating system, application code, and enterprise security policies and procedures all have a role in addressing security issues.

IP Security (IPsec) protects IP packets and defines the means with which a packet will be encrypted. IPsec is a valuable option for protecting data in transit. It is an Internet Engineering Task Force (IETF) standard, and many operating systems vendors have incorporated IPsec in their product offerings.

By using a standards-based implementation, interoperability between heterogeneous operating systems can be handled with *lesser* pain. However, IPsec is only one piece in the security solution space. The operating system, application code, and enterprise security policies and procedures all need to address security issues.

This article provides an overview of IPsec and how it is used to secure IP traffic between two systems. Software architects can use this information to deploy multitier applications, based on the Sun™ ONE Portal, Identity, Application, and Web servers. IPsec can also be used in setting up a virtual private network (VPN). The description of that setup is beyond the scope of this article. For more information on tunnel mode, refer to the documents listed in "IPsec Reference" on page 13.

IPsec in Multitiered Deployments

IPsec is an IETF standard for securing IP traffic (IPv4 and IPv6). The following information is taken verbatim from “IP Security (IPsec) Components” (see “IPsec Reference” on page 13). This document captures the relevant information about IPsec.

In typical multitier application production deployments, the presentation tier (for example, web servers) communicates with the business logic tier (for example, application servers, servlets, or JavaServer Pages™ software running in web containers). The business logic tier communicates with the data tier (for example, LDAP and RDBMS). The communications use TCP as the transport layer protocol. For the most part, this traffic is not encrypted, so it is subject to eavesdropping, confidential information snooping, and possible tampering. Security studies show that most incidents are caused by people within the enterprise.

You can use IPsec effectively to provide confidentiality and integrity by using interoperable, cryptography-based security.

This solution is applicable across a wide range of products from different vendors. IPsec operates at the network layer, so it is transparent to application. IPsec performs the encryption and decryption using the configured cryptographic algorithms and keys.

Packet Security

IPsec protects IP packets and defines the means with which a packet will be encrypted. There are two methods, either of which can be used to protect IP communication and works at the network layer of the TCP/IP stack:

- Encapsulating security payload (ESP)
- Authentication header (AH)

In addition, the security association (SA) defines the protection of the message in only one way by using various IPsec mechanisms, and the Internet Key Exchange (IKE) mechanism provides key management and management of various other components.

Encapsulating Security Payload

Encapsulating security payload (ESP) provides authentication, integrity, and confidentiality:

- *Authentication* confirms that the packet received is actually from the sender of the packet. Only the data portion of an IP packet is authenticated. The IP header is not authenticated.
- *Integrity* ensures that data in the packet has not changed in transit.
- *Confidentiality* is achieved by encrypting the message (that is, data).

Only the data is encrypted, which adds to the size of the data. ESP adds its own "header" component between the IP header and other components of the packet before the TCP or UDP header.

ESP uses the following algorithms:

- 3DES in CBC mode
- AES in CBC mode
- Blowfish in CBC mode
- HMAC with MD5
- HMAC with SHA-1

Authentication Header

The authentication header (AH) provides authentication and integrity, as ESP does. However, AH also provides optional security against retransmission of packets by someone else. The AH header is inserted between the IP header and other components of the packet, before the UDP-TCP header. Unlike ESP, some IP header fields might be changed by someone listening to the communication on the Internet or wire. Confidentiality is also not provided.

AH uses authentication algorithms like:

- HMAC-MD5
- HMAC-SHA-1

Security Association

Security association (SA) defines the one-way protection of the message for a specific IPsec mechanisms. There are usually two SAs used in protecting traffic between two hosts. SA can control what to encrypt and what not to encrypt.

SA consists of the keys, algorithms, and lifetime values. The database containing this information is called the security association database (SADB). An SA is identified by the security parameter index (SPI), the destination IP address, and the protocol information (ESP or AH).

In the Solaris Operating System (Solaris OS), the elements that relate to the SADB are:

- `ipseckey`(1M) to configure SADB
- `in.iked`(1M) to manipulate the SADB
- `PF_KEY` socket interface that enables `ipseckey`(1M) and `in.iked`(1M) to perform their tasks

You can implement SA by using the *tunnel* method or the *transport* method. In tunnel mode, every communication between two networks behind a host (for example, a firewall or gateway) is encrypted. In transport mode, communication between two computers (hosts) is secured. Only the data part of the packet is encrypted, apart from the IP header. Therefore, the IP header is not protected. However, in tunnel mode, IPsec encapsulates the complete packet including the IP header.

The IP header has source and destination as the IP address of the gateways or the firewalls that exchange the encrypted information. The hosts behind the gateways communicate in plain messages. The applications or hosts behind the gateway do not need to encrypt or decrypt. IPsec AH and ESP can be used in transport or tunnel mode.

Security Policy Database

The SA(s) used in transmitting and receiving packets is controlled by the security policy. The security policy database (SPD) contains this information. The SPD entries define the source, destination addresses, port numbers, and action to be taken (for instance, allow, permit, drop, or bypass).

In the Solaris OS, the elements that relate to SPD are:

- `ipsecconf`(1M) command to set up a policy for a host
- Per-socket policy
- `{encr,encr_auth,auth}_algs` keywords to `ifconfig`(1M)

Internet Key Exchange

IPsec uses the Internet Key Exchange (IKE) protocol to automate the SA setup and to exchange keys between the two ends (that is, the hosts and gateway). This ensures that, by using keys, the sender and receiver only get the cipher text. IKE can also recreate the keys and exchange again to make sure that even if someone breaks the keys, the keys are already recreated before they are broken again. IKE basically manages this process of updating and recreating the keys.

In IKE, the SAs are set up first. Then, the traffic can be secured. The two hosts or gateways finalize on the encryption and authentication mechanisms. The two gateways then authenticate each other by a mechanism already known to each other. The Diffie-Hellman (DH) public-key algorithm then generates the shared master key. The same master key is then used to define the IPsec keys for SAs.

The two gateways decide on the encryption and authentication mechanism to be used in SAs. The master key is used to get the IPsec keys for the SA. Hence, secure communication is established.

The same master key *can* be used for the IPsec keys for SAs, *or* a new DH can be used. If the new Diffie-Hellman is used, the property of perfect forward secrecy (PFS) is preserved.

For detailed information on setting up IKE with pre-shared keys and public keys, refer to the *IPsec and IKE Administration Guide* at:

`http://docs.sun.com/db/doc/816-7264?q=IPSEC`

Trade-Offs

The nature and value of business data drive the need for securing data traffic inside a corporate network. Some examples of drivers for enhanced enterprise security policies include legal mandates, a need to protect against unauthorized disclosures, and demands or requirements of application owners and users.

To secure data in transit, vendors provide proprietary protocols and tools. For example, `sqlnet` can be configured with an encryption option to protect data flow between an ORACLE® client and server. Application security methods, though not public and possibly not based on industry standards, might still offer the needed security. If multiple applications run on a single server system, each might have its own data encryption and protection for the data in transit. The advantages of IPsec are that it is standards-based, available across different operating systems, and totally transparent to applications.

The Secure Sockets Layer (SSL) is an application layer protocol that can also be used to encrypt data flow. The client and server components must be SSL aware. They must be built with SSL libraries and with proper versions to interoperate. SSL is

used to protect the web browser and web server data flow (for example, HTTPS protocol). Most of the SSL implementations require certificates for the server and, in some cases, the clients too. IPsec can use certificates, but it can be implemented without certificates by using manual-key information.

In a typical corporate applications environment, multiple software products from different vendors run on different operation systems. It is more appropriate to use a technology that is transparent to applications to secure data flow. Some software applications cannot use SSL. Any kind of encryption of data flow impacts performance. This also applies to IPsec.

Hardware crypto-accelerators are useful for performance tuning. For example, you can use the Sun™ Crypto Accelerator 4000 board. The application load characteristics, performance characteristics, and service-level agreement (SLA) requirements should be considered with this security-to-performance evaluation.

A specific trade off in deploying IPsec is the *keys* used for encryption and decryption. IKE-based key management might not be available in the target IPsec implementation. If you manually generate keys, be careful to keep them secure, and change them on a regular schedule.

Deployment Scenario

IPsec can secure all of the traffic between two systems, between specific ports, or between specific boxes. The traffic can be bypassed (that is, it is not encrypted) or dropped. SSL secures traffic mainly between client and server application components.

In the following deployment example, an iPlanet™ web proxy server is deployed outside the internal firewall. The proxy server uses an iPlanet LDAP server to authenticate users. The user ID and password information are transmitted without encryption. Therefore, someone inside the enterprise could snoop the traffic for passwords. In addition, the version of the proxy server cannot communicate over SSL to LDAP.

A servlet or JavaServer Pages software is deployed on a web container using Java™ Database Connectivity (“JDC™”) software to communicate to an RDBMS server on a different machine. The traffic between the web container box and the RDBMS server box needs to be secure.

IPsec Configuration

The IPsec configuration required for the deployment example involves setting up an IPsec policy file and the keys for encryption and creating cryptographic hashes. This section contains procedures to set up the policy file and keys, to create the hashes, to test the setup, to enable it, and to check the status of the network traffic.

The examples in this section are shown using DES encryption. It is recommended to use 3DES, AES, or Blowfish. The availability of these encryption algorithms could be different on different versions of the operating system.

▼ To Set Up the IPsec Policy File

The `ipsecconf`(1M) utility (`/usr/sbin/ipsecconf`) configures the IPsec policy file for a host. After the policy is configured, all inbound and outbound datagrams are subject to policy checks. If no entry is found, no policy check will be completed, and all of the traffic will pass through. The policy is configured in the `/etc/inet/ipsecinit.conf` file. Each entry protects traffic in one direction. For details on the syntax of the entries, refer to `ipsecconf`(1M). A sample policy file, `/etc/inet/ipsecinit.sample`, is available in the Solaris OS.

1. **Copy and rename the `ipsecinit.sample` file to `/etc/init/ipsecinit.conf`.**
2. **Edit the file to add your requirements.**

   The following is a sample entry in the policy file.

```
{dport 23} apply {encr_algs des encr_auth_algs md5 sa shared}
{sport 23} permit {encr_algs des encr_auth_algs md5}
```

   The above entry secures `telnet`(1) traffic (port 23) with DES encryption and MD5 for hash. The first entry applies to the outbound traffic to be encrypted with DES. The second entry applies to the inbound traffic.

3. **Load the policy file by typing:**

```
$ ipsecconf  -a /etc/inet/ipsecinit.conf
```

   The policy file needs to be set up on both systems. The manual key file is identical on both machines—only the policy differs (that is, source and destination entries).

IPsec Keys

In an IPsec deployment for a financial company, the `/usr/sbin/ipseckey` command was used to set up manual key information for the communicating hosts.

The Solaris 8 OS does not provide automated key management. The Solaris 9 OS provides IKE-based key management. When creating keys, take care to provide the proper key length. For example, DES requires a 64-bit key, and 3DES requires a 192-bit key. MD5 requires a 128-bit key, and SHA-1 requires a 160-bit key. When you translate these into, for example, hex digits, 3DES would have 48-hex digits and SHA-1 would have 50 digits.

For the Solaris 8 OS, you can use the output from an `od -x < /dev/random` command to generate random digits and to get as many numbers as are required by the encryption, hashing algorithm. The following is sample random output.

```
# od -x </dev/random | head -4
0000000 89e9 f7d5 d1da 3a51 7805 44fb 6b44 e442
0000020 fbb8 ffd4 4835 58c6 10b2 cd35 a783 68ed
0000040 d726 15ff 22f1 32d3 e2ae 72a0 f847 ee17
0000060 f13d 8dfb 0539 cc41 4871 d40a cd43 7814
```

A sample key file consists of the following code.

```
add esp spi 1135 src 100.27.364.17 dst 19.15.57.46  auth_alg sha1
authkey bde359723576fdea08e56cbe876e24ad0123456 encr_alg 3des
encrkey 8bd4a52e10127deb4057210346122e1f283cb644d2c88012
add esp spi 1138 src 19.15.57.46 dst 100.27.364.17  auth_alg sha1
authkey bde359723576fdea08e56cbe876e24ad0123456 encr_alg 3des
encrkey 8bd4a52e10127deb4057210346122e1f283cb644d2c88012
```

The first entry adds ESP as the IPsec protocol (AH is the other option) between two systems with IP addresses 100.27.364.17 and 19.15.57.46. The hash algorithm is SHA-1, and the encryption algorithm is 3DES. You can observe 40 digits in `authkey` (that is, 160 bits for SHA-1) and 48 digits in `encrkey` (that is, 192 bits for 3DES). The security parameter index (SPI) number is 1135. This is an arbitrary number. The SPI number should match between the two systems. For detailed syntax of the policy file, refer to `ipseckey`(1M).

Load the keys with the following command. Provide the correct path name for the *keyfile* with the `-f` argument.

```
# ipseckey  -f keyfile.myhost
```

Sample IPsec Configuration

This section contains a sample IPsec configuration between 19.15.57.46 and 100.27.36.17 to encrypt LDAP traffic, but to bypass HTTP traffic.

The following is the policy file in 19.15.57.46.

```
{sport 8080} bypass {dir out}
{dport 8080} bypass {dir in}
{daddr 19.15.57.46 sport 389} apply {encr_algs 3DES encr_auth_algs
sha1 sa shared}
{saddr 19.15.57.46 dport 389} permit {encr_algs 3DES
encr_auth_algs sha1}
```

The following is the key file in 19.15.57.46.

```
add esp spi 1135 src 100.27.36.17 dst 19.15.57.46  auth_alg sha1
authkey bde359723576fdea08e56cbe876e24ad0123456 encr_alg 3des
encrkey 8bd4a52e10127deb4057210346122e1f283cb644d2c88012
add esp spi 1138 src 19.15.57.46 dst 100.27.36.17  auth_alg sha1
authkey bde359723576fdea08e56cbe876e24ad0123456 encr_alg 3des
encrkey 8bd4a52e10127deb4057210346122e1f283cb644d2c88012
```

The following is the policy file in 100.27.36.17.

```
{sport 8080} bypass {dir in}
{dport 8080} bypass {dir out}
{daddr 100.27.36.17 dport 389} apply {encr_algs 3DES
encr_auth_algs sha1 sa shared}
{saddr 100.27.36.17 sport 389} permit {encr_algs 3DES
encr_auth_algs sha1
```

The following is the key file in 100.27.36.17.

```
add esp spi 1135 src 100.27.36.17 dst 19.15.57.46  auth_alg sha1
authkey bde359723576fdea08e56cbe876e24ad0123456 encr_alg 3des
encrkey 8bd4a52e10127deb4057210346122e1f283cb644d2c88012
add esp spi 1138 src 19.15.57.46 dst 100.27.36.17  auth_alg sha1
authkey bde359723576fdea08e56cbe876e24ad0123456 encr_alg 3des
encrkey 8bd4a52e10127deb4057210346122e1f283cb644d2c88012
```

▼ To Sanity Test IPsec

1. **Capture** snoop**(1M) output between the two systems before IPsec is enabled, as in:**

```
# snoop -v -o snoop.out fromsystem tosystem
```

2. **Use the application (for example, the proxy authenticating to LDAP).**

▼ To Enable IPsec

1. **Capture** snoop**(1M) output between the two systems after IPsec is enabled, as in:**

```
# snoop -v -o snoop.out fromsystem tosystem
```

2. **Use the application.**

   You should observe the ESP packets in the post IPsec traffic. The snoop(1M) output should be similar to the following.

```
IP:
ESP:  ----- Encapsulating Security Payload -----
ESP:
ESP:  SPI = 0xa8dccd8e
ESP:  Replay = 9
ESP:     ....ENCRYPTED DATA....
```

   This example shows that the packets are encrypted and that there is no plain text data.

The following command can be used for status reports.

```
# ndd /dev/ipsecesp ipsecesp_status
ESP status
----------
Authentication algorithms          =   2
Encryption Algorithms              =   3
Packets passing authentication     =   0
Packets failing authentication     =   0
Packets apparently decrypting badly =   0
Packets failing replay checks      =   0
Packets failing early replay checks =   0
Failed inbound SA lookups          =   80
Inbound PF_KEY messages            =   12
Inbound ESP packets                =   80
Outbound ESP requests              =   78
PF_KEY ACQUIRE messages            =   23
Expired associations (# of bytes)  =   0
Discarded inbound packets          =   80
```

IPsec Caveats

If you use IPsec, keep the following in mind:

- IPsec in the current implementation does not work with Network Address Translation (NAT).
- For the Solaris 8 OS, you must download the encryption packages separately. Download the encryption package from the following site:

  `http://wwws.sun.com/software/solaris/encryption/download.html`

  DES and 3DES are included with the package in the Solaris 8 OS.
- For the Solaris 9 OS, DES and 3DES are built-in, but you can get AES and Blowfish with a *different* cryptography package from the following site:

  `http://wwws.sun.com/software/download/security.html`
- In the Solaris 9 OS, IPsec configuration provides more `ipsecconf`(1M) options with different syntax. You can use one directive for inbound or outbound traffic. For more information about the Solaris 9 OS IPsec configuration, refer to the Solaris 9 OS documentation:

  `http://docs.sun.com/db/doc/806-4075/6jd69oahv?q=IPSEC&a=view`

SSL and IPsec

The primary objective of the Secure Socket Language (SSL, also referred to as the transport-layer security protocol—TLS) is to provide secure connections between two applications (for example, an HTTP server and an HTTP client). SSL is an application-layer protocol.

IPsec operates at the network layer, so it is transparent to applications. Because it secures all of the traffic between two systems, it also secures all of the applications between the two systems. You can configure IPsec policy for specific applications at the port level (for example, Port 80 for HTTP and Port 389 for LDAP).

The hashing and encryption algorithms for TLS and IPsec are similar. To use SSL to secure traffic, the client and server component of the application must support SSL.

Recommendations

Software architects should look at IPsec to provide application-transparent security. As more and more security breaches occur from within the enterprise, IPsec offers valuable encryption that could minimize security risks. IPsec encryption and authentication are standards-based and implemented in multiple operating systems. You can selectively encrypt data flow on a protocol basis (for example, encrypting LDAP, but allowing HTTP).

As mentioned, IPsec handles one part of a total information security system. The proper hardening and minimization of operating systems, the application of security patches, secure coding, and enforcement of corporate security policies all complement IPsec deployments.

IPsec Reference

The following URLs are useful finding more information about IPsec.

Sun URLs

- ——. “Implementing IPsec.” *System Administration Guide, Volume 3*. Solaris 8 Administrator Collection. Santa Clara, CA: Sun Microsystems, Inc., 2003.

  `http://docs.sun.com/db/doc/806-0916/`

- Sun Global Sales. *IPSec in the Solaris™ 9 Operating Environment: A Technical White Paper*. FE1810-0. Palo Alto, CA: Sun Microsystems, Inc., 2002.

  `http://www.sun.com/software/whitepapers/solaris9/ipsec.pdf`

- ——. *IPsec and IKE Administration Guide*. Solaris™ 9 8/03 System Administrator Collection. Santa Clara, CA: Sun Microsystems, Inc., 2003.

  `http://docs.sun.com/db/doc/816-7264?q=IPSEC`

- SunSolve[SM] Program. “IPSec Encryption Algorithms Still Unavailable After Installing the Solaris 8 Data Encryption Package” Sun BugID: 72356. Santa Clara, CA: Sun Microsystems, Inc., 2004.

  `http://sunsolve.sun.com/`

- SunSolve Program. “Solaris Security: ‘Is `des`(1) needed for IPSec encryption functionality in Solaris 9?’” Sun BugID: 72664. Santa Clara, CA: Sun Microsystems, Inc., 2004.

  `http://sunsolve.sun.com/`

- SunSolve Program. “IP Security (IPsec) Components” Santa Clara, CA: Sun Microsystems, Inc.

  `http://sunsolve.central/`

RFCs

- "Security Architecture for the Internet Protocol" (RFC 2401)

  `ftp://ftp.rfc-editor.org/in-notes/rfc2401.txt`

- "IP Authentication Header" (RFC 2402)

  `ftp://ftp.rfc-editor.org/in-notes/rfc2402.txt`

- "IP Encapsulating Security Payload (ESP)" (RFC 2406)

  `ftp://ftp.rfc-editor.org/in-notes/rfc2406.txt`

- "The Internet Key Exchange (IKE)" (RFC 2409)

  `ftp://ftp.rfc-editor.org/in-notes/rfc2409.txt`

- "The OAKLEY Key Determination Protocol" (RFC 2412)

  `ftp://ftp.rfc-editor.org/in-notes/rfc2412.txt`

Acknowledgements

I would like to thank Mark Thacker, Sharon Read Veach, and Dan McDonald for their reviews of this article. I would also like to thank Mark Paulis and Thin-Fong for their help in the field. Finally, I would like to thank Dan Barnett and the Sun BluePrints program for their support and the SunSolve program for their helpful documentation.

Author Bio

Regu Rajaiah is a security technology ambassador at Sun. He currently works as a Solution Architect in the Northeast financial services group. Regu has more than ten years of experience in design and deployment of multitier applications. Regu holds a B.S. in Computer Science and an MBA from Indian Institute of Management, Ahemedabad, India, and he is a Certified Information Systems Security Professional (CISSP). Regu is deeply interested in information security, specifically application security, and he loves books on astronomy, nature, science, and technology.

Third-Party URLs

Third-party URLs are referenced in this document and provide additional, related information.

Note – Sun is not responsible for the availability of third-party web sites mentioned in this document. Sun does not endorse and is not responsible or liable for any content, advertising, products, or other materials that are available on or through such sites or resources. Sun will not be responsible or liable for any actual or alleged damage or loss caused by or in connection with the use of or reliance on any such content, goods, or services that are available on or through such sites or resources.

Ordering Sun Documents

The SunDocs^{SM} 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`