Gestión de cifrado y certificados en Oracle® Solaris 11.2

Referencia: E53958-02
Septiembre de 2014

ORACLE®

Contenido

Uso de esta documentación

En *Gestión centralizada de cifrado y certificados en Oracle® Solaris 11.2*, se explica cómo administrar y usar el cifrado, y cómo crear y gestionar certificados de clave privada o pública.

- **Descripción general**: describe conceptos sobre la estructura criptográfica y la estructura de gestión de claves y tareas para usar estar tecnologías para proteger archivos.
- **Destinatarios**: los administradores del sistema que deben implementar la seguridad en la empresa.
- **Conocimiento requerido**: estar familiarizado con terminología y conceptos de seguridad.

Biblioteca de documentación del producto

En la biblioteca de documentación (http://www.oracle.com/pls/topic/lookup?ctx=E56339), se incluye información de última hora y problemas conocidos para este producto.

Acceso a My Oracle Support

Los clientes de Oracle tienen acceso a soporte electrónico por medio de My Oracle Support. Para obtener más información, visite http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info o, si tiene alguna discapacidad auditiva, visite http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs.

Comentarios

Envíenos comentarios acerca de esta documentación mediante http://www.oracle.com/goto/docfeedback.

♦ ♦ ♦ CAPÍTULO 1

Estructura criptográfica

En este capítulo, se describe la función de estructura criptográfica de Oracle Solaris y se tratan los siguientes temas:

- “Introducción a la estructura criptográfica” [7]
- “Conceptos en la estructura criptográfica” [9]
- “Comandos y complementos de la estructura criptográfica” [11]
- “Zonas y servicios criptográficos” [13]
- “Estructura criptográfica y FIPS 140” [13]
- “Compatibilidad con OpenSSL en Oracle Solaris” [14]

Para administrar y utilizar la estructura criptográfica, consulte Capítulo 3, Estructura criptográfica.

Novedades de Criptografía para Oracle Solaris 11.2

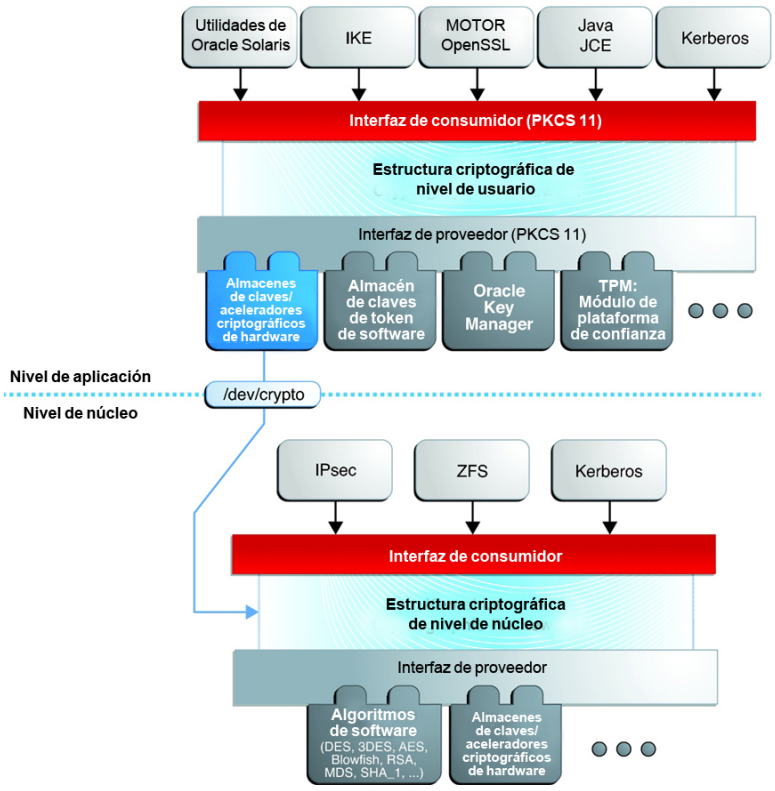
Esta sección resalta información para los clientes existentes sobre nuevas funciones de soporte de cifrado en esta versión.

- Oracle Solaris admite versiones tanto compatibles como no compatibles con FIPS de OpenSSL.
- En los sistemas SPARC T4 con optimizaciones criptográficas, están disponibles instrucciones criptográficas directamente en el hardware, que permite que las operaciones criptográficas se ejecuten más rápidamente.
- La estructura criptográfica admite Camellia, un cifrado de bloques de 128 bits que es similar a AES y se utiliza principalmente en el mercado japonés.

Introducción a la estructura criptográfica

La estructura criptográfica proporciona un almacén común de algoritmos y bibliotecas PKCS #11 para manejar los requisitos criptográficos. Las bibliotecas PKCS #11 se implementan según el estándar RSA Security Inc. PKCS #11 Cryptographic Token Interface (Cryptoki).

FIGURA 1-1 Niveles de estructura criptográfica



En el nivel del núcleo, la estructura actualmente administra los requisitos criptográficos para ZFS, Kerberos e IPsec, así como para el hardware. Los consumidores en el nivel de usuario incluyen el motor de OpenSSL, extensiones criptográficas Java (JCE), `libsasl` e IKE (protocolo de clave de Internet). El proxy de SSL de núcleo (`kssl`) utiliza la estructura criptográfica. Para obtener más información, consulte “Proxy SSL en el nivel del núcleo cifra las comunicaciones con el servidor web” de “Protección de la red en Oracle Solaris 11.2 ” y la página del comando man `kssl`cfg(1M).

La ley de exportación de los Estados Unidos exige que el uso de interfaces criptográficas abiertas sea bajo licencia. La estructura criptográfica cumple con la ley actual mediante la solicitud de que los proveedores criptográficos de núcleo y los proveedores criptográficos de PKCS #11 estén registrados. Para obtener más información, consulte la información sobre el comando `elfsign` en “Comandos de nivel de usuario de la estructura criptográfica” [12].

La estructura activa *providers* de servicios criptográficos para que sus servicios sean utilizados por muchos *consumidores* en Oracle Solaris. Otro nombre para los proveedores es *complementos*. La estructura admite tres tipos de complementos:

- Complementos de nivel de usuario: objetos compartidos que prestan servicios mediante bibliotecas PKCS #11, como `/var/user/$USER/pkcs11_softtoken.so.1`.
- Complementos de nivel de núcleo: módulos de núcleo que proporcionan implementaciones de algoritmos criptográficos en software, como AES.

  Muchos de los algoritmos de la estructura están optimizados para x86 con el conjunto de instrucciones SSE2 y para hardware SPARC. Para optimizaciones de series T, consulte "Estructura criptográfica y servidores SPARC T-Series" [17].
- Complementos de hardware: controladores de dispositivos y sus aceleradores de hardware asociados. Los chips Niagara y los controladores de dispositivos `ncp` y `n2cp` de Oracle son un ejemplo. Un acelerador de hardware descarga funciones criptográficas que consumen muchos recursos del sistema operativo. La placa Crypto Accelerator 6000 de Sun es un ejemplo.

La estructura implementa una interfaz estándar, la biblioteca PKCS #11, v2.20 enmienda 3, para proveedores de nivel de usuario. La biblioteca puede ser utilizada por aplicaciones de terceros para acceder a los proveedores. Terceros también pueden agregar bibliotecas registradas, módulos de algoritmos de núcleo registrados y controladores de dispositivos registrados a la estructura. Estos complementos se agregan cuando Image Packaging System (IPS) instala el software de terceros. Para ver un diagrama de los principales componentes de la estructura, consulte la Figura 1-1, "Niveles de estructura criptográfica".

Conceptos en la estructura criptográfica

Tenga en cuenta las siguientes descripciones de conceptos y los ejemplos correspondientes que son útiles al trabajar con la estructura criptográfica.

- **Algoritmos:** se establecen algoritmos criptográficos, procedimientos informáticos recursivos que cifran la entrada o le aplican hash. Los algoritmos de cifrado pueden ser simétricos o asimétricos. Los algoritmos simétricos utilizan la misma clave para el cifrado y el descifrado. Los algoritmos asimétricos, que se utilizan en la criptografía de claves públicas, necesitan dos claves. Las funciones de hashing también son algoritmos.

  Ejemplos de algoritmos:

  - Algoritmos simétricos, como AES y ECC
  - Algoritmos asimétricos, como Diffie-Hellman y RSA
  - Funciones de hashing, como SHA256
- **Consumidores:** usuarios de los servicios criptográficos prestados por los proveedores. Los consumidores pueden ser aplicaciones, usuarios finales u operaciones de núcleo.

  Ejemplos de consumidores:

  - Aplicaciones, como IKE

- Usuarios finales, como un usuario común que ejecuta el comando `encrypt`
- Operaciones de núcleo, como IPsec

- **Almacén de claves:** en la estructura criptográfica, el almacenamiento persistente para los objetos de token (se suele utilizar indistintamente con **token**). Para obtener información sobre un almacén de claves reservado, consulte **Metarranura** en esta lista de definiciones.
- **Mecanismo:** la aplicación de un modo de un algoritmo para un fin particular.

  Por ejemplo, un mecanismo DES que se aplica a la autenticación, como CKM_DES_MAC, es un mecanismo distinto de un mecanismo DES que se aplica al cifrado, CKM_DES_CBC_PAD.
- **Metarranura:** una ranura única que presenta una unión de las capacidades de otras ranuras que se cargan en la estructura. La metarranura facilita la tarea de manejar todas las capacidades de los proveedores que están disponibles mediante la estructura. Cuando una aplicación que utiliza la metarranura solicita una operación, la metarranura determina qué ranura actual realizará la operación. Las capacidades de la metarranura son configurables, pero no se requiere configuración. La metarranura está activada de manera predeterminada.

  Para obtener más información, consulte la página del comando man `cryptoadm`(1M).

  La metarranura no tiene su propio almacén de claves. En su lugar, la metarranura se reserva el uso de un almacén de claves a partir de las ranuras reales en la estructura criptográfica. De manera predeterminada, la metarranura se reserva el almacén de claves `Sun Crypto Softtoken`. El almacén de claves que utiliza la metarranura no se muestra como una de las ranuras disponibles.

  Los usuarios pueden especificar un almacén de claves alternativo para la metarranura mediante la configuración de las variables de entorno `${METASLOT_OBJECTSTORE_SLOT}` y `${METASLOT_OBJECTSTORE_TOKEN}` o mediante la ejecución del comando `cryptoadm`.

  Para obtener más información, consulte las páginas del comando man `libpkcs11`(3LIB), `pkcs11_softtoken`(5) y `cryptoadm`(1M).
- **Modo:** una versión de un algoritmo criptográfico. Por ejemplo, CBC (Cipher Block Chaining) es un modo distinto de ECB (Electronic Code Book). El algoritmo AES tiene dos modos: CKM_AES_ECB y CKM_AES_CBC.
- **Política:** la elección, por parte de un administrador, de qué mecanismos estarán disponibles para su uso. De manera predeterminada, todos los proveedores y todos los mecanismos están disponibles para su uso. La activación o desactivación de cualquier mecanismo sería una aplicación de la política. Para obtener ejemplos de configuración y aplicación de políticas, consulte “Administración de la estructura criptográfica” [34].
- **Proveedores:** servicios criptográficos que utilizan los consumidores. Los proveedores se conectan a la estructura, por lo que también se denominan *complementos*.

  Ejemplos de proveedores:

  - Bibliotecas PKCS #11, como `/var/user/$USER/pkcs11_softtoken.so`
  - Módulos de algoritmos criptográficos, como `aes` y `arcfour`
  - Controladores de dispositivos y sus aceleradores de hardware asociados, como el controlador `mca` para Sun Crypto Accelerator 6000

- **Ranura:** una interfaz de uno o más dispositivos criptográficos. Cada ranura, que corresponde a un lector físico o a otra interfaz de dispositivo, puede contener un token. Un token proporciona una vista lógica de un dispositivo criptográfico en la estructura.
- **Token:** en una ranura, un token proporciona una vista lógica de un dispositivo criptográfico en la estructura.

Comandos y complementos de la estructura criptográfica

La estructura proporciona comandos para los administradores, los usuarios y los desarrolladores que suministran proveedores.

- Comandos administrativos: el comando `cryptoadm` ofrece un subcomando `list` para mostrar los proveedores disponibles y sus capacidades. Los usuarios comunes pueden ejecutar los comandos `cryptoadm list` y `cryptoadm --help`.

  Para todos los demás subcomandos `cryptoadm`, es necesario que asuma un rol que incluya el perfil de derechos de gestión de criptografía o que se convierta en superusuario. Los subcomandos como `disable`, `install` y `uninstall` están disponibles para administrar la estructura. Para obtener más información, consulte la página del comando man `cryptoadm`(1M).

  El comando `svcadm` se utiliza para gestionar el daemon `kcfd` y para actualizar la política criptográfica en el núcleo. Para obtener más información, consulte la página del comando man `svcadm`(1M).
- Comandos de nivel de usuario: los comandos `digest` y `mac` proporcionan servicios de integridad de archivos. Los comandos `encrypt` y `decrypt` protegen los archivos contra intrusos. Para utilizar estos comandos, consulte Tabla 3-1, “Protección de los archivos con la estructura criptográfica (mapa de tareas)”.

Comandos administrativos de la estructura criptográfica

El comando `cryptoadm` administra una estructura criptográfica en ejecución. El comando forma parte del perfil de derechos de gestión de criptografía. Este perfil se puede asignar a un rol para una administración segura de la estructura criptográfica. Puede utilizar el comando `cryptoadm` para realizar lo siguiente:

- Activar o desactivar mecanismos de proveedores
- Activar o desactivar la metarranura

El comando `svcadm` se utiliza para activar, actualizar y desactivar el daemon de servicios criptográficos, `kcfd`. Este comando forma parte de la utilidad de gestión de servicios (SMF)

de Oracle Solaris. `svc:/system/cryptosvcs` es la instancia de servicio para la estructura criptográfica. Para obtener más información, consulte las páginas del comando man `smf`(5) y `svcadm`(1M).

Comandos de nivel de usuario de la estructura criptográfica

La estructura criptográfica proporciona comandos de nivel de usuario para comprobar la integridad de los archivos, cifrar archivos y descifrar archivos.

- Comando `digest`: procesa un resumen de mensaje para uno o varios archivos o para stdin. Un resumen es útil para verificar la integridad de un archivo. SHA1 y MD5 son ejemplos de funciones de resumen.
- Comando `mac`: procesa un código de autenticación de mensajes (MAC) para uno o varios archivos o para stdin. Un MAC asocia datos con un mensaje autenticado. Un MAC le permite a un receptor verificar que el mensaje provenga del remitente y no haya sido alterado. Los mecanismos `sha1_mac` y `md5_hmac` pueden procesar un MAC.
- Comando `encrypt`: cifra los archivos o stdin con un cifrado simétrico. El comando `encrypt -l` muestra los algoritmos que están disponibles. Los mecanismos incluidos en una biblioteca de nivel de usuario están disponibles para el comando `encrypt`. La estructura proporciona mecanismos AES, DES, 3DES (Triple-DES) y ARCFOUR para el cifrado del usuario.
- Comando `decrypt`: descifra archivos o stdin que se cifraron con el comando `encrypt`. El comando `decrypt` utiliza la misma clave y el mismo mecanismo que se utilizaron para cifrar el archivo original.
- Comando `elfsign`: proporciona un medio para firmar los proveedores que se utilizarán con la estructura criptográfica. Normalmente, este comando es ejecutado por el desarrollador de un proveedor. El comando `elfsign` tiene subcomandos para solicitar un certificado, firmar binarios y verificar la firma en un binario. Los archivos binarios no registrados no pueden ser utilizados por la estructura criptográfica. Los proveedores que tengan binarios firmados verificables pueden utilizar la estructura.

Complementos de la estructura criptográfica

Los terceros pueden conectar sus proveedores a la estructura criptográfica. Un proveedor de terceros puede ser uno de los siguientes objetos:

- Biblioteca compartida PKCS #11

- Módulo de software de núcleo cargable, como un algoritmo de cifrado, una función MAC o una función de resumen
- Controlador de dispositivo de núcleo para un acelerador de hardware

Los objetos de un proveedor se deben firmar con un certificado de Oracle. La solicitud de certificados se basa en una clave privada que un tercero selecciona y un certificado que proporciona Oracle. La solicitud de certificado se envía a Oracle, que registra al tercero y, a continuación, expide el certificado. El tercero, a continuación, registra su objeto de proveedor con el certificado de Oracle.

Los módulos de software de núcleo cargables y los controladores de dispositivos de núcleo para aceleradores de hardware también se deben registrar en el núcleo. El registro se lleva a cabo mediante la interfaz del proveedor de servicios (SPI) de la estructura criptográfica

Zonas y servicios criptográficos

La zona global y cada zona no global tienen su propio servicio `/system/cryptosvc`. Cuando se activa o se actualiza el servicio criptográfico en la zona global, se inicia el daemon `kcfd` en la zona global, se define la política de nivel de usuario para la zona global y se establece la política de núcleo para el sistema. Cuando se activa o se actualiza el servicio en una zona no global, se inicia el daemon `kcfd` en la zona y se define la política de nivel de usuario para la zona. La política de núcleo fue definida por la zona global.

Para obtener más información sobre las zonas, consulte “Introducción a Zonas de Oracle Solaris ”. Para obtener más información sobre el uso de la SMF para gestionar aplicaciones persistentes, consulte Capítulo 1, “Introducción a la Utilidad de gestión de servicios” de “Gestión de los servicios del sistema en Oracle Solaris 11.2 ” y la página del comando man `smf`(5).

Estructura criptográfica y FIPS 140

FIPS 140 es un estándar de seguridad informática del Gobierno de los Estados Unidos para módulos criptográficos. Los sistemas Oracle Solaris ofrecen dos proveedores de algoritmos criptográficos aprobados para el nivel 1 de FIPS 140-2.

Esos proveedores son:

- La estructura criptográfica de Oracle Solaris proporciona dos módulos aprobados por FIPS 140. El módulo *userland* reemplaza la criptografía para aplicaciones que se ejecutan en el espacio del usuario. El módulo *kernel* proporciona criptografía para procesos de nivel de núcleo.

- El módulo del objeto OpenSSL proporciona criptografía aprobada por FIPS 140 para SSH y aplicaciones web.

Tenga en cuenta las siguientes consideraciones clave:

- Debido a que los módulos del proveedor FIPS 140-2 hacen un uso intensivo del CPU, no están activados de forma predeterminada. Como administrador del sistema, usted tiene la responsabilidad de activar los proveedores en el modo FIPS 140 y configurar aplicaciones que usan algoritmos aprobados por FIPS.
- Si tiene un requisito estricto para usar solo criptografía validada por FIPS 140-2, debe ejecutar la versión Oracle Solaris 11.1 SRU 5.5 o la versión Oracle Solaris 11.1 SRU 3. Oracle completó una validación de FIPS 140-2 en la estructura criptográfica de Solaris en estas dos versiones específicas. Oracle Solaris11.2 se crea en esta base validada e incluye mejoras de software que apuntan al rendimiento, la funcionalidad y la confiabilidad. Siempre que sea posible, debería configurar Oracle Solaris11.2 en el modo FIPS 140-2 para aprovechar estas mejoras.

Para obtener información, consulte "Using a FIPS 140 Enabled System in Oracle Solaris 11.2 ". En este artículo, se tratan los siguientes temas:

- Descripción general de la criptografía de nivel 1 de FIPS 140-2 en Oracle Solaris
- Activación de proveedores FIPS 140
- Activación de consumidores FIPS 140
- Ejemplo de activación de dos aplicaciones en modo FIPS 140
- Referencias de certificados y algoritmos aprobados por FIPS 140

La siguiente información adicional está disponible:

- Cómo pasar a la implementación de OpenSSL compatible con FIPS 140 [15]
- "Creación de un entorno de inicio con FIPS 140 activado " [42]

Compatibilidad con OpenSSL en Oracle Solaris

Oracle Solaris admite dos implementaciones de OpenSSL:

- OpenSSL compatible con FIPS 140
- OpenSSL no compatible con FIPS 140

Ambas implementaciones se actualizaron para ser compatibles con la versión más reciente de OpenSSL desde el proyecto OpenSSL, que es OpenSSL 1.0.1. Con respecto a bibliotecas de versiones, ambas son compatibles con API/ABI.

Mientras que ambas implementaciones están presentes en el sistema operativo, sólo una implementación puede estar activa a la vez. Para determinar qué implementación de OpenSSL está activa en el sistema, utilice el comando `pkg mediator openssl`.

▼ Cómo pasar a la implementación de OpenSSL compatible con FIPS 140

De manera predeterminada, la implementación de OpenSSL no compatible con FIPS 140 está activa en Oracle Solaris. Sin embargo, puede seleccionar la seguridad de su sistema y seleccionar la implantación que desee.

1. **Conviértase en un administrador.**

2. **Asegúrese de que ambas implementaciones estén en el sistema.**

```
$ pkg mediator -a openssl
```

Atención - La implementación de OpenSSL a la que está alternando debe existir en el sistema. De lo contrario, si conmuta a una implementación que no está en el sistema, el sistema puede convertirse en inutilizable.

3. **Conmute a otra implementación de OpenSSL.**

```
# pkg set-mediator [--be-name name] -I implementation openssl
```

donde *implementation* es `default` o `fips-140` y *name* es un nombre para un clon nuevo del entorno de inicio actual. El clon tendrá la implementación especificada activa.

Nota - Donde *--be-name* está especificado, el comando crea una copia de seguridad del entorno de inicio actual. Al reiniciar, el sistema ejecutará el nuevo entorno de inicio clonado con la nueva implementación.

Para obtener más información sobre el comando `pkg set-mediator`, consulte “Cambio de la aplicación preferida” de “Agregación y actualización de software en Oracle Solaris 11.2 ”.

4. **Reinicie el sistema.**

5. **(Opcional) Verifique que la conmutación se haya realizado correctamente y que la implementación de OpenSSL preferida está activa.**

```
# pkg mediator openssl
```

ejemplo 1-1 Cómo alternar a la implementación de OpenSSL compatible con FIPS 140

En este ejemplo, se cambia la implementación de OpenSSL de un sistema para que sea compatible con FIPS 140.

```
# pkg mediator -a openssl
MEDIATOR   VER. SRC.   VERSION IMPL.   SRC. IMPLEMENTATION
openssl       vendor           vendor               default
```

```
openssl       system          system              fips-140

# pkg set-mediator --be-name BE2 -I fips-140 openssl
# reboot

# pkg mediator openssl
MEDIATOR   VER. SRC.   VERSION IMPL.   SRC. IMPLEMENTATION
openssl      vendor           vendor              default
```

♦ ♦ ♦ CAPÍTULO 2

Acerca de los sistemas SPARC T-Series y la estructura criptográfica

En este capítulo se describen la estructura criptográfica en servidores SPARC T-Series y las optimizaciones en Oracle Solaris 11 que mejoran el rendimiento de las funciones criptográficas.

Estructura criptográfica y servidores SPARC T-Series

La estructura criptográfica le proporciona mecanismos criptográficos a los sistemas SPARC T-Series y optimiza algunos mecanismos para estos servidores. Hay tres mecanismos criptográficos optimizados para datos en reposo y en movimiento: `AES-CBC`, `AES-CFB128` y `ARCFOUR`. El mecanismo criptográfico DES está optimizado para OpenSSL y mediante la optimización de aritmética de precisión arbitraria (bignum), RSA y DSA. Otras optimizaciones incluyen el rendimiento de paquetes pequeños para establecimientos de comunicación y datos en movimiento.

Los siguientes mecanismos criptográficos están disponibles en esta versión:

- `AES-XTS`: se usa para los datos en reposo
- `SHA-224`: mecanismo `SHA2`
- `AES-XCBC-MAC`: se usa para IPsec

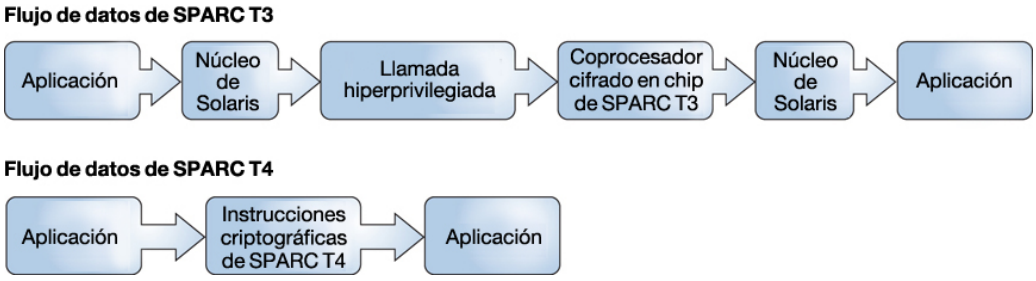
Optimizaciones criptográficas en Sistemas SPARC T-4

Desde el microprocesador SPARC T4, hay nuevas instrucciones para llevar a cabo funciones criptográficas directamente en el hardware. Las instrucciones son sin privilegios. Por lo tanto, cualquier programa puede utilizar las instrucciones sin requerir ningún entorno de núcleo, permisos de usuario root u otra configuración especial. La criptografía se realiza directamente en el hardware en lugar de utilizar diversas instrucciones de bajo nivel. Por lo tanto, las operaciones criptográficas son más rápidas comparadas con las operaciones en sistemas cuyos

procesadores SPARC anteriores tenían unidades de procesamiento independientes para la criptografía.

La siguiente comparación muestra las diferencias en el flujo de datos entre sistemas SPARC T-3 y sistemas SPARC T-4 con optimizaciones criptográficas.

FIGURA 2-1 Comparación de flujo de datos entre Sistemas SPARC T



En la siguiente tabla se proporciona una comparación detallada de funciones criptográficas en unidades de microprocesador de SPARC T combinadas con versiones específicas de Oracle Solaris.

TABLA 2-1 Rendimiento criptográfico en servidores SPARC T-Series

Consumidor de función/ software	T -3 y sistemas anteriores	Sistemas T-4 que ejecutan Oracle Solaris 10	Sistemas T-4 que ejecutan Oracle Solaris 11
SSH	Activado automáticamente en Solaris 10 5/09 y versiones posteriores. Activar/desactivar con la cláusula `UseOpenSSLEngine` in `/etc/ssh/sshd_config`.	Requiere el parche 147707-01. Activar/desactivar con la cláusula `UseOpenSSLEngine` in `/etc/ssh/sshd_config`.	Activado automáticamente. Activar/desactivar con la cláusula `UseOpenSSLEngine` in `/etc/ssh/sshd_config`
Java/JCE	Activado automáticamente. Configurar en `$JAVA_HOME/jre/lib/ security/java.security`.	Activado automáticamente. Configurar en `$JAVA_HOME/jre/lib/ security/java.security`.	Activado automáticamente. Configurar en `$JAVA_HOME/jre/lib/ security/java.security`.
ZFS criptográfico	No disponible.	No disponible.	HW criptográfico se activa automáticamente si el conjunto de datos está cifrado.
IPsec	Activado automáticamente.	Activado automáticamente.	Activado automáticamente.
OpenSSL	Utilizar `-engine pkcs11`.	Requiere el parche 147707-01. Utilizar `-engine pkcs11`.	La optimización de T4 se utiliza de forma automática.

Consumidor de función/ software	T -3 y sistemas anteriores	Sistemas T-4 que ejecutan Oracle Solaris 10	Sistemas T-4 que ejecutan Oracle Solaris 11
			(Opcionalmente, utilice `-engine pkcs11`). Se recomienda `pkcs11` para RSA/DSA en este punto.
KSSL (proxy SSL de núcleo)	Activado automáticamente.	Activado automáticamente.	Activado automáticamente.
Oracle TDE	No admitida.	Parche pendiente.	Activado automáticamente en Oracle DB 11.2.0.3 y ASO.
Apache SSL	Configurar con `SSLCrypto Device pkcs11`.	Configurar con `SSLCrypto Device pkcs11`.	Configurar con `SSLCryptoDevice pkcs11`.
Dominios lógicos	Asignar unidades criptográficas a los dominios.	La funcionalidad siempre está disponible; no se requiere configuración.	La funcionalidad siempre está disponible; no se requiere configuración.

Las instrucciones criptográficas de T4 incluyen lo siguiente:

- `aes_kexpand0`, `aes_kexpand1`, `aes_kexpand2`

  Estas instrucciones realizan la *expansión de clave*. Expanden la clave proporcionada por el usuario de 128 bits, 192 bits o 256 bits en una planificación de clave que se utiliza de manera interna durante el cifrado y el descifrado. La instrucción de `aes_kexpand2` sólo se utiliza para AES-256. Las otras dos instrucciones de `aes_kexpand` se utilizan para las tres longitudes de clave: AES-128, AES-192 y AES-256.
- `aes_eround01`, `aes_eround23`, `aes_eround01_l`, `aes_eround_23_l`

  Estas instrucciones se utilizan para *rondas* o transformaciones del cifrado AES. Según el estándar AES en FIPS 197, la cantidad de rondas utilizadas (por ejemplo 10, 12 o 14) varía según la longitud de la clave AES, ya que las claves más largas, en principio, indican el deseo de un cifrado más sólido a costas de mayores cálculos.
- `aes_dround01`, `aes_dround23`, `aes_dround01_l`, `aes_dround_23_l`

  Estas instrucciones se utilizan para rondas del descifrado AES de forma similar que con el cifrado.
- Instrucciones para DES/DES-3, Kasumi, Camellia, multiplicar/raíz cuadrada de Montgomery (para RSA Bignum) y sumas de comprobación de CRC32c
- Instrucciones de resumen de MD5, SHA1 y SHA2

Las instrucciones criptográficas de hardware de SPARC T4 están disponibles y se utilizan automáticamente en sistemas SPARC T4 que ejecutan Oracle Solaris 11 mediante el motor t4 integrado en el microprocesador T4 del sistema. A partir de Oracle Solaris 11.2, las instrucciones están incluidas en el código ascendente de OpenSSL. Por lo tanto, en esta versión, OpenSSL 1.0.1e se entrega con un parche para activar el uso de las instrucciones.

Para obtener más información sobre las instrucciones de T4, consulte los siguientes artículos.

- "Motor de OpenSSL SPARC T4" (https://blogs.oracle.com/DanX/entry/sparc_t4_openssl_engine)

- "Cómo identificar si se está utilizando SPARC T4 criptográfico" (https://blogs.oracle.com/DanX/entry/how_to_tell_if_sparc)
- "Esperando avances criptográficos con el procesador T4 y Oracle Solaris 11" (http://bubbva.blogspot.com/2011/11/exciting-crypto-advances-with-t4.html)
- "Optimizaciones criptográficas y resumen de SPARC T4 en Solaris 11.1" (https://blogs.oracle.com/DanX/entry/sparc_t4_digest_and_crypto)

Cómo determinar si el sistema admite las optimizaciones SPARC T4

Para determinar si las optimizaciones de T4 se emplean, utilice el comando `isainfo`. La inclusión de `sparcv9` y `aes` en la salida indica que el sistema está utilizando las optimizaciones.

```
$ isainfo -v
64-bit sparcv9 applications
        crc32c cbcond pause mont mpmul sha512 sha256 sha1 md5 camellia kasumi
        des aes ima hpc vis3 fmaf asi_blk_init vis2 vis popc
```

Cómo determinar la versión de OpenSSL del sistema

Para comprobar la versión de OpenSSL que se está ejecutando en el sistema, escriba `openssl version`. La salida es similar a la siguiente:

```
OpenSSL 1.0.0j 10 May 2012
```

Cómo verificar que el sistema tiene OpenSSL con optimizaciones de SPARC T4

Para determinar si su sistema admite OpenSSL con optimizaciones de SPARC T4, compruebe la biblioteca de `libcrypto.so` de la siguiente forma:

```
# nm /lib/libcrypto.so.1.0.0 | grep des_t4
[5239]  |    504192|       300|FUNC |GLOB |3    |12     |des_t4_cbc_decrypt
[5653]  |    503872|       300|FUNC |GLOB |3    |12     |des_t4_cbc_encrypt
[4384]  |    505024|       508|FUNC |GLOB |3    |12     |des_t4_ede3_cbc_decrypt
[2963]  |    504512|       508|FUNC |GLOB |3    |12     |des_t4_ede3_cbc_encrypt
[4111]  |    503712|       156|FUNC |GLOB |3    |12     |des_t4_key_expand
```

Si el comando no genera ninguna salida, el sistema no admite las optimizaciones de SPARC T4 para OpenSSL.

♦ ♦ ♦ CAPÍTULO 3

Estructura criptográfica

En este capítulo se describe cómo utilizar la estructura criptográfica y se tratan los siguientes temas:

Protección de los archivos con la estructura criptográfica

En esta sección, se describe cómo generar claves simétricas, cómo crear sumas de comprobación para la integridad de archivos y cómo proteger los archivos contra intrusos. Los comandos incluidos en esta sección pueden ser ejecutados por usuarios comunes. Los desarrolladores pueden escribir secuencias de comandos que utilicen estos comandos.

Para configurar el sistema en modo FIPS 140, debe usar algoritmos, modos y longitudes de claves validados por FIPS. Consulte "FIPS 140 Algorithm Lists and Certificate References for Oracle Solaris Systems" de "Using a FIPS 140 Enabled System in Oracle Solaris 11.2 ".

La estructura criptográfica puede ayudar a proteger los archivos. En el siguiente mapa de tareas se hace referencia a los procedimientos para mostrar los algoritmos disponibles y para proteger los archivos criptográficamente.

TABLA 3-1 Protección de los archivos con la estructura criptográfica (mapa de tareas)

Tarea	Descripción	Para obtener instrucciones
Generar una clave simétrica.	Genera una clave de la longitud especificada por el usuario. También puede almacenar la clave en un archivo, en un almacén de claves PKCS #11 o en un almacén de claves NSS. Para el modo aprobado por FIPS 140, seleccione un tipo de clave, un modo y la longitud de clave validados para FIPS. Consulte "FIPS 140 Algorithms in the Cryptographic Framework" de "Using a FIPS 140 Enabled System in Oracle Solaris 11.2 ".	Cómo generar una clave simétrica con el comando `pktool` [22]

Tarea	Descripción	Para obtener instrucciones
Proporcionar una suma de comprobación que garantice la integridad de un archivo.	Verifica que la copia de un archivo del receptor sea idéntica al archivo que se envió.	Cómo calcular un resumen de un archivo [28]
Proteger un archivo con un código de autenticación de mensajes (MAC).	Le comprueba al receptor del mensaje que usted era el remitente.	Cómo calcular un MAC de un archivo [29]
Cifrar un archivo y, a continuación, descifrar el archivo cifrado.	Protege el contenido de los archivos al cifrar el archivo. Proporciona los parámetros de cifrado para descifrar el archivo.	Cómo cifrar y descifrar un archivo [31]

▼ Cómo generar una clave simétrica con el comando `pktool`

Algunas aplicaciones requieren una clave simétrica para el cifrado y el descifrado de las comunicaciones. En este procedimiento, se crea una clave simétrica y se la almacena.

Si su sitio cuenta con un generador de números aleatorios, puede utilizar el generador para crear un número aleatorio para la clave. Este procedimiento no utiliza el generador de números aleatorios de su sitio.

1. **(Opcional) Si tiene previsto utilizar un almacén de claves, créelo.**

   - **Para crear e inicializar un almacén de claves PKCS #11, consulte Cómo generar una frase de contraseña mediante el comando `pktool setpin` [61].**
   - **Para crear e inicializar una base de datos NSS, consulte el comando de ejemplo en Ejemplo 4-5, "Protección de un almacén de claves con una frase de contraseña".**

2. **Genere un número aleatorio para usarlo como clave simétrica.**
   Utilice uno de los métodos siguientes.

   - **Genere una clave y almacénela en un archivo.**
     La ventaja de almacenar una clave en un archivo es que se puede extraer la clave de este archivo para usarla en el archivo de claves de una aplicación, como el archivo `/etc/inet/secret/ipseckeys` o IPsec. La instrucción de uso muestra los argumentos.

     ```
     % pktool genkey keystore=file
     ...genkey keystore=file
     outkey=key-fn
     [ keytype=aes|arcfour|des|3des|generic ]
     [ keylen=key-size (AES, ARCFOUR or GENERIC only)]
     ```

```
[ print=y|n ]
```

`outkey=`*key-fn*

El nombre de archivo donde se almacena la clave.

`keytype=`*specific-symmetric-algorithm*

Para una clave simétrica de cualquier longitud, el valor es `generic`. Para un algoritmo determinado, especifique `aes`, `arcfour`, `des` o `3des`.

Para los algoritmos aprobados por FIPS 140, seleccione un tipo de clave validado para FIPS. Consulte "FIPS 140 Algorithms in the Cryptographic Framework" de "Using a FIPS 140 Enabled System in Oracle Solaris 11.2 ".

`keylen=`*size-in-bits*

La longitud de la clave en bits. El número debe ser divisible por 8. *No* especifique para `des` ni `3des`.

Para los algoritmos aprobados por FIPS 140, seleccione una longitud de clave validada para FIPS. Consulte "FIPS 140 Algorithms in the Cryptographic Framework" de "Using a FIPS 140 Enabled System in Oracle Solaris 11.2 ".

`print=n`

Imprime la clave en la ventana de terminal. De manera predeterminada, el valor de `print` es `n`.

- **Genere una clave y almacénela en un almacén de claves PKCS #11.**

  La ventaja del almacén de claves PKCS #11 es que se puede recuperar la clave por su etiqueta. Este método es útil para las claves para cifrar y descifrar archivos. Debe completar el Paso 1 antes de utilizar este método. La instrucción de uso muestra los argumentos. Los corchetes alrededor del argumento del almacén de claves indican que cuando el argumento de almacén de claves no estás especificado, la clave se almacena en el almacén de claves PKCS #11.

```
$ pktool genkey keystore=pkcs11
...genkey [ keystore=pkcs11 ]
label=key-label
[ keytype=aes|arcfour|des|3des|generic ]
[ keylen=key-size (AES, ARCFOUR or GENERIC only)]
[ token=token[:manuf[:serial]]]
[ sensitive=y|n ]
[ extractable=y|n ]
[ print=y|n ]
```

`label=`*key-label*

Una etiqueta especificada por el usuario para la clave. La clave se puede recuperar del almacén de claves por su etiqueta.

`keytype=`*specific-symmetric-algorithm*

Para una clave simétrica de cualquier longitud, el valor es `generic`. Para un algoritmo determinado, especifique `aes`, `arcfour`, `des` o `3des`.

Para los algoritmos aprobados por FIPS 140, seleccione un tipo de clave validado para FIPS. Consulte "FIPS 140 Algorithms in the Cryptographic Framework" de "Using a FIPS 140 Enabled System in Oracle Solaris 11.2 ".

`keylen=`*size-in-bits*

La longitud de la clave en bits. El número debe ser divisible por 8. *No* especifique para `des` ni `3des`.

Para los algoritmos aprobados por FIPS 140, seleccione una longitud de clave validada para FIPS. Consulte "FIPS 140 Algorithms in the Cryptographic Framework" de "Using a FIPS 140 Enabled System in Oracle Solaris 11.2 ".

`token=`*token*

El nombre del token. De manera predeterminada, el token es `Sun Software PKCS#11 softtoken`.

`sensitive=n`

Especifica la sensibilidad de la clave. Cuando el valor es `y`, la clave no se puede imprimir utilizando el argumento `print=y`. De manera predeterminada, el valor de `sensitive` es `n`.

`extractable=y`

Especifica que la clave se puede extraer del almacén de claves. Especifique `n` para evitar que se extraiga la clave.

`print=n`

Imprime la clave en la ventana de terminal. De manera predeterminada, el valor de `print` es `n`.

- **Genere una clave y almacénela en un almacén de claves NSS.**

  Debe completar el Paso 1 antes de utilizar este método. La instrucción de uso muestra los argumentos.

```
$ pktool genkey keystore=nss
...genkey keystore=nss
label=key-label
[ keytype=aes|arcfour|des|3des|generic ]
[ keylen=key-size (AES, ARCFOUR or GENERIC only)]
[ token=token[:manuf[:serial]]]
[ dir=directory-path ]
[ prefix=DBprefix ]
```

`label=`*key-label*

Una etiqueta especificada por el usuario para la clave. La clave se puede recuperar del almacén de claves por su etiqueta.

`keytype=`*specific-symmetric-algorithm*

Para una clave simétrica de cualquier longitud, el valor es `generic`. Para un algoritmo determinado, especifique `aes`, `arcfour`, `des` o `3des`.

Para los algoritmos aprobados por FIPS 140, seleccione un tipo de clave validado para FIPS. Consulte "FIPS 140 Algorithms in the Cryptographic Framework" de "Using a FIPS 140 Enabled System in Oracle Solaris 11.2 ".

`keylen=`*size-in-bits*

La longitud de la clave en bits. El número debe ser divisible por 8. *No* especifique para `des` ni `3des`.

Para los algoritmos aprobados por FIPS 140, seleccione una longitud de clave validada para FIPS. Consulte "FIPS 140 Algorithms in the Cryptographic Framework" de "Using a FIPS 140 Enabled System in Oracle Solaris 11.2 ".

`token=`*token*

El nombre del token. De manera predeterminada, el token es el token interno NSS.

`dir=`*directorio*

La ruta de directorio a la base de datos NSS. De manera predeterminada, el valor de *directorio* es el directorio actual.

`prefix=`*directory*

El prefijo a la base de datos NSS. El valor predeterminado es sin prefijo.

3. **(Opcional) Compruebe que la clave exista.**

Utilice uno de los siguientes comandos, según dónde haya guardado la clave.

- **Verifique la clave en el archivo** *key-fn***.**

```
% pktool list keystore=file objtype=key [infile=key-fn]
Found n keys.
Key #1 - keytype:location (keylen)
```

- **Verifique la clave en el almacén de claves PKCS #11 o NSS.**

```
For PKCS #11, use the following command:

$ pktool list keystore=pkcs11 objtype=key
Enter PIN for keystore:
Found n keys.
```

`Key #1 -` *keytype*`:`*location (keylen)*

Como alternativa, reemplace `keystore=pkcs11` con `keystore=nss` en el comando.

ejemplo 3-1 Creación de una clave simétrica con el comando `pktool`

En el siguiente ejemplo, un usuario crea un almacén de claves PKCS #11 por primera vez y, a continuación, genera una clave simétrica de gran tamaño para una aplicación. Por último, el usuario verifica que la clave se encuentre en el almacén de claves.

Tenga en cuenta que la contraseña inicial para un almacén de claves PKCS #11 es `changeme`. La contraseña inicial para un almacén de claves NSS es una contraseña vacía.

```
# pktool setpin
Create new passphrase:      Type password
Re-enter new passphrase:      Retype password
Passphrase changed.
% pktool genkey label=specialappkey keytype=generic keylen=1024
Enter PIN for Sun Software PKCS#11 softtoken  :     Type password

% pktool list objtype=key
Enter PIN for Sun Software PKCS#11 softtoken  :     Type password
No.      Key Type      Key Len.      Key Label
-------------------------------------------------
Symmetric keys:
1        Symmetric     1024          specialappkey
```

ejemplo 3-2 Creación de una clave AES aprobada por FIPS con el comando `pktool`

En el siguiente ejemplo, se crea una clave secreta para el algoritmo AES mediante un algoritmo y una longitud de clave aprobados por FIPS. La clave se almacena en un archivo local para un posterior descifrado. El comando protege el archivo con `400` permisos. Cuando se crea la clave, la opción `print=y` muestra la clave generada en la ventana de terminal.

El usuario que posee el archivo de claves recupera la clave mediante el comando `od`.

```
% pktool genkey keystore=file outkey=256bit.file1 keytype=aes keylen=256 print=y
Key Value ="aaa2df1d10f02eaee2595d48964847757a6a49cf86c4339cd5205c24ac8c8873"
% od -x 256bit.file1

0000000 aaa2 df1d 10f0 2eae e259 5d48 9648 4775
0000020 7a6a 49cf 86c4 339c d520 5c24 ac8c 8873
0000040
```

ejemplo 3-3 Creación de una clave simétrica para las asociaciones de seguridad IPsec

En el siguiente ejemplo, el administrador crea manualmente el material clave para las asociaciones de seguridad de IPsec y las almacena en archivos. A continuación, el administrador copia las claves al archivo `/etc/inet/secret/ipseckeys` y destruye los archivos originales.

En primer lugar, el administrador crea y muestra las claves que la política IPsec requiere:

```
# pktool genkey keystore=file outkey=ipencrin1 keytype=generic keylen=192 print=y
Key Value ="294979e512cb8e79370dabecadc3fcbb849e78d2d6bd2049"
# pktool genkey keystore=file outkey=ipencrout1 keytype=generic keylen=192 print=y
Key Value ="9678f80e33406c86e3d1686e50406bd0434819c20d09d204"
# pktool genkey keystore=file outkey=ipspi1 keytype=generic keylen=32 print=y
Key Value ="acbeaa20"
# pktool genkey keystore=file outkey=ipspi2 keytype=generic keylen=32 print=y
Key Value ="19174215"
# pktool genkey keystore=file outkey=ipsha21 keytype=generic keylen=256 print=y
Key Value ="659c20f2d6c3f9570bcee93e96d95e2263aca4eeb3369f72c5c786af4177fe9e"
# pktool genkey keystore=file outkey=ipsha22 keytype=generic keylen=256 print=y
Key Value ="b041975a0e1fce0503665c3966684d731fa3dbb12fcf87b0a837b2da5d82c810"
```

A continuación, el administrador crea el siguiente archivo `/etc/inet/secret/ipseckeys`:

```
##   SPI values require a leading 0x.
##   Backslashes indicate command continuation.
##
## for outbound packets on this system
add esp spi 0xacbeaa20 \
src 192.168.1.1 dst 192.168.2.1 \
encr_alg aes auth_alg sha256  \
encrkey  294979e512cb8e79370dabecadc3fcbb849e78d2d6bd2049 \
authkey  659c20f2d6c3f9570bcee93e96d95e2263aca4eeb3369f72c5c786af4177fe9e
##
## for inbound packets
add esp spi 0x19174215 \
src 192.168.2.1 dst 192.168.1.1 \
encr_alg aes auth_alg sha256  \
encrkey 9678f80e33406c86e3d1686e50406bd0434819c20d09d204 \
authkey b041975a0e1fce0503665c3966684d731fa3dbb12fcf87b0a837b2da5d82c810
```

Después de verificar que la sintaxis del archivo `ipseckeys` sea válida, el administrador destruye los archivos de claves originales.

```
# ipseckey -c /etc/inet/secret/ipseckeys
# rm ipencrin1 ipencrout1 ipspi1 ipspi2 ipsha21 ipsha22
```

El administrador copia el archivo `ipseckeys` al sistema de comunicación mediante el comando `ssh` u otro mecanismo seguro. En el sistema de comunicación, las protecciones se revierten. La primera entrada en el archivo `ipseckeys` protege los paquetes entrantes y la segunda entrada protege los paquetes salientes. No se generan claves en el sistema de comunicación.

Pasos siguientes Para continuar con el uso de la clave para crear un código de autenticación de mensajes (MAC) para un archivo, consulte Cómo calcular un MAC de un archivo [29].

▼ Cómo calcular un resumen de un archivo

Cuando se calcula un resumen de un archivo, se puede comprobar que el archivo no haya sido alterado comparando los resultados del resumen. Un resumen no modifica el archivo original.

1. **Muestre los algoritmos de resumen disponibles.**

```
% digest -l
md5
sha1
sha224
sha256
sha384
sha512
```

Nota - Siempre que sea posible, seleccione un algoritmo aprobado por FIPS, de la lista ubicada en "FIPS 140 Algorithms in the Cryptographic Framework" de "Using a FIPS 140 Enabled System in Oracle Solaris 11.2 ".

2. **Calcule el resumen del archivo y guarde la lista de resumen.**

Proporcione un algoritmo con el comando `digest`.

```
% digest -v -a algorithm input-file > digest-listing
```

-v	Muestra el resultado en el siguiente formato: *algorithm* (*input-file*) = *digest*
-a *algorithm*	El algoritmo que se utilizará para calcular un resumen del archivo. Escriba el algoritmo tal como aparece en el resultado del Paso 1.

Nota - Siempre que sea posible, seleccione un algoritmo aprobado por FIPS de la lista ubicada en "FIPS 140 Algorithms in the Cryptographic Framework" de "Using a FIPS 140 Enabled System in Oracle Solaris 11.2 ".

input-file	El archivo de entrada para el comando `digest`.
digest-listing	El archivo de salida para el comando `digest`.

ejemplo 3-4 Cálculo de un resumen con el mecanismo SHA1

En el ejemplo siguiente, el comando `digest` usa el mecanismo SHA1 para proporcionar una lista de directorios. Los resultados se colocarán en un archivo.

```
% digest -v -a sha1 docs/* > $HOME/digest.docs.legal.05.07
% more ~/digest.docs.legal.05.07
```

```
sha1 (docs/legal1) = 1df50e8ad219e34f0b911e097b7b588e31f9b435
sha1 (docs/legal2) = 68efa5a636291bde8f33e046eb33508c94842c38
sha1 (docs/legal3) = 085d991238d61bd0cfa2946c183be8e32cccf6c9
sha1 (docs/legal4) = f3085eae7e2c8d008816564fdf28027d10e1d983
```

▼ Cómo calcular un MAC de un archivo

Un código de autenticación de mensajes, o MAC, calcula un resumen del archivo y utiliza una clave secreta para proteger aún más el resumen. Un MAC no modifica el archivo original.

1. **Muestre los mecanismos disponibles.**

```
% mac -l
Algorithm       Keysize:  Min   Max
---------------------------------
des_mac                   64    64
sha1_hmac                  8   512
md5_hmac                   8   512
sha224_hmac                8   512
sha256_hmac                8   512
sha384_hmac                8  1024
sha512_hmac                8  1024
```

Nota - Cada algoritmo admitido es un alias de la versión más usada y con menos restricciones de un determinado tipo de algoritmo. La salida anterior muestra los nombres de algoritmo disponibles y los tamaños de clave para cada uno. Siempre que sea posible, use un algoritmo admitido que coincida con un algoritmo aprobado por FIPS con la longitud de clave aprobada por FIPS; selecciónelo de la lista ubicada en “FIPS 140 Algorithms in the Cryptographic Framework” de “Using a FIPS 140 Enabled System in Oracle Solaris 11.2 ”.

2. **Genere una clave simétrica de la longitud adecuada.**
Puede proporcionar una frase de contraseña a partir de la cual se generará una clave o puede proporcionar una clave.

   - Si proporciona una frase de contraseña, deberá almacenarla o recordarla. Si almacena la frase de contraseña en línea, sólo usted debe poder leer el archivo de frases de contraseña.
   - Si proporciona una clave, ésta debe ser del tamaño correcto para el mecanismo. Puede utilizar el comando `pktool`. Para conocer el procedimiento y algunos ejemplos, consulte Cómo generar una clave simétrica con el comando `pktool` [22].

3. **Cree un MAC para un archivo.**
Proporcione una clave y utilice un algoritmo de clave simétrico con el comando `mac`.

```
% mac [-v] -a algorithm [-k keyfile | -K key-label [-T token]] input-file
```

-v Muestra el resultado en el siguiente formato:

algorithm (*input-file*) = *mac*

-a *algorithm* — El algoritmo que se utiliza para calcular el MAC. Escriba el algoritmo tal como aparece en el resultado del comando `mac -l`.

-k *keyfile* — El archivo que contiene una clave con la longitud especificada por el algoritmo.

-K *key-label* — Es la etiqueta de la clave en el almacén de claves PKCS #11.

-T *token* — El nombre del token. De manera predeterminada, el token es `Sun Software PKCS#11 softtoken`. Sólo se utiliza cuando la opción -K *key-label* se utiliza.

input-file — El archivo de entrada para el MAC.

ejemplo 3-5 Cálculo de un MAC con SHA1_HMAC y una frase de contraseña

En el siguiente ejemplo, el anexo de correo electrónico se autentica con el mecanismo SHA1_HMAC y una clave que se obtiene a partir de una frase de contraseña. La lista de MAC se guarda en un archivo. Si la frase de contraseña se almacena en un archivo, el usuario debe ser la única persona que pueda leer el archivo.

```
% mac -v -a sha1_hmac email.attach
Enter passphrase:      Type passphrase
sha1_hmac (email.attach) = 2b31536d3b3c0c6b25d653418db8e765e17fe07b
% echo "sha1_hmac (email.attach) = 2b31536d3b3c0c6b25d653418db8e765e17fe07b" \
>> ~/sha1hmac.daily.05.12
```

ejemplo 3-6 Cálculo de un MAC con SHA1_HMAC y un archivo de claves

En el ejemplo siguiente, el manifiesto de directorio se autentica con el mecanismo SHA1_HMAC y una clave secreta. Los resultados se colocarán en un archivo.

```
% mac -v -a sha1_hmac \
-k $HOME/keyf/05.07.mack64 docs/* > $HOME/mac.docs.legal.05.07
% more ~/mac.docs.legal.05.07
sha1_hmac (docs/legal1) = 9b31536d3b3c0c6b25d653418db8e765e17fe07a
sha1_hmac (docs/legal2) = 865af61a3002f8a457462a428cdb1a88c1b51ff5
sha1_hmac (docs/legal3) = 076c944cb2528536c9aebd3b9fbe367e07b61dc7
sha1_hmac (docs/legal4) = 7aede27602ef6e4454748cbd3821e0152e45beb4
```

ejemplo 3-7 Cálculo de un MAC con SHA1_HMAC y una etiqueta de clave

En el ejemplo siguiente, el manifiesto de directorio se autentica con el mecanismo SHA1_HMAC y una clave secreta. Los resultados se ubican en el almacén de claves PKCS #11

del usuario. El usuario creó inicialmente el almacén de claves y la contraseña para el almacén de claves mediante el comando `pktool setpin`.

```
% mac -a sha1_hmac -K legaldocs0507 docs/*
Enter pin for Sun Software PKCS#11 softtoken:     Type password
```

Para recuperar el MAC desde el almacén de claves, el usuario utiliza la opción detallada y proporciona la etiqueta de clave y el nombre del directorio que se ha autenticado.

```
% mac -v -a sha1_hmac -K legaldocs0507  docs/*
Enter pin for Sun Software PKCS#11 softtoken:     Type password
sha1_hmac (docs/legal1) = 9b31536d3b3c0c6b25d653418db8e765e17fe07a
sha1_hmac (docs/legal2) = 865af61a3002f8a457462a428cdb1a88c1b51ff5
sha1_hmac (docs/legal3) = 076c944cb2528536c9aebd3b9fbe367e07b61dc7
sha1_hmac (docs/legal4) = 7aede27602ef6e4454748cbd3821e0152e45beb4
```

▼ Cómo cifrar y descifrar un archivo

Al cifrar un archivo, el archivo original no se elimina ni modifica. Se cifra el archivo de salida.

Para ver las soluciones a los errores comunes relacionados con el comando `encrypt`, consulte sección que aparece a continuación de los ejemplos.

Nota - Cuando el cifrado y el descifrado de archivos, intente usar algoritmos aprobados por FIPS con las longitudes de clave aprobadas, siempre que sea posible. Consulte la lista en "FIPS 140 Algorithms in the Cryptographic Framework" de "Using a FIPS 140 Enabled System in Oracle Solaris 11.2 ". Ejecute el comando `encrypt -l` para ver los algoritmos disponibles y sus longitudes de clave.

---

1. **Cree una clave simétrica de la longitud adecuada.**

   Puede proporcionar una frase de contraseña a partir de la cual se generará una clave o puede proporcionar una clave.

   - Si proporciona una frase contraseña, deberá almacenarla o recordarla. Si almacena la frase de contraseña en línea, sólo usted debe poder leer el archivo de frases de contraseña.
   - Si proporciona una clave, ésta debe ser del tamaño correcto para el mecanismo. Puede utilizar el comando `pktool`. Para conocer el procedimiento y algunos ejemplos, consulte Cómo generar una clave simétrica con el comando `pktool` [22].

2. **Cifre un archivo.**

   Proporcione una clave y utilice un algoritmo de clave simétrico con el comando `encrypt`.

   ```
   % encrypt -a algorithm [-v] \
   [-k keyfile | -K key-label [-T token]] [-i input-file] [-o output-file]
   ```

-a *algorithm*	El algoritmo que se utiliza para cifrar el archivo. Escriba el algoritmo tal como aparece en el resultado del comando `encrypt -l`. Siempre que sea posible, seleccione un algoritmo aprobado por FIPS, de la lista ubicada en "FIPS 140 Algorithms in the Cryptographic Framework" de "Using a FIPS 140 Enabled System in Oracle Solaris 11.2 ".
-k *keyfile*	El archivo que contiene una clave con la longitud especificada por el algoritmo. La longitud de la clave para cada algoritmo se muestra, en bits, en la salida del comando `encrypt -l`.
-K *key-label*	Es la etiqueta de la clave en el almacén de claves PKCS #11.
-T *token*	El nombre del token. De manera predeterminada, el token es `Sun Software PKCS#11 softtoken`. Sólo se utiliza cuando la opción -K *key-label* se utiliza.
-i *input-file*	El archivo de entrada que desea cifrar. Este archivo no es modificado por el comando.
-o *output-file*	El archivo de salida, que es el formato cifrado del archivo de entrada.

ejemplo 3-8 Creación de una clave AES para cifrar los archivos

En el siguiente ejemplo, un usuario crea y almacena una clave AES en un almacén de claves PKCS #11 existente para utilizar en el cifrado y descifrado. El usuario puede comprobar que la clave existe y puede usar la clave, pero no puede verla.

```
% pktool genkey label=MyAESkeynumber1 keytype=aes keylen=256
Enter PIN for Sun Software PKCS#11 softtoken  :     Type password

% pktool list objtype=key
Enter PIN for Sun Software PKCS#11 softtoken  :     Type password
No.      Key Type      Key Len.      Key Label
-----------------------------------------------------
Symmetric keys:
1        AES           256           MyAESkeynumber1
```

Para utilizar la clave para cifrar un archivo, el usuario recupera la clave por su etiqueta.

```
% encrypt -a aes -K MyAESkeynumber1 -i encryptthisfile -o encryptedthisfile
```

Para descifrar el archivo `encryptedthisfile`, el usuario recupera la clave por su etiqueta.

```
% decrypt -a aes -K MyAESkeynumber1 -i encryptedthisfile -o sameasencryptthisfile
```

ejemplo 3-9 Cifrado y descifrado con AES y una frase de contraseña

En el siguiente ejemplo, se cifra un archivo con el algoritmo AES. La clave se genera a partir de una frase de contraseña. Si la frase de contraseña se almacena en un archivo, el usuario debe ser la única persona que pueda leer el archivo.

```
% encrypt -a aes -i ticket.to.ride -o ~/enc/e.ticket.to.ride
Enter passphrase:         Type passphrase
Re-enter passphrase:      Type passphrase again
```

El archivo de entrada, `ticket.to.ride`, todavía existe en su formato original.

Para descifrar el archivo de salida, el usuario utiliza la misma frase de contraseña y el mismo mecanismo de cifrado que utilizó para cifrar el archivo.

```
% decrypt -a aes -i ~/enc/e.ticket.to.ride -o ~/d.ticket.to.ride
Enter passphrase:      Type passphrase
```

ejemplo 3-10 Cifrado y descifrado con AES y un archivo de claves

En el ejemplo siguiente, se cifra un archivo con el algoritmo AES. Los mecanismos AES utilizan una clave de 128 bits o 16 bytes.

```
% encrypt -a aes -k ~/keyf/05.07.aes16 \
-i ticket.to.ride -o ~/enc/e.ticket.to.ride
```

El archivo de entrada, `ticket.to.ride`, todavía existe en su formato original.

Para descifrar el archivo de salida, el usuario utiliza la misma clave y el mismo mecanismo de cifrado que utilizó para cifrar el archivo.

```
% decrypt -a aes -k ~/keyf/05.07.aes16  \
-i ~/enc/e.ticket.to.ride -o ~/d.ticket.to.ride
```

Errores más frecuentes

Los siguientes mensajes indican que el algoritmo que está utilizando no permite la clave que proporcionó para el comando `encrypt`.

- `encrypt: unable to create key for crypto operation: CKR_ATTRIBUTE_VALUE_INVALID`
- `encrypt: failed to initialize crypto operation: CKR_KEY_SIZE_RANGE`

Si utiliza una clave que no cumple con los requisitos del algoritmo, debe proporcionar una clave mejor con uno de los siguientes métodos:

- Utilice una frase de contraseña. La estructura proporciona una clave que cumple con los requisitos.
- Utilice un tamaño de clave que sea aceptado por el algoritmo. Por ejemplo, el algoritmo DES requiere una clave de 64 bits. El algoritmo 3DES requiere una clave de 192 bits.

Administración de la estructura criptográfica

En esta sección se describe cómo administrar proveedores de software y hardware en la estructura criptográfica. Los proveedores de software y hardware se pueden eliminar para no ser utilizados cuando se desee. Por ejemplo, puede desactivar la implementación de un algoritmo de un proveedor de software. A continuación, puede forzar al sistema a utilizar el algoritmo de otro proveedor de software.

Nota - Un componente importante de la administración de la estructura criptográfica es planificar e implementar la política relacionada con FIPS 140, el estándar de seguridad informática del Gobierno de los EE. UU. para módulos de cifrado.

Si tiene un requisito estrictopara usar solo criptografía validada por FIPS 140-2, debe ejecutar la versión Oracle Solaris11.1 SRU5.5 o la versión Oracle Solaris11.1 SRU3. Oracle completó una validación de FIPS 140-2 en la estructura criptográfica de Solaris en estas dos versiones específicas. Oracle Solaris11.2 se crea en esta base validada e incluye mejoras de software que apuntan al rendimiento, la función y la confiabilidad. Siempre que sea posible, debería configurar Oracle Solaris11.2 en el modo FIPS 140-2 para aprovechar estas mejoras.

Revise “Using a FIPS 140 Enabled System in Oracle Solaris 11.2 ” y planifique una política FIPS general para sus sistemas.

En el siguiente mapa de tareas se hace referencia a los procedimientos para administrar a los proveedores de software y hardware en la estructura criptográfica.

TABLA 3-2 Mapa de tareas de la administración de la estructura criptográfica

Tarea	Descripción	Para obtener instrucciones
Planifique su política FIPS para sus sistemas.	Decida sobre su plan para activar los proveedores y consumidores aprobados por FIPS e implemente el plan.	“Using a FIPS 140 Enabled System in Oracle Solaris 11.2 ”
Mostrar los proveedores en la estructura criptográfica.	Muestra los algoritmos, las bibliotecas y los dispositivos de hardware que están disponibles para su uso en la estructura criptográfica.	“Lista de proveedores disponibles” [35]
Active el modo FIPS 140.	Ejecuta la estructura criptográfica para un estándar del gobierno de EE. UU. para módulos de criptografía.	Cómo crear un entorno de inicio con FIPS 140 activado [42]
Agregar un proveedor de software.	Agrega una biblioteca PKCS #11 o un módulo de núcleo a la estructura criptográfica. El proveedor debe estar registrado.	Cómo agregar un proveedor de software [40]
Evitar el uso de un mecanismo de nivel de usuario.	Elimina un mecanismo de software para que no sea utilizado. El mecanismo se puede volver a activar.	Cómo evitar el uso de un mecanismo de nivel de usuario [44]
Desactivar temporalmente mecanismos de un módulo de núcleo.	Elimina temporalmente un mecanismo para que no sea utilizado. Se suele utilizar para realizar pruebas.	Prevención del uso de un mecanismo de software de núcleo [46]

Tarea	Descripción	Para obtener instrucciones
Desinstalar una biblioteca.	Elimina un proveedor de software de nivel de usuario para que no sea utilizado.	Ejemplo 3-17, “Eliminación permanente de una biblioteca de nivel de usuario”
Desinstalar un proveedor del núcleo.	Elimina un proveedor de software de núcleo para que no sea utilizado.	Ejemplo 3-19, “Eliminación temporal de la disponibilidad de un proveedor de software de núcleo”
Desactivar los mecanismos de un proveedor de hardware.	Garantiza que los mecanismos seleccionados en un acelerador de hardware no sean utilizados.	Cómo desactivar funciones y mecanismos del proveedor de hardware [49]
Reiniciar o actualizar los servicios criptográficos.	Garantiza que los servicios criptográficos estén disponibles.	Cómo refrescar o reiniciar todos los servicios criptográficos [51]

Lista de proveedores disponibles

Los proveedores de hardware se ubican y cargan automáticamente. Para obtener más información, consulte la página del comando man `driver.conf`(4).

Cuando cuenta con hardware que piensa conectar a la estructura criptográfica, el hardware se registra con el SPI en el núcleo. La estructura comprueba que el controlador de hardware esté registrado. Específicamente, la estructura comprueba que el archivo de objeto del controlador esté registrado con un certificado emitido por Oracle.

Por ejemplo, la placa Sun Crypto Accelerator 6000 (`mca`), el controlador `ncp` para el acelerador criptográfico en los procesadores UltraSPARC T1 y T2 (`ncp`), el controlador n2cp para los procesadores UltraSPARC T2 (`n2cp`) y el controlador `/dev/crypto` para los sistemas T-Series conectan los mecanismos de hardware a la estructura.

Para obtener información sobre cómo registrar a su proveedor, consulte la información sobre el comando `elfsign` en “Comandos de nivel de usuario de la estructura criptográfica” [12].

Para ver los proveedores disponibles, utilice los comandos `cryptoadm list` con opciones diferentes en función de la información específica que desee obtener.

- Lista de todos los proveedores en el sistema.

  El contenido y el formato de la lista de proveedores varía para las distintas versiones y plataformas de Oracle Solaris. Ejecute el comando `cryptoadm list` en el sistema, para ver los proveedores que admite el sistema. Sólo los mecanismos en el nivel de usuario están disponibles para ser utilizados directamente por los usuarios comunes.

```
% cryptoadm list
User-level providers:       /* for applications */
Provider: /usr/lib/security/$ISA/pkcs11_kernel.so
Provider: /usr/lib/security/$ISA/pkcs11_softtoken.so
Provider: /usr/lib/security/$ISA/pkcs11_tpm.so
```

```
Kernel software providers:        /* for IPsec, kssl, Kerberos */
des
aes
arcfour
blowfish
camellia
ecc
sha1
sha2
md4
md5
rsa
swrand
n2rng/0       /* for hardware */
ncp/0
n2cp/0
```

- Lista de los proveedores y sus mecanismos en la estructura criptográfica.

  Puede ver la potencia y los modos, como `ECB` y `CBC`, de los mecanismos disponibles. Sin embargo, es posible que algunos de los mecanismos de la lista no estén disponibles para su uso. Consulte el siguiente elemento para obtener instrucciones sobre cómo enumerar los mecanismos que se pueden utilizar.

  La siguiente salida se trunca con fines de visualización.

```
% cryptoadm list -m [provider=provider]
User-level providers:
=====================

Provider: /usr/lib/security/$ISA/pkcs11_kernel.so

Mechanisms:
CKM_DSA
CKM_RSA_X_509
CKM_RSA_PKCS
...
CKM_SHA256_HMAC_GENERAL
CKM_SSL3_MD5_MAC

Provider: /usr/lib/security/$ISA/pkcs11_softtoken.so
Mechanisms:
CKM_DES_CBC
CKM_DES_CBC_PAD
CKM_DES_ECB
CKM_DES_KEY_GEN
CKM_DES_MAC_GENERAL
```

```
...
CKM_ECDSA_SHA1
CKM_ECDH1_DERIVE

Provider: /usr/lib/security/$ISA/pkcs11_tpm.so
/usr/lib/security/$ISA/pkcs11_tpm.so: no slots presented.

Kernel providers:
==========================
des: CKM_DES_ECB,CKM_DES_CBC,CKM_DES3_ECB,CKM_DES3_CBC
aes: CKM_AES_ECB,CKM_AES_CBC,CKM_AES_CTR,CKM_AES_CCM, \
     CKM_AES_GCM,CKM_AES_GMAC,
CKM_AES_CFB128,CKM_AES_XTS,CKM_AES_XCBC_MAC
arcfour: CKM_RC4
blowfish: CKM_BLOWFISH_ECB,CKM_BLOWFISH_CBC
ecc: CKM_EC_KEY_PAIR_GEN,CKM_ECDH1_DERIVE,CKM_ECDSA, \
     CKM_ECDSA_SHA1
sha1: CKM_SHA_1,CKM_SHA_1_HMAC,CKM_SHA_1_HMAC_GENERAL
sha2: CKM_SHA224,CKM_SHA224_HMAC,...CKM_SHA512_256_HMAC_GENERAL

md4: CKM_MD4
md5: CKM_MD5,CKM_MD5_HMAC,CKM_MD5_HMAC_GENERAL
rsa: CKM_RSA_PKCS,CKM_RSA_X_509,CKM_MD5_RSA_PKCS, \
     CKM_SHA1_RSA_PKCS,CKM_SHA224_RSA_PKCS,
CKM_SHA256_RSA_PKCS,CKM_SHA384_RSA_PKCS,CKM_SHA512_RSA_PKCS
swrand: No mechanisms presented.
n2rng/0: No mechanisms presented.
ncp/0: CKM_DSA,CKM_RSA_X_509,CKM_RSA_PKCS,CKM_RSA_PKCS_KEY_PAIR_GEN,
CKM_DH_PKCS_KEY_PAIR_GEN,CKM_DH_PKCS_DERIVE,CKM_EC_KEY_PAIR_GEN,
CKM_ECDH1_DERIVE,CKM_ECDSA
n2cp/0: CKM_DES_CBC,CKM_DES_CBC_PAD,CKM_DES_ECB,CKM_DES3_CBC, \
     ...CKM_SSL3_SHA1_MAC
```

- Lista de los mecanismos criptográficos disponibles.

  La política determina qué mecanismos están disponibles para su uso. El administrador define la política. Un administrador puede elegir desactivar los mecanismos de un proveedor determinado. La opción `-p` muestra la lista de los mecanismos permitidos por la política que el administrador ha definido.

  ```
  % cryptoadm list -p [provider=provider]
  User-level providers:
  ====================
  /usr/lib/security/$ISA/pkcs11_kernel.so: \
       all mechanisms are enabled.random is enabled.
  /usr/lib/security/$ISA/pkcs11_softtoken.so: \
       all mechanisms are enabled, random is enabled.
  /usr/lib/security/$ISA/pkcs11_tpm.so: all mechanisms are enabled.
  ```

```
Kernel providers:
=========================
des: all mechanisms are enabled.
aes: all mechanisms are enabled.
arcfour: all mechanisms are enabled.
blowfish: all mechanisms are enabled.
ecc: all mechanisms are enabled.
sha1: all mechanisms are enabled.
sha2: all mechanisms are enabled.
md4: all mechanisms are enabled.
md5: all mechanisms are enabled.
rsa: all mechanisms are enabled.
swrand: random is enabled.
n2rng/0: all mechanisms are enabled. random is enabled.
ncp/0: all mechanisms are enabled.
n2cp/0: all mechanisms are enabled.
```

Los siguientes ejemplos muestran usos específicos adicionales del comando `cryptoadm list`.

EJEMPLO 3-11 Lista de información criptográfica de un proveedor específico

Especificar el proveedor en el comando `cryptoadm` *options* limita la salida sólo a información aplicable al proveedor.

```
# cryptoadm enable provider=dca/0 random
# cryptoadm list -p provider=dca/0
dca/0: all mechanisms are enabled, except CKM_MD5, CKM_MD5_HMAC,...
random is enabled.
```

La siguiente salida muestra sólo que los mecanismos están activados. El generador aleatorio continúa desactivado.

```
# cryptoadm list -p provider=dca/0
dca/0: all mechanisms are enabled, except CKM_MD5,CKM_MD5_HMAC,....

# cryptoadm enable provider=dca/0 mechanism=all
# cryptoadm list -p provider=dca/0
dca/0: all mechanisms are enabled. random is disabled.
```

En la salida siguiente, se muestra que todas las funciones y los mecanismos de la placa están activados.

```
# cryptoadm list -p provider=dca/0
dca/0: all mechanisms ar enabled, except CKM_DES_ECB,CKM_DES3_ECB.
random is disabled.

# cryptoadm enable provider=dca/0 all
# cryptoadm list -p provider=dca/0
dca/0: all mechanisms are enabled. random is enabled.
```

EJEMPLO 3-12 Búsqueda sólo de mecanismos criptográficos de nivel de usuario

En el siguiente ejemplo, se muestran todos los mecanismos que ofrece la biblioteca de nivel de usuario, `pkcs11_softtoken`.

```
% cryptoadm list -m provider=/usr/lib/security/\
   $ISA/pkcs11_softtoken.so
Mechanisms:
CKM_DES_CBC
CKM_DES_CBC_PAD
CKM_DES_ECB
CKM_DES_KEY_GEN
CKM_DES_MAC_GENERAL
CKM_DES_MAC
…
CKM_ECDSA
CKM_ECDSA_SHA1
CKM_ECDH1_DERIVE
```

EJEMPLO 3-13 Determinación de qué mecanismo criptográfico realiza cada función

Los mecanismos realizan funciones criptográficas específicas, como la firma o generación de claves. Las opciones `-v` y `-m` muestran cada mecanismo y sus funciones.

En esta instancia, el administrador desea determinar para qué funciones los mecanismos `CKM_ECDSA*` se pueden utilizar.

```
% cryptoadm list -vm
User-level providers:
=====================
Provider: /usr/lib/security/$ISA/pkcs11_kernel.so
Number of slots: 3
Slot #2
Description: ncp/0 Crypto Accel Asym 1.0
...
CKM_ECDSA                   163  571  X  .  .  .  X  .  X  .  .  .  .  .  .  .

...

Provider: /usr/lib/security/$ISA/pkcs11_softtoken.so
...
CKM_ECDSA       112 571  .  .  .  .  X  .  X  .  .  .  .  .  .  .  .
CKM_ECDSA_SHA1  112 571  .  .  .  .  X  .  X  .  .  .  .  .  .  .  .
...
Kernel providers:
=================
...
ecc: CKM_EC_KEY_PAIR_GEN,CKM_ECDH1_DERIVE,CKM_ECDSA,CKM_ECDSA_SHA1
...
```

La lista indica que estos mecanismos están disponibles de los siguientes proveedores de nivel de usuario:

- `CKM_ECDSA` y `CKM_ECDSA_SHA1`: como una implementación de software en la biblioteca `/usr/lib/security/$ISA/pkcs11_softtoken.so`.
- `CKM_ECDSA`: acelerado por `ncp/0 Crypto Accel Asym 1.0` en la biblioteca `/usr/lib/security/$ISA/pkcs11_kernel.so`.

Cada artículo en una entrada representa una parte de la información sobre el mecanismo. Para estos mecanismos ECC, la lista indica lo siguiente:

- Longitud mínima: 112 bytes.
- Longitud máxima: 571 bytes.
- Hardware: está disponible o no está disponible en el hardware.
- Cifrar: no se utiliza para cifrar datos.
- Descifrar: no se utiliza para descifrar datos.
- Resumir: no se utiliza para crear resúmenes de mensajes.
- Firmar: se utiliza para firmar datos.
- Firmar + recuperar: no se utiliza para firmar datos, donde los datos se pueden recuperar de la firma.
- Verificar: se utiliza para verificar datos firmados.
- Verificar + recuperar: no se utiliza para verificar los datos que se pueden recuperar de la firma.
- Generación de claves: no se utiliza para generar una clave privada.
- Generación de par: no se utiliza para generar un par de claves.
- Ajustar: no se utiliza para ajustar. Es decir, cifrar una clave existente.
- Desajustar: no se utiliza para desajustar una clave ajustada.
- Derivar: no se utiliza para derivar una nueva clave de una clave de base.
- Funciones EC: funciones EC ausentes que no se tratan en elementos anteriores.

Agregación de un proveedor de software

El siguiente procedimiento explica cómo agregar proveedores al sistema. Debe convertirse en un administrador con el perfil de derechos de gestión de criptografía asignado. Para obtener más información, consulte “Uso de sus derechos administrativos asignados” de “Protección de los usuarios y los procesos en Oracle Solaris 11.2 ”.

▼ Cómo agregar un proveedor de software

1. **Muestre los proveedores de software que están disponibles para el sistema.**

```
% cryptoadm list
User-level providers:
Provider: /usr/lib/security/$ISA/pkcs11_kernel.so
Provider: /usr/lib/security/$ISA/pkcs11_softtoken.so
```

```
/usr/lib/security/$ISA/pkcs11_tpm.so: all mechanisms are enabled.

Kernel software providers:
des
aes
arcfour
blowfish
camellia
sha1
sha2
md4
md5
rsa
swrand
n2rng/0
ncp/0
n2cp/0
```

2. **Agregue el proveedor de un repositorio.**
   Oracle emitió un certificado al proveedor de software existente.

3. **Refresque los proveedores.**
   Si agregó un proveedor de software o si agregó hardware y especificó una política para el hardware, debe refrescar los proveedores.

```
# svcadm refresh svc:/system/cryptosvc
```

4. **Ubique al nuevo proveedor en la lista.**
   En este caso, se instaló un nuevo proveedor de software de núcleo.

```
# cryptoadm list
…
Kernel software providers:
des
aes
arcfour
blowfish
camellia
ecc
sha1
sha2
md4
md5
rsa
swrand
sha3      <-- added provider
…
```

ejemplo 3-14 Agregación de un proveedor de software de nivel de usuario

En el ejemplo siguiente, se instala una biblioteca PKCS #11 registrada.

```
# pkgadd -d /cdrom/cdrom0/PKCSNew
     Answer the prompts
# svcadm refresh system/cryptosvc
# cryptoadm list
user-level providers:
=========================
/usr/lib/security/$ISA/pkcs11_kernel.so
/usr/lib/security/$ISA/pkcs11_softtoken.so
/usr/lib/security/$ISA/pkcs11_tpm.so
/opt/lib/$ISA/libpkcs11.so.1      <-- added provider
```

Los desarrolladores que estén probando una biblioteca con la estructura criptográfica pueden instalar la biblioteca manualmente.

```
# cryptoadm install provider=/opt/lib/\$ISA/libpkcs11.so.1
```

Creación de un entorno de inicio con FIPS 140 activado

De manera predeterminada, el modo FIPS 140 está desactivado en Oracle Solaris. En este procedimiento, puede crear un nuevo entorno de inicio (BE) para el modo FIPS 140 y luego activar FIPS 140 en el nuevo entorno de inicio. Al ofrecerle un entorno de inicio de copia de seguridad, este método le permite recuperarse rápidamente de una situación crítica del sistema que puede surgir de pruebas de cumplimiento de FIPS 140.

Para obtener una descripción general sobre FIPS, consulte "Using a FIPS 140 Enabled System in Oracle Solaris 11.2 ". Consulte también la página del comando man `cryptoadm`(1M) y "Estructura criptográfica y FIPS 140" [13].

▼ Cómo crear un entorno de inicio con FIPS 140 activado

Antes de empezar Debe asumir el rol `root`. Para obtener más información, consulte "Uso de sus derechos administrativos asignados" de "Protección de los usuarios y los procesos en Oracle Solaris 11.2 ".

1. Determine si el sistema está en modo FIPS 140.

```
% cryptoadm list fips-140
User-level providers:
=====================
/usr/lib/security/$ISA/pkcs11_softtoken: FIPS-140 mode is disabled.

Kernel software providers:
==========================
des: FIPS-140 mode is disabled.
```

```
aes: FIPS-140 mode is disabled.
ecc: FIPS-140 mode is disabled.
sha1: FIPS-140 mode is disabled.
sha2: FIPS-140 mode is disabled.
rsa: FIPS-140 mode is disabled.
swrand: FIPS-140 mode is disabled.

Kernel hardware providers:
========================:
```

2. **Cree un nuevo entorno de inicio para su versión de FIPS 140 de la estructura criptográfica.**

   Antes de activar el modo FIPS 140, debe primero crear, activar e iniciar un nuevo entorno de inicio utilizando el comando `beadm`. Un sistema compatible con FIPS 140 ejecuta pruebas de cumplimiento que pueden mostrar un aviso grave en caso de error. Por lo tanto, es importante tener un entorno de inicio disponible que pueda iniciar para poner el sistema en funcionamiento mientras depura problemas con el límite FIPS 140.

   a. **Cree un entorno de inicio basado en el entorno de inicio actual.**

      En este ejemplo, se crea un entorno de inicio denominado `S11.1-FIPS`.

      ```
      # beadm create S11.1-FIPS-140
      ```

   b. **Active el entorno de inicio.**

      ```
      # beadm activate S11.1-FIPS-140
      ```

   c. **Reinicie el sistema.**

   d. **Active el modo FIPS 140 en el nuevo entorno de inicio.**

      ```
      # cryptoadm enable fips-140
      ```

   Nota - Este subcomando no desactiva los algoritmos aprobados que no pertenecen a FIPS 140 de la biblioteca `pkcs11_softtoken` de nivel de usuario y de los proveedores de software de núcleo. Los consumidores de la estructura son responsables del uso de algoritmos aprobados por FIPS 140.

   Para obtener más información sobre los efectos del modo FIPS 140, consulte “Using a FIPS 140 Enabled System in Oracle Solaris 11.2 ”. Consulte también la página del comando man `cryptoadm`(1M).

3. **Cuando desee trabajar sin FIPS 140 activado, desactívelo.**

   Puede reiniciar el entorno de inicio original o desactivar FIPS 140 en el entorno de inicio actual.

   - **Inicie el entorno de inicio original.**

```
# beadm list
BE               Active Mountpoint Space   Policy Created
--               ------ ---------- -----   ------ -------
S11.1            -      -          48.22G   static 2012-10-10 10:10
S11.1-FIPS-140   NR     /          287.01M  static 2012-11-18 18:18
# beadm activate S11.1
# beadm list
BE               Active Mountpoint Space   Policy Created
--               ------ ---------- -----   ------ -------
S11.1            R      -          48.22G   static 2012-10-10 10:10
S11.1-FIPS-140   N      /          287.01M  static 2012-11-18 18:18
# reboot
```

- **Desactive el modo FIPS 140 en el entorno de inicio actual y reinicie el sistema.**

  ```
  # cryptoadm disable fips-140
  ```

  El modo FIPS 140 permanece en funcionamiento hasta que se reinicie el sistema.

  ```
  # reboot
  ```

Prevención del uso de mecanismos

Si algunos de los mecanismos criptográficos de un proveedor de biblioteca no se debe utilizar, puede eliminar los mecanismos seleccionados. Puede considerar impedir el uso de mecanismos si, por ejemplo, el mismo mecanismo de otra biblioteca funciona mejor o si una vulnerabilidad de seguridad se está investigando.

Si la estructura criptográfica proporciona múltiples modos de un proveedor como AES, puede eliminar un mecanismo lento para no utilizarlo o un mecanismo dañado. También puede utilizar este procedimiento para eliminar un algoritmo con vulnerabilidades de seguridad probadas.

Puede desactivar de manera selectiva los mecanismos y la función de números aleatorios de un proveedor de hardware. Para activarlos nuevamente, consulte Ejemplo 3-22, “Activación de mecanismos y funciones en un proveedor de hardware”. El hardware de este ejemplo, la placa Crypto Accelerator 1000 de Sun, proporciona un generador de números aleatorios.

▼ Cómo evitar el uso de un mecanismo de nivel de usuario

Antes de empezar Debe convertirse en un administrador con el perfil de derechos de gestión de criptografía asignado. Para obtener más información, consulte “Uso de sus derechos administrativos asignados” de “Protección de los usuarios y los procesos en Oracle Solaris 11.2 ”.

1. **Muestre los mecanismos ofrecidos por un proveedor de software de nivel de usuario determinado.**

```
% cryptoadm list -m provider=/usr/lib/security/\$ISA/pkcs11_softtoken.so
/usr/lib/security/$ISA/pkcs11_softtoken.so:
CKM_DES_CBC,CKM_DES_CBC_PAD,CKM_DES_ECB,CKM_DES_KEY_GEN,
CKM_DES3_CBC,CKM_DES3_CBC_PAD,CKM_DES3_ECB,CKM_DES3_KEY_GEN,
CKM_AES_CBC,CKM_AES_CBC_PAD,CKM_AES_ECB,CKM_AES_KEY_GEN,
…
```

2. **Muestre los mecanismos que están disponibles para su uso.**

```
$ cryptoadm list -p
user-level providers:
=====================
…
/usr/lib/security/$ISA/pkcs11_softtoken.so: all mechanisms are enabled.
random is enabled.
…
```

3. **Desactive los mecanismos que no se deben utilizar.**

```
$ cryptoadm disable provider=/usr/lib/security/\$ISA/pkcs11_softtoken.so \
> mechanism=CKM_DES_CBC,CKM_DES_CBC_PAD,CKM_DES_ECB
```

4. **Muestre los mecanismos que están disponibles para su uso.**

```
$ cryptoadm list -p provider=/usr/lib/security/\$ISA/pkcs11_softtoken.so
/usr/lib/security/$ISA/pkcs11_softtoken.so: all mechanisms are enabled,
except CKM_DES_ECB,CKM_DES_CBC_PAD,CKM_DES_CBC. random is enabled.
```

ejemplo 3-15 Activación de un mecanismo de proveedor de software de nivel de usuario

En el ejemplo siguiente, un mecanismo DES desactivado se vuelve a poner a disposición para su uso.

```
$ cryptoadm list -m provider=/usr/lib/security/\$ISA/pkcs11_softtoken.so
/usr/lib/security/$ISA/pkcs11_softtoken.so:
CKM_DES_CBC,CKM_DES_CBC_PAD,CKM_DES_ECB,CKM_DES_KEY_GEN,
CKM_DES3_CBC,CKM_DES3_CBC_PAD,CKM_DES3_ECB,CKM_DES3_KEY_GEN,
…
$ cryptoadm list -p provider=/usr/lib/security/\$ISA/pkcs11_softtoken.so
/usr/lib/security/$ISA/pkcs11_softtoken.so: all mechanisms are enabled,
except CKM_DES_ECB,CKM_DES_CBC_PAD,CKM_DES_CBC. random is enabled.
$ cryptoadm enable provider=/usr/lib/security/\$ISA/pkcs11_softtoken.so \
> mechanism=CKM_DES_ECB
$ cryptoadm list -p provider=/usr/lib/security/\$ISA/pkcs11_softtoken.so
/usr/lib/security/$ISA/pkcs11_softtoken.so: all mechanisms are enabled,
except CKM_DES_CBC_PAD,CKM_DES_CBC. random is enabled.
```

ejemplo 3-16 Activación de todos los mecanismos de proveedor de software de nivel de usuario

En el siguiente ejemplo, todos los mecanismos de la biblioteca de nivel de usuario están activados.

```
$ cryptoadm enable provider=/usr/lib/security/\$ISA/pkcs11_softtoken.so all
$ cryptoadm list -p provider=/usr/lib/security/\$ISA/pkcs11_softtoken.so
/usr/lib/security/$ISA/pkcs11_softtoken.so: all mechanisms are enabled.
random is enabled.
```

ejemplo 3-17 Eliminación permanente de una biblioteca de nivel de usuario

En el siguiente ejemplo, una biblioteca `libpkcs11.so.1` se elimina del directorio `/opt`.

```
$ cryptoadm uninstall provider=/opt/lib/\$ISA/libpkcs11.so.1
$ cryptoadm list
user-level providers:
/usr/lib/security/$ISA/pkcs11_kernel.so
/usr/lib/security/$ISA/pkcs11_softtoken.so
/usr/lib/security/$ISA/pkcs11_tpm.so

kernel providers:
…
```

▼ Prevención del uso de un mecanismo de software de núcleo

Antes de empezar Debe convertirse en un administrador con el perfil de derechos de gestión de criptografía asignado. Para obtener más información, consulte “Uso de sus derechos administrativos asignados” de “Protección de los usuarios y los procesos en Oracle Solaris 11.2 ”.

1. **Muestre los mecanismos ofrecidos por un proveedor de software de núcleo determinado.**

```
$ cryptoadm list -m provider=aes
aes: CKM_AES_ECB,CKM_AES_CBC,CKM_AES_CTR,CKM_AES_CCM,CKM_AES_GCM,
CKM_AES_GMAC,CKM_AES_CFB128,CKM_AES_XTS,CKM_AES_XCBC_MAC
```

2. **Muestre los mecanismos que están disponibles para su uso.**

```
$ cryptoadm list -p provider=aes
aes: all mechanisms are enabled.
```

3. **Desactive los mecanismos que no se deben utilizar.**

```
$ cryptoadm disable provider=aes mechanism=CKM_AES_ECB
```

4. **Muestre los mecanismos que están disponibles para su uso.**

```
$ cryptoadm list -p provider=aes
aes: all mechanisms are enabled, except CKM_AES_ECB.
```

ejemplo 3-18 Activación de un mecanismo de proveedor de software de núcleo

En el ejemplo siguiente, un mecanismo AES desactivado se vuelve a poner a disposición para su uso.

```
cryptoadm list -m provider=aes
aes: CKM_AES_ECB,CKM_AES_CBC,CKM_AES_CTR,CKM_AES_CCM,
CKM_AES_GCM,CKM_AES_GMAC,CKM_AES_CFB128,CKM_AES_XTS,CKM_AES_XCBC_MAC
$ cryptoadm list -p provider=aes
aes: all mechanisms are enabled, except CKM_AES_ECB.
$ cryptoadm enable provider=aes mechanism=CKM_AES_ECB
$ cryptoadm list -p provider=aes
aes: all mechanisms are enabled.
```

ejemplo 3-19 Eliminación temporal de la disponibilidad de un proveedor de software de núcleo

En el siguiente ejemplo, se elimina temporalmente el proveedor AES para no utilizarlo. El subcomando `unload` es útil para evitar que un proveedor se cargue automáticamente mientras el proveedor se está desinstalando. Por ejemplo, se podría utilizar el subcomando `unload` al modificar un mecanismo de este proveedor.

```
$ cryptoadm unload provider=aes

$ cryptoadm list
…
Kernel software providers:
des
aes (inactive)
arcfour
blowfish
ecc
sha1
sha2
md4
md5
rsa
swrand
n2rng/0
ncp/0
n2cp/0
```

El proveedor AES no estará disponible hasta que la estructura criptográfica se haya refrescado.

```
$ svcadm refresh system/cryptosvc

$ cryptoadm list
…
Kernel software providers:
des
aes
arcfour
blowfish
```

```
camellia
ecc
sha1
sha2
md4
md5
rsa
swrand
n2rng/0
ncp/0
n2cp/0
```

Si un consumidor de núcleo está utilizando el proveedor de software de núcleo, el software no se descarga. Se muestra un mensaje de error y el proveedor sigue estando disponible para su uso.

ejemplo 3-20 Eliminación permanente de la disponibilidad de un proveedor de software

En el siguiente ejemplo, se elimina el proveedor AES para no utilizarlo. Una vez eliminado, el proveedor AES no aparece en la lista de la política de los proveedores de software de núcleo.

```
$ cryptoadm uninstall provider=aes

$ cryptoadm list
…
Kernel software providers:
des
arcfour
blowfish
camellia
ecc
sha1
sha2
md4
md5
rsa
swrand
n2rng/0
ncp/0
n2cp/0
```

Si el consumidor de núcleo está utilizando el proveedor de software de núcleo, se muestra un mensaje de error y el proveedor sigue estando disponible para su uso.

ejemplo 3-21 Reinstalación de un proveedor de software de núcleo eliminado

En el siguiente ejemplo, se reinstala el proveedor de software de núcleo AES. Para reinstalar un proveedor de núcleo que fue eliminado, debe enumerar los mecanismos que se instalarán.

```
$ cryptoadm install provider=aes \
mechanism=CKM_AES_ECB,CKM_AES_CBC,CKM_AES_CTR,CKM_AES_CCM,
CKM_AES_GCM,CKM_AES_GMAC,CKM_AES_CFB128,CKM_AES_XTS,CKM_AES_XCBC_MAC
```

```
$ cryptoadm list
…
Kernel software providers:
des
aes
arcfour
blowfish
camellia
ecc
sha1
sha2
md4
md5
rsa
swrand
n2rng/0
ncp/0
n2cp/0
```

▼ Cómo desactivar funciones y mecanismos del proveedor de hardware

Antes de empezar Debe convertirse en un administrador con el perfil de derechos de gestión de criptografía asignado. Para obtener más información, consulte “Uso de sus derechos administrativos asignados” de “Protección de los usuarios y los procesos en Oracle Solaris 11.2 ”.

- **Seleccione los mecanismos o la función que desea desactivar.**

  Muestre el proveedor de hardware.

  ```
  # cryptoadm list
  ...
  Kernel hardware providers:
  dca/0
  ```

  - **Desactive los mecanismos seleccionados.**

    ```
    # cryptoadm list -m provider=dca/0
    dca/0: CKM_RSA_PKCS, CKM_RSA_X_509, CKM_DSA, CKM_DES_CBC, CKM_DES3_CBC
    random is enabled.
    # cryptoadm disable provider=dca/0 mechanism=CKM_DES_CBC,CKM_DES3_CBC
    # cryptoadm list -p provider=dca/0
    dca/0: all mechanisms are enabled except CKM_DES_CBC,CKM_DES3_CBC.
    random is enabled.
    ```

  - **Desactive el generador de números aleatorios.**

    ```
    # cryptoadm list -p provider=dca/0
    dca/0: all mechanisms are enabled. random is enabled.
    # cryptoadm disable provider=dca/0 random
    # cryptoadm list -p provider=dca/0
    ```

```
dca/0: all mechanisms are enabled. random is disabled.
```

- **Desactive todos los mecanismos. No desactive el generador de números aleatorios.**

```
# cryptoadm list -p provider=dca/0
dca/0: all mechanisms are enabled. random is enabled.
# cryptoadm disable provider=dca/0 mechanism=all
# cryptoadm list -p provider=dca/0
dca/0: all mechanisms are disabled. random is enabled.
```

- **Desactive todas las funciones y los mecanismos en el hardware.**

```
# cryptoadm list -p provider=dca/0
dca/0: all mechanisms are enabled. random is enabled.
# cryptoadm disable provider=dca/0 all
# cryptoadm list -p provider=dca/0
dca/0: all mechanisms are disabled. random is disabled.
```

ejemplo 3-22 Activación de mecanismos y funciones en un proveedor de hardware

En los siguientes ejemplos, los mecanismos desactivados en una herramienta de hardware se activan de manera selectiva.

```
# cryptoadm list -p provider=dca/0
dca/0: all mechanisms are enabled except CKM_DES_ECB,CKM_DES3_ECB

.
random is enabled.
# cryptoadm enable provider=dca/0 mechanism=CKM_DES3_ECB
# cryptoadm list -p provider=dca/0
dca/0: all mechanisms are enabled except CKM_DES_ECB.
random is enabled.
```

En el ejemplo siguiente, sólo se activa el generador aleatorio.

```
# cryptoadm list -p provider=dca/0
dca/0: all mechanisms are enabled, except CKM_MD5,CKM_MD5_HMAC,….
random is disabled.
# cryptoadm enable provider=dca/0 random
# cryptoadm list -p provider=dca/0
dca/0: all mechanisms are enabled, except CKM_MD5,CKM_MD5_HMAC,….
random is enabled.
```

En el ejemplo siguiente, sólo se activan los mecanismos. El generador aleatorio continúa desactivado.

```
# cryptoadm list -p provider=dca/0
dca/0: all mechanisms are enabled, except CKM_MD5,CKM_MD5_HMAC,….
random is disabled.
# cryptoadm enable provider=dca/0 mechanism=all
# cryptoadm list -p provider=dca/0
```

```
dca/0: all mechanisms are enabled. random is disabled.
```

En el ejemplo siguiente, se activan todas las funciones y los mecanismos de la placa.

```
# cryptoadm list -p provider=dca/0
dca/0: all mechanisms are enabled, except CKM_DES_ECB,CKM_DES3_ECB.
random is disabled.
# cryptoadm enable provider=dca/0 all
# cryptoadm list -p provider=dca/0
dca/0: all mechanisms are enabled. random is enabled.
```

Refrescamiento o reinicio de todos los servicios criptográficos

De manera predeterminada, la estructura criptográfica está activada. Cuando el daemon `kcfd` falla por cualquier motivo, la utilidad de gestión de servicios (SMF) se puede utilizar para reiniciar los servicios criptográficos. Para obtener más información, consulte las páginas del comando man `smf`(5) y `svcadm`(1M). Para ver el efecto del reinicio de servicios criptográficos en las zonas, consulte “Zonas y servicios criptográficos” [13].

▼ Cómo refrescar o reiniciar todos los servicios criptográficos

Antes de empezar Debe convertirse en un administrador con el perfil de derechos de gestión de criptografía asignado. Para obtener más información, consulte “Uso de sus derechos administrativos asignados” de “Protección de los usuarios y los procesos en Oracle Solaris 11.2 ”.

1. **Compruebe el estado de los servicios criptográficos.**

```
% svcs cryptosvc
STATE          STIME    FMRI
offline         Dec_09   svc:/system/cryptosvc:default
```

2. **Active los servicios criptográficos.**

```
# svcadm enable svc:/system/cryptosvc
```

ejemplo 3-23 Refrescamiento de los servicios criptográficos

En el siguiente ejemplo, se actualizan los servicios criptográficos en la zona global. Por lo tanto, también se actualiza la política criptográfica de nivel de núcleo de cada zona no global.

```
# svcadm refresh system/cryptosvc
```

♦ ♦ ♦ CAPÍTULO 4

Estructura de gestión de claves

La estructura de gestión de claves (KMF) de Oracle Solaris proporciona herramientas e interfaces de programación para gestionar objetos de clave pública. Los objetos de clave pública incluyen certificados X. 509 y pares de claves públicas o privadas. Los formatos para almacenar estos objetos pueden variar. KMF también proporciona una herramienta para administrar políticas que definan el uso de certificados X. 509 por parte de las aplicaciones. KMF admite complementos de terceros

En este capítulo, se tratan los siguientes temas:

Administración de tecnologías de clave pública

KMF centraliza la gestión de tecnologías de clave pública (PKI). Oracle Solaris tiene varias aplicaciones que utilizan tecnologías PKI. Cada aplicación proporciona su propias interfaces de programación, mecanismos de almacenamiento de claves y utilidades administrativas. Si una aplicación proporciona un mecanismo de cumplimiento de políticas, el mecanismo se aplica sólo a esa aplicación. Con KMF, las aplicaciones utilizan un conjunto unificado de herramientas administrativas, un conjunto único de interfaces de programación y un mecanismo único de cumplimiento de políticas. Estas funciones gestionan las necesidades de PKI de todas las aplicaciones que adoptan estas interfaces.

KMF unifica la gestión de tecnologías de clave pública con las siguientes interfaces:

- Comando `pktool`: administra objetos PKI, como certificados, en una variedad de almacenes de claves.
- Comando `kmfcfg`: gestiona la base de datos de políticas PKI y complementos de terceros.

  Las decisiones de políticas PKI incluyen operaciones, como el método de validación para una operación. Además, una política PKI puede limitar el ámbito de un certificado. Por ejemplo, una política PKI puede afirmar que un certificado sólo se puede utilizar para fines específicos. Una política de ese tipo puede impedir que ese certificado se utilice para otras solicitudes.

- Biblioteca KMF: contiene interfaces de programación que abstraen el mecanismo subyacente de almacenes de claves.

  Las aplicaciones no tienen que elegir un determinado mecanismo de almacenes de claves, sino que pueden migrar de un mecanismo a otro. Los almacenes de claves admitidos son PKCS #11, NSS y OpenSSL. La biblioteca incluye una estructura conectable de modo que puedan agregarse mecanismos de almacenes de claves nuevos. Por lo tanto, las aplicaciones que utilizan los mecanismos nuevos requerirían sólo pequeñas modificaciones para poder utilizar un almacén de claves nuevo.

Utilidades de la estructura de gestión de claves

KMF proporciona métodos para administrar el almacenamiento de claves y proporciona la política global para utilizar esas claves. KMF puede gestionar la política, las claves y los certificados para tres tecnologías de clave pública:

- Tokens de los proveedores PKCS #11, es decir, de la estructura criptográfica
- NSS, es decir, servicios de seguridad de red
- OpenSSL, un almacén de claves basado en archivos

La herramienta `kmfcfg` puede crear, modificar o suprimir entradas de políticas KMF. La herramienta también gestiona complementos para la estructura. KMF administra almacenes de claves a través del comando `pktool`. Para obtener más información, consulte las páginas del comando man `kmfcfg`(1) and `pktool`(1) y las siguientes secciones.

Gestión de políticas KMF

La política KMF se almacena en una base de datos. Todas las aplicaciones que utilizan las interfaces de programación KMF acceden internamente a esta base de datos de políticas. La base de datos puede restringir el uso de las claves y los certificados administrados por la biblioteca KMF. Cuando una aplicación intenta verificar un certificado, la aplicación comprueba la base de datos de políticas. El comando `kmfcfg` modifica la base de datos de políticas.

Gestión de complementos KMF

El comando `kmfcfg` proporciona los siguientes subcomandos para complementos:

- `list plugin`: enumera complementos gestionados por KMF.

- `install` *plugin*: instala el complemento por nombre de ruta del módulo y crea un almacén de claves para el complemento. Para eliminar el complemento de KMF, elimine el almacén de claves.
- `uninstall` *plugin*: elimina el complemento de KMF eliminando su almacén de claves.
- `modify` *plugin*: permite que se ejecute el complemento con una opción definida en el código para el complemento, como `debug`.

Para obtener más información, consulte la página del comando man `kmfcfg`(1). Para conocer el procedimiento, consulte Cómo gestionar complementos de terceros en KMF [69].

Gestión de almacenes de claves KMF

KMF gestiona los almacenes de claves para tres tecnologías de clave pública: tokens PKCS #11, NSS y OpenSSL. Para todas estas tecnologías, el comando `pktool` permite realizar las tareas siguientes:

- Generar un certificado autofirmado
- Generar una solicitud de certificado
- Generar una clave simétrica
- Generar un par de claves públicas/privadas
- Generar una solicitud de firma de certificado (CSR) PKCS #10 para que se envíe a una autoridad de certificado (CA) para que la firme
- Firmar una solicitud de firma de certificado PKCS #10
- Importar objetos al almacén de claves
- Enumerar los objetos del almacén de claves
- Suprimir objetos del almacén de claves
- Descargar una CRL

Para las tecnologías PKCS #11 y NSS, el comando `pktool` también permite definir un PIN generando una frase de contraseña para el almacén de claves o para un objeto en él.

Para ver ejemplos de uso de la utilidad `pktool`, consulte la página del comando man `pktool`(1) y Tabla 4-1, “Mapa de tareas del uso de la estructura de gestión de claves”.

Uso de la estructura de gestión de claves

En esta sección, se describe cómo utilizar el comando `pktool` para gestionar los objetos de clave pública, como contraseñas, frases de contraseña, archivos, almacenes de claves, certificados y CRL.

La estructura de gestión de claves (KMF) permite gestionar de manera centralizada las tecnologías de clave pública.

TABLA 4-1 Mapa de tareas del uso de la estructura de gestión de claves

Tarea	Descripción	Para obtener instrucciones
Crear un certificado.	Crea un certificado para uso de PKCS #11, NSS u OpenSSL.	Cómo crear un certificado mediante el comando `pktool gencert` [56]
Exportar un certificado.	Crea un archivo con el certificado y sus claves admitidas. El archivo puede protegerse con una contraseña.	Cómo exportar un certificado y una clave privada en formato PKCS #12 [60]
Importar un certificado.	Importa un certificado desde otro sistema.	Cómo importar un certificado al almacén de claves [58]
	Importa un certificado en formato PKCS #12 desde otro sistema.	Ejemplo 4-2, "Importación de un archivo PKCS #12 al almacén de claves"
Generar una frase de contraseña.	Genera una frase de contraseña para acceder a un almacén de claves PKCS #11 o a un almacén de claves NSS.	Cómo generar una frase de contraseña mediante el comando `pktool setpin` [61]
Generar una clave simétrica.	Genera claves simétricas para utilizar en el cifrado de archivos, la creación de un MAC de un archivo, y para aplicaciones.	Cómo generar una clave simétrica con el comando `pktool` [22]
Generar un par de claves.	Genera un par de claves públicas/privadas para utilizar con aplicaciones.	Cómo generar un par de claves utilizando el comando `pktool genkeypair` [63]
Generar una solicitud de firma de certificado PKCS #10.	Genera una solicitud de firma de certificado (CSR) PKCS #10 para que una autoridad de certificación (CA) externa la firme.	Página del comando man `pktool`(1)
Firmar una solicitud de firma de certificado PKCS #10.	Firma una solicitud de firma de certificado PKCS #10.	Cómo firmar una solicitud de certificación utilizando el comando `pktool signcsr` [67]
Agregar un complemento a KMF.	Instala, modifica y muestra un complemento. También, elimina el complemento de la KMF.	Cómo gestionar complementos de terceros en KMF [69]

▼ Cómo crear un certificado mediante el comando `pktool gencert`

Este procedimiento crea un certificado autofirmado y almacena el certificado en el almacén de claves PKCS #11. Como parte de esta operación, también se crea un par de claves RSA públicas/privadas. La clave privada está almacenada en el almacén de claves con el certificado.

1. **Genere un certificado autofirmado.**

```
% pktool gencert [keystore=keystore] label=label-name \
subject=subject-DN serial=hex-serial-number keytype=rsa/dsa keylen=key-size
```

`keystore=`*keystore*	Especifica el almacén de claves por tipo de objeto de clave pública. El valor puede ser `nss`, `pkcs11` o `file`. La palabra clave es opcional.
`label=`*label-name*	Especifica un nombre único que el emisor asigna al certificado.
`subject=`*subject-DN*	Especifica el nombre distintivo para el certificado.
`serial=`*hex-serial-number*	Especifica el número de serie en formato hexadecimal. El emisor del certificado elige el número, como `0x0102030405`.
`keytype=`*tipo de clave*	Variable opcional que especifica el tipo de clave privada asociada con el certificado. Consulte la página del comando man `pktool`(1) para encontrar los tipos de clave disponibles para el almacén de claves seleccionado. Para usar una clave aprobada FIPS 140, consulte los tipos de claves aprobadas en "FIPS 140 Algorithms in the Cryptographic Framework" de "Using a FIPS 140 Enabled System in Oracle Solaris 11.2 ".
`keylen=`*tamaño de clave*	Variable opcional que especifica la longitud de la clave privada asociada con el certificado. Para usar una clave aprobada FIPS 140, compruebe las longitudes de clave aprobadas para el tipo de clave que seleccionó en "FIPS 140 Algorithms in the Cryptographic Framework" de "Using a FIPS 140 Enabled System in Oracle Solaris 11.2 ".

2. Verifique el contenido del almacén de claves.

```
% pktool list
Found number certificates.
1. (X.509 certificate)
Label:  label-name
ID: fingerprint that binds certificate to private key
Subject: subject-DN
Issuer:  distinguished-name
Serial:  hex-serial-number
n. ...
```

Este comando muestra todos los certificados del almacén de claves. En el ejemplo siguiente, el almacén de claves contiene un solo certificado.

ejemplo 4-1 Creación de un certificado autofirmado mediante `pktool`

En el ejemplo siguiente, un usuario de `My Company` crea un certificado autofirmado y almacena el certificado en un almacén de claves para objetos PKCS #11. El almacén de claves está vacío inicialmente. Si el almacén de claves no fue inicializado, el PIN para el softtoken es `changeme` y puede usar el comando `pktool setpin` para restablecer el PIN. Tenga en cuenta que el tipo de clave aprobada FIPS y la longitud de la clave, RSA 2048, se especifica en las opciones del comando.

```
% pktool gencert keystore=pkcs11 label="My Cert" \
subject="C=US, O=My Company, OU=Security Engineering Group, CN=MyCA" \
serial=0x000000001 keytype=rsa keylen=2048
Enter pin for Sun Software PKCS#11 softtoken:     Type PIN for token

% pktool list
No.  Key Type  Key Len.  Key Label
-----------------------------------------------------
Asymmetric public keys:
1    RSA                My Cert
Certificates:
1    X.509 certificate
Label: My Cert
ID: d2:7e:20:04:a5:66:e6:31:90:d8:53:28:bc:ef:55:55:dc:a3:69:93
Subject: C=US, O=My Company, OU=Security Engineering Group, CN=MyCA
Issuer: C=US, O=My Company, OU=Security Engineering Group, CN=MyCA
...
...
Serial: 0x00000010
...
```

▼ Cómo importar un certificado al almacén de claves

Este procedimiento describe cómo importar al almacén de claves un archivo con información PKI que se codifica con PEM o con DER sin procesar. Para un procedimiento de exportación, consulte el Ejemplo 4-4, “Exportación de un certificado y una clave privada en formato PKCS #12”.

1. **Importe el certificado.**

   ```
   % pktool import keystore=keystore infile=infile-name label=label-name
   ```

2. **Si va a importar objetos PKI privados, proporcione contraseñas cuando se le solicite.**

   a. **En el indicador, escriba la contraseña para el archivo.**

Si está importando información PKI que es privada, como un archivo de exportación en formato PKCS #12, el archivo requiere una contraseña. El creador del archivo que está importando proporciona la contraseña PKCS #12.

```
Enter password to use for accessing the PKCS12 file:     Type PKCS #12 password
```

b. En la petición de datos, escriba la contraseña para el almacén de claves.

```
Enter pin for Sun Software PKCS#11 softtoken:     Type PIN for token
```

3. Verifique el contenido del almacén de claves.

```
% pktool list
Found number certificates.
1. (X.509 certificate)
Label:  label-name
ID: fingerprint that binds certificate to private key
Subject: subject-DN
Issuer:  distinguished-name
Serial:  hex-serial-number

2. ...
```

ejemplo 4-2 Importación de un archivo PKCS #12 al almacén de claves

En el ejemplo siguiente, el usuario importa un archivo PKCS #12 de terceros. El comando `pktool import` extrae la clave privada y el certificado del archivo `gracedata.p12`, y los almacena en el almacén de claves preferido del usuario.

```
% pktool import keystore=pkcs11 infile=gracedata.p12 label=GraceCert
Enter password to use for accessing the PKCS12 file:     Type PKCS #12 password
Enter pin for Sun Software PKCS#11 softtoken:     Type PIN for token
Found 1 certificate(s) and 1 key(s) in gracedata.p12
% pktool list
No.  Key Type  Key Len.  Key Label
-----------------------------------------------------
Asymmetric public keys:
1    RSA                GraceCert
Certificates:
1    X.509 certificate
Label: GraceCert
ID: 71:8f:11:f5:62:10:35:c2:5d:b4:31:38:96:04:80:25:2e:ad:71:b3
Subject: C=US, O=My Company, OU=Security Engineering Group, CN=MyCA
Issuer: C=US, O=My Company, OU=Security Engineering Group, CN=MyCA
Serial: 0x00000010
```

ejemplo 4-3 Importación de un certificado X.509 al almacén de claves

En el ejemplo siguiente, el usuario importa un certificado X.509 en formato PEM al almacén de claves preferido del usuario. Este certificado público no está protegido con una contraseña. El almacén de claves del usuario tampoco está protegido con una contraseña.

```
% pktool import keystore=pkcs11 infile=somecert.pem label="TheirCompany Root Cert"
% pktool list
No.  Key Type  Key Len.  Key Label
Certificates:
1    X.509 certificate
Label: TheirCompany Root Cert
ID: ec:a2:58:af:83:b9:30:9d:de:b2:06:62:46:a7:34:49:f1:39:00:0e
Subject: C=US, O=TheirCompany, OU=Security, CN=TheirCompany Root CA
Issuer: C=US, O=TheirCompany, OU=Security, CN=TheirCompany Root CA
Serial: 0x00000001
```

▼ Cómo exportar un certificado y una clave privada en formato PKCS #12

Puede crear un archivo en formato PKCS #12 para exportar a otros sistemas las claves privadas y su certificado X.509 asociado. El acceso al archivo está protegido con una contraseña.

1. Encuentre el certificado para exportar.

```
% pktool list
Found number certificates.
1. (X.509 certificate)
Label:  label-name
ID: fingerprint that binds certificate to private key
Subject: subject-DN
Issuer:  distinguished-name
Serial:  hex-serial-number

2. ...
```

2. Exporte las claves y el certificado.

Utilice el almacén de claves y la etiqueta del comando `pktool list`. Proporcione un nombre de archivo para el archivo de exportación. Si el nombre contiene un espacio, escríbalo entre comillas dobles.

```
% pktool export keystore=keystore outfile=outfile-name label=label-name
```

3. Proteja el archivo de exportación con una contraseña.

En la petición de datos, escriba la contraseña actual para el almacén de claves. En este punto, puede crear una contraseña para el archivo de exportación. El destinatario debe proporcionar esta contraseña al importar el archivo.

```
Enter pin for Sun Software PKCS#11 softtoken:      Type PIN for token
Enter password to use for accessing the PKCS12 file:      Create PKCS #12 password
```

Sugerencia - Envíe la contraseña por separado del archivo de exportación. De acuerdo con las prácticas recomendadas, es aconsejable proporcionar la contraseña fuera de banda, por ejemplo, durante una llamada telefónica.

ejemplo 4-4 Exportación de un certificado y una clave privada en formato PKCS #12

En el ejemplo siguiente, un usuario exporta a un archivo PKCS #12 estándar las claves privadas con su certificado X.509 asociado. Este archivo se puede importar a otros almacenes de claves. La contraseña PKCS #11 protege el almacén de claves de origen. La contraseña PKCS #12 se utiliza para proteger los datos privados en el archivo PKCS #12. Esta contraseña es necesaria para importar el archivo.

```
% pktool list
No.  Key Type  Key Len.  Key Label
-------------------------------------------------
Asymmetric public keys:
1    RSA                My Cert
Certificates:
1    X.509 certificate
Label: My Cert
ID: d2:7e:20:04:a5:66:e6:31:90:d8:53:28:bc:ef:55:55:dc:a3:69:93
Subject: C=US, O=My Company, OU=Security Engineering Group, CN=MyCA
Issuer: C=US, O=My Company, OU=Security Engineering Group, CN=MyCA
Serial: 0x000001

% pktool export keystore=pkcs11 outfile=mydata.p12 label="My Cert"
Enter pin for Sun Software PKCS#11 softtoken:      Type PIN for token
Enter password to use for accessing the PKCS12 file:     Create PKCS #12 password
```

A continuación, el usuario llama por teléfono al destinatario y proporciona la contraseña PKCS #12.

▼ Cómo generar una frase de contraseña mediante el comando `pktool setpin`

Puede generar una frase de contraseña para un objeto de un almacén de claves y para el almacén de claves en sí. La frase de contraseña es necesaria para acceder al objeto o al almacén de claves. Para ver un ejemplo de generación de una frase de contraseña para un objeto de un almacén de claves, consulte el Ejemplo 4-4, “Exportación de un certificado y una clave privada en formato PKCS #12”.

1. **Genere una frase de contraseña para acceder a un almacén de claves.**

```
% pktool setpin keystore=nss|pkcs11 [dir=directory]
```

El directorio predeterminado para el almacenamiento de claves es `/var/`*username*.

La contraseña inicial para un almacén de claves PKCS #11 es `changeme`. La contraseña inicial para un almacén de claves NSS es una contraseña vacía.

2. Responda a las peticiones de datos.

Cuando se le solicite el token de frase de contraseña actual, escriba el PIN para un almacén de claves PKCS #11 o presione la tecla Intro para un almacén de claves NSS.

```
Enter current token passphrase:      Type PIN or press the Return key
Create new passphrase:               Type the passphrase that you want to use
Re-enter new passphrase:             Retype the passphrase
Passphrase changed.
```

El almacén de claves está ahora protegido por la *frase de contraseña*. Si pierde la frase de contraseña, perderá el acceso a los objetos del almacén de claves.

3. (Opcional) Muestre una lista de tokens.

```
# pktool tokens
```

La salida depende de si la metarranura está activada. Para obtener más información sobre la metarranura, consulte “Conceptos en la estructura criptográfica” [9].

- Si la metarranura está activada, el comando `pktools token` genera una salida similar a la siguiente:

```
ID Slot   Name                      Token Name                      Flags
--        ---------                 ----------                      -----
0         Sun Metaslot              Sun Metaslot
1         Sun Crypto Softtoken      Sun Software PKCS#11 softtoken  LIX
2         PKCS#11 Interface for TPM TPM                             LXS
```

- Si la metarranura está desactivada, el comando `pktools token` genera una salida similar a la siguiente:

```
ID Slot   Name                      Token Name                      Flags
--        ---------                 ----------                      -----
1         Sun Crypto Softtoken      Sun Software PKCS#11 softtoken  LIX
2         PKCS#11 Interface for TPM TPM                             LXS
```

En las dos versiones de salida, los indicadores pueden ser cualquier combinación de lo siguiente:

- `L`: es necesario iniciar sesión
- `I`: inicializado
- `X`: el PIN del usuario caducó
- `S`: el PIN del sistema operativo caducó

ejemplo 4-5 Protección de un almacén de claves con una frase de contraseña

El ejemplo siguiente muestra cómo establecer la frase de contraseña para una base de datos NSS. Debido a que no se creó ninguna frase de contraseña, el usuario presiona la tecla de retorno en la primera petición de datos.

```
% pktool setpin keystore=nss dir=/var/nss
Enter current token passphrase:      Press the Return key
Create new passphrase: has8n0NdaH
Re-enter new passphrase: has8n0NdaH
Passphrase changed.
```

▼ Cómo generar un par de claves utilizando el comando `pktool genkeypair`

Algunas aplicaciones requieren un par de claves públicas/privadas. En este procedimiento, podrá crear estos pares de claves y almacenarlos.

1. **(Opcional) Si tiene previsto utilizar un almacén de claves, cree el almacén de claves.**

   - **Para crear e inicializar un almacén de claves PKCS #11, consulte Cómo generar una frase de contraseña mediante el comando `pktool setpin` [61].**
   - **Para crear e inicializar un almacén de claves NSS, consulte el Ejemplo 4-5, “Protección de un almacén de claves con una frase de contraseña”.**

2. **Cree el par de claves.**
   Utilice uno de los métodos siguientes.

   - **Cree el par de claves y almacene el par de claves en un archivo.**
     Las claves basadas en archivos se crean para aplicaciones que leen claves directamente de archivos en el disco. Normalmente, las aplicaciones que utilizan directamente bibliotecas criptográficas OpenSSL requieren que almacene las claves y los certificados de la aplicación en archivos.

Nota - El almacén de claves `file` no admite claves y certificados de curva elíptica (`ec`).

```
% pktool genkeypair keystore=file outkey=key-filename \
[format=der|pem] [keytype=rsa|dsa] [keylen=key-size]

keystore=file
```

El valor `file` especifica la ubicación de almacenamiento de tipo archivo para la clave.

`outkey=`*key-filename*

Especifica el nombre del archivo donde el par de claves se almacena.

`format=der|pem`

Especifica el formato de codificación del par de claves. La salida `der` es binaria y la salida `pem` es ASCII.

`keytype=rsa|dsa`

Especifica el tipo de par de claves que se puede almacenar en un almacén de claves `file`. Para obtener definiciones, consulte DSA y RSA.

`keylen=`*key-size*

Especifica la longitud de la clave en bits. El número debe ser divisible por 8. Para determinar los posibles tamaños de clave, utilice el comando `cryptoadm list -vm`.

- **Cree el par de claves y almacénelo en un almacén de claves PKCS #11.**

  Debe completar el Paso 1 antes de utilizar este método.

  El almacén de claves PKCS #11 se utiliza para almacenar objetos en un dispositivo de hardware. El dispositivo puede ser una tarjeta Sun Crypto Accelerator 6000, un dispositivo de módulo de plataforma de confianza (TPM) o una tarjeta inteligente que se conecta a la estructura criptográfica. PKCS #11 se puede utilizar para almacenar objetos en `softtoken`, o token basado en software, que almacena los objetos en un subdirectorio privado en el disco. Para obtener más información, consulte la página del comando man `pkcs11_softtoken`(5).

  Puede recuperar el par de claves del almacén de claves mediante una etiqueta que especifique.

  ```
  % pktool genkeypair label=key-label \
  [token=token[:manuf[:serial]]] \
  [keytype=rsa|dsa|ec]  [curve=ECC-Curve-Name]\
  [keylen=key-size] [listcurves]
  ```

  `label=`*key-label*

  Especifica una etiqueta para el par de claves. El par de claves se puede recuperar del almacén de claves por su etiqueta.

  `token=`*token*[:*manuf*[:*serial*]]

  Especifica el nombre del token. De manera predeterminada, el nombre del token es `Sun Software PKCS#11 softtoken`.

`keytype=rsa|dsa|ec [curve=`*ECC-Curve-Name*`]`

Especifica el tipo de par de claves. Para el tipo de curva elíptica (`ec`), especifica opcionalmente un nombre de curva. Los nombres de curva se muestran como salida a la opción `listcurves`.

`keylen=`*key-size*

Especifica la longitud de la clave en bits. El número debe ser divisible por 8.

`listcurves`

Muestra los nombres de curva elíptica que se pueden utilizar como valores para la opción `curve=` para un tipo de clave `ec`.

- **Genere un par de claves y almacénelo en un almacén de claves NSS.**

  El almacén de claves NSS es utilizado por servidores que dependen de NSS como interfaz criptográfica primaria.

  Debe completar el Paso 1 antes de utilizar este método.

```
% pktool keystore=nss genkeypair label=key-nickname \
[token=token[:manuf[:serial]]] \
[dir=directory-path] [prefix=database-prefix] \
[keytype=rsa|dsa|ec] [curve=ECC-Curve-Name]] \
[keylen=key-size] [listcurves]
```

`keystore=nss`

El valor `nss` especifica la ubicación de almacenamiento de tipo NSS para la clave.

`label=`*nickname*

Especifica una etiqueta para el par de claves. El par de claves se puede recuperar del almacén de claves por su etiqueta.

`token=`*token*[:*manuf*[:*serial*]]

Especifica el nombre del token. De manera predeterminada, el token es `Sun Software PKCS#11 softtoken`.

`dir=`*directorio*

Especifica la ruta de directorio a la base de datos NSS. De manera predeterminada, el valor de *directorio* es el directorio actual.

`prefix=`*database-prefix*

Especifica el prefijo a la base de datos NSS. El valor predeterminado es sin prefijo.

`keytype=rsa|dsa|ec [curve=`*ECC-Curve-Name*`]`

Especifica el tipo de par de claves. Para el tipo de curva elíptica, especifica opcionalmente un nombre de curva. Los nombres de curva se muestran como salida a la opción `listcurves`.

`keylen=`*key-size*

Especifica la longitud de la clave en bits. El número debe ser divisible por 8.

`listcurves`

Muestra los nombres de curva elíptica que se pueden utilizar como valores para la opción `curve=` para un tipo de clave `ec`.

3. (Opcional) Compruebe que la clave exista.

Utilice uno de los siguientes comandos, según dónde haya guardado la clave:

- **Verifique la clave en el archivo** *key-filename***.**

```
% pktool list keystore=file objtype=key infile=key-filename
Found n keys.
Key #1 - keytype:location (keylen)
```

- **Verifique la clave en el almacén de claves PKCS #11.**

```
$ pktool list objtype=key
Enter PIN for keystore:
Found n keys.
Key #1 - keytype:location (keylen)
```

- **Verifique la clave en el almacén de claves NSS.**

```
% pktool list keystore=nss dir=directory objtype=key
```

ejemplo 4-6 Creación de un par de claves utilizando el comando `pktool`

En el siguiente ejemplo, un usuario crea un almacén de claves PKCS #11 por primera vez. Después de determinar los tamaños de clave para los pares de claves RSA, el usuario genera un par de claves para una aplicación. Por último, el usuario verifica que el par de claves se encuentre en el almacén de claves. El usuario nota que la segunda instancia del par de claves RSA se puede almacenar en el hardware. Dado que el usuario no especifica un argumento `token`, el par de claves se almacena como un `Sun Software PKCS#11 softtoken`.

```
# pktool setpin
Create new passphrase:
Re-enter new passphrase:     Retype password
Passphrase changed.
% cryptoadm list -vm | grep PAIR
```

```
...
CKM_DSA_KEY_PAIR_GEN       512  3072 .  .  .  .  .  .  .  .  X  .  .  .  .
CKM_RSA_PKCS_KEY_PAIR_GEN  256  8192 .  .  .  .  .  .  .  .  X  .  .  .  .
...
CKM_RSA_PKCS_KEY_PAIR_GEN  256  2048 X  .  .  .  .  .  .  .  X  .  .  .  .
ecc: CKM_EC_KEY_PAIR_GEN,CKM_ECDH1_DERIVE,CKM_ECDSA,CKM_ECDSA_SHA1
% pktool genkeypair label=specialappkeypair keytype=rsa keylen=2048
Enter PIN for Sun Software PKCS#11 softtoken  :     Type password

% pktool list
Enter PIN for Sun Software PKCS#11 softtoken  :     Type password
No.      Key Type      Key Len.      Key Label
-------------------------------------------------
Asymmetric public keys:
1        RSA                         specialappkeypair
```

ejemplo 4-7 Creación de un par de claves que utiliza el algoritmo de curva elíptica

En el siguiente ejemplo, un usuario agrega un par de claves de curva elíptica (`ec`) al almacén de claves, especifica un nombre de curva y verifica que el par de claves se encuentre en el almacén de claves.

```
% pktool genkeypair listcurves
secp112r1, secp112r2, secp128r1, secp128r2, secp160k1
.
.
.
c2pnb304w1, c2tnb359v1, c2pnb368w1, c2tnb431r1, prime192v2
prime192v3
% pktool genkeypair label=eckeypair keytype=ec curves=c2tnb431r1
% pktool list
Enter PIN for Sun Software PKCS#11 softtoken  :     Type password
No.  Key Type  Key Len.  Key Label
-------------------------------------------------
Asymmetric public keys:
1    ECDSA              eckeypair
```

▼ Cómo firmar una solicitud de certificación utilizando el comando `pktool signcsr`

Este procedimiento se utiliza para firmar una solicitud de firma de certificado (CSR) PKCS #10. CSR puede estar en formato PEM o DER. El proceso de firma emite un certificado X.509 v3. Para generar una CSR PKCS #10, consulte la página del comando man `pktool`(1).

Antes de empezar Este procedimiento asume que usted es una autoridad de certificación (CA), ha recibido una CSR y está almacenada en un archivo.

1. **Recopile la siguiente información para los argumentos necesarios en el comando `pktool signcsr`:**

`signkey`	Si ha almacenado la clave del firmante en un almacén de claves PKCS #11, `signkey` es la etiqueta que recupera esta clave privada. Si ha almacenado la clave del firmante en un almacén de claves NSS o un almacén de claves de archivos, `signkey` es el nombre de archivo que alberga esta clave privada.
`csr`	Especifica el nombre de archivo de la CSR.
`serial`	Especifica el número de serie del certificado firmado.
`outcer`	Especifica el nombre de archivo para el certificado firmado.
`issuer`	Especifica el nombre del emisor de CA en formato de nombre distinguido (DN).

   Para obtener más información sobre argumentos opcionales para el subcomando `signcsr`, consulte la página del comando man `pktool`(1).

2. **Firme la solicitud y emita el certificado.**

   Por ejemplo, el siguiente comando firma el certificado con la clave del firmante del repositorio PKCS #11:

```
# pktool signcsr signkey=CASigningKey \
csr=fromExampleCoCSR \
serial=0x12345678 \
outcert=ExampleCoCert2010 \
issuer="O=Oracle Corporation, \
OU=Oracle Solaris Security Technology, L=Redwood City, ST=CA, C=US, \
CN=rootsign Oracle"
```

   El siguiente comando firma el certificado con la clave del firmante de un archivo:

```
# pktool signcsr signkey=CASigningKey \
csr=fromExampleCoCSR \
serial=0x12345678 \
outcert=ExampleCoCert2010 \
issuer="O=Oracle Corporation, \
OU=Oracle Solaris Security Technology, L=Redwood City, ST=CA, C=US, \
CN=rootsign Oracle"
```

3. **Envíe el certificado al solicitante.**

   Puede utilizar el correo electrónico, un sitio web u otro mecanismo para entregar el certificado al solicitante.

Por ejemplo, puede utilizar el correo electrónico para enviar el archivo `ExampleCoCert2010` al solicitante.

▼ Cómo gestionar complementos de terceros en KMF

Identifica el complemento proporcionándole un nombre de almacén de claves. Al agregar el complemento a KMF, el software lo identifica por su nombre de almacén de claves. El complemento se puede definir para aceptar una opción. Este procedimiento incluye cómo eliminar el complemento de KMF.

1. **Instale el complemento.**

```
% /usr/bin/kmfcfg install keystore=keystore-name \
modulepath=path-to-plugin [option="option-string"]
```

Donde

keystore-name	Especifica un nombre único para el almacén de claves que proporciona.
path-to-plugin	Especifica la ruta completa al objeto de biblioteca compartida para el complemento KMF.
option-string	Especifica un argumento opcional a un objeto de biblioteca compartida.

2. **Enumere los complementos.**

```
% kmfcfg list plugin
keystore-name:path-to-plugin [(built-in)] | [;option=option-string]
```

3. **Para eliminar el complemento, desinstálelo y verifique que se haya quitado.**

```
% kmfcfg uninstall keystore=keystore-name
% kmfcfg plugin list
```

ejemplo 4-8 Llamada a un complemento KMF con una opción

En el siguiente ejemplo, el administrador almacena un complemento KMF en un directorio específico de sitio. El complemento se define para aceptar una opción `debug`. El administrador agrega el complemento y verifica que el complemento esté instalado.

```
# /usr/bin/kmfcfg install keystore=mykmfplug \
modulepath=/lib/security/site-modules/mykmfplug.so
# kmfcfg list plugin
KMF plugin information:
```

```
----------------------
pkcs11:kmf_pkcs11.so.1 (built-in)
file:kmf_openssl.so.1 (built-in)
nss:kmf_nss.so.1 (built-in)
mykmfplug:/lib/security/site-modules/mykmfplug.so
# kmfcfg modify plugin keystore=mykmfplug option="debug"
# kmfcfg list plugin
KMF plugin information:
----------------------
...
mykmfplug:/lib/security/site-modules/mykmfplug.so;option=debug
```

El complemento ahora se ejecuta en modo de depuración.

Glosario de seguridad

AES Estándar de cifrado avanzado. Una técnica de cifrado de datos en bloques de 128 bits simétricos. En octubre de 2000, el gobierno de los Estados Unidos adoptó la variante Rijndael del algoritmo como estándar de cifrado. AES sustituye el cifrado principal de usuario como estándar gubernamental.

algoritmo Un algoritmo criptográfico. Se trata de un procedimiento informático establecido que realiza el cifrado o el hashing de una entrada.

algoritmo criptográfico Consulte algoritmo.

almacén de claves Un almacén de claves contiene contraseñas, frases de contraseña, certificados y otros objetos de autenticación para que recuperen las aplicaciones. Un almacén de claves puede ser específico para una tecnología o puede ser una ubicación que utilizan varias aplicaciones.

ámbito del servicio de nombres El ámbito en el que un rol puede operar, es decir, un host individual o todos los hosts gestionados por un servicio de nombres especificado, como NIS o LDAP.

aplicación con privilegios Una aplicación que puede sustituir los controles del sistema. La aplicación comprueba los atributos de seguridad, como UID, GID, autorizaciones o privilegios específicos.

archivo de ticket Consulte caché de credenciales.

archivo intermedio Un archivo intermedio contiene una copia cifrada de la clave maestra para el KDC. Esta clave maestra se utiliza cuando un servidor se reinicia para autenticar automáticamente el KDC antes de que inicie los procesos `kadmind` y `krb5kdc`. Dado que el archivo intermedio incluye la clave maestra, el archivo y sus copias de seguridad deben mantenerse en un lugar seguro. Si el cifrado está en peligro, la clave podría utilizarse para acceder o modificar la base de datos del KDC.

archivo keytab Un archivo de tabla de claves que contiene una o varias claves (principales). Un host o servicio utiliza un archivo keytab de la misma manera que un usuario utiliza una contraseña.

archivos de auditoría Logs de auditoría binarios. Los archivos de auditoría se almacenan de manera independiente en un sistema de archivos de auditoría.

asignación de dispositivos Protección de dispositivos en el nivel de usuario. La asignación de dispositivos restringe el uso exclusivo de un dispositivo a un usuario a la vez. Los datos del dispositivo se depuran antes de volver a utilizar el dispositivo. Las autorizaciones se pueden utilizar para limitar quién tiene permiso para asignar un dispositivo.

atributos de seguridad Sustituye la política de seguridad que permite que un comando administrativo se ejecute correctamente al ser ejecutado por un usuario y no por un superusuario. En el modelo de superusuario, los programas `setuid root` y `setgid` son atributos de seguridad. Cuando estos atributos se aplican a un comando, el comando se ejecuta correctamente sin importar quién lo ejecuta. En el modelo de modelo de privilegios, los privilegios del núcleo y otros derechos reemplazan programas de `setuid root` como atributos de seguridad. El modelo de privilegios es compatible con el modelo de superusuario, ya que el modelo de privilegios reconoce también los programas `setuid` y `setgid` como atributos de seguridad.

autenticación Proceso de verificación de la identidad reclamada de un principal.

autenticador Los clientes trasfieren autenticadores al solicitar tickets (desde un KDC) y servicios (desde un servidor). Contienen información que se genera mediante una clave de sesión conocida sólo por el cliente y el servidor y que se puede verificar como de origen reciente, lo cual indica que la transacción es segura. Cuando se utiliza con un ticket, un autenticador sirve para autenticar un principal de usuario. Un autenticador incluye el nombre de principal del usuario, la dirección IP del host del usuario y un registro de hora. A diferencia de un ticket, un autenticador se puede utilizar sólo una vez, generalmente, cuando se solicita acceso a un servicio. Un autenticador se cifra mediante la clave de sesión para ese cliente y ese servidor.

autorización 1. En Kerberos, el proceso para determinar si un principal puede utilizar un servicio, a qué objetos puede acceder el principal y el tipo de acceso permitido para cada objeto.

2. En la gestión de derechos de usuario, un derecho que se puede asignar a un rol o a un usuario (o que está incrustado en un perfil de derechos) para realizar una clase de operaciones que, de lo contrario, están prohibidas por la política de seguridad. Las autorizaciones se aplican en el nivel de aplicación del usuario, no en el núcleo.

Blowfish Algoritmo cifrado de bloques simétricos con una clave de tamaño variable que va de 32 a 448 bits. Bruce Schneier, su creador, afirma que Blowfish se optimiza en el caso de aplicaciones en que la clave se modifica con poca frecuencia.

caché de credenciales Un espacio de almacenamiento (generalmente, un archivo) que contiene credenciales recibidas del KDC.

certificado Un certificado de clave pública es un conjunto de datos que codifica un valor de clave pública, incluida cierta información acerca de la generación del certificado, como un nombre y la persona quién lo firmo, un hash o una suma de comprobación del certificado y una firma digital del hash. Juntos, estos valores forman el certificado. La firma digital garantiza que el certificado no se ha modificado.

Para obtener más información, consulte clave.

cifrado de clave privada En el cifrado de clave privada, el remitente y el receptor utilizan la misma clave para el cifrado. Consulte también cifrado de clave pública.

cifrado de clave pública Un esquema de cifrado en el que cada usuario tiene dos claves, una clave pública y una clave privada. En el cifrado de clave pública, el remitente utiliza la clave pública del receptor para cifrar el mensaje y el receptor utiliza una clave privada para descifrarlo. El servicio Kerberos es un sistema de clave privada. Consulte también cifrado de clave privada.

clave 1. Generalmente, uno de los dos tipos principales de claves:

- *Clave simétrica*: una clave de cifrado que es idéntica a la clave de descifrado. Las claves simétricas se utilizan para cifrar archivos.
- *Claves asimétrica* o *clave pública*: una clave que se utiliza en algoritmos de clave pública, como Diffie-Hellman o RSA. Las claves públicas incluyen una clave privada que sólo conoce un usuario, una clave pública utilizada por el servidor o recurso general y un par de claves privada-pública que combina ambas. La clave privada también se denomina clave *secreta*. La clave pública también se denomina clave *compartida* o clave *común*.

2. Una entrada (nombre de principal) en un archivo keytab. Consulte también archivo keytab.

3. En Kerberos, una clave de cifrado, que puede ser de tres tipos:

- *Clave privada*: una clave de cifrado que comparten un principal y el KDC, y que se distribuye fuera de los límites del sistema. Consulte también clave privada.
- *Clave de servicio*: esta clave tiene el mismo propósito que la clave privada, pero la utilizan servidores y servicios. Consulte también clave de servicio.
- *Clave de sesión*: una clave de cifrado temporal que se utiliza entre dos principales y cuya duración se limita a la duración de una única sesión de inicio. Consulte también clave de sesión.

clave de servicio Una clave de cifrado que se comparte entre un principal de servicio y el KDC, y se distribuye fuera de los límites del sistema. Consulte también clave.

clave de sesión Una clave generada por el servicio de autenticación o el servicio de otorgamiento de tickets. Una clave de sesión se genera para proporcionar transacciones seguras entre un cliente y un servicio. La duración de una clave de sesión está limitada a una única sesión de inicio. Consulte también clave.

clave privada Una clave que se asigna a cada principal de usuario y que sólo conocen el usuario del principal y el KDC. Para los principales de usuario, la clave se basa en la contraseña del usuario. Consulte también clave.

clave secreta Consulte clave privada.

cliente De manera restringida, un proceso que utiliza un servicio de red en nombre de un usuario; por ejemplo, una aplicación que utiliza `rlogin`. En algunos casos, un servidor puede ser el cliente de algún otro servidor o servicio.

De manera más amplia, un host que: a) recibe una credencial de Kerberos y b) utiliza un servicio proporcionado por un servidor.

Informalmente, un principal que utiliza un servicio.

código de autenticación de mensajes (MAC) MAC proporciona seguridad en la integridad de los datos y autentica el origen de los datos. MAC no proporciona protección contra intromisiones externas.

confidencialidad Consulte privacidad.

conjunto básico El conjunto de privilegios asignados al proceso de un usuario en el momento de inicio de sesión. En un sistema sin modificaciones, cada conjunto heredable inicial del usuario es equivalente al conjunto básico en el inicio de sesión.

conjunto de privilegios Una recopilación de privilegios. Cada proceso tiene cuatro conjuntos de privilegios que determinan si un proceso puede utilizar un privilegio determinado. Consulte límite definido, conjunto vigente, conjunto permitido y conjunto heredable.

Además, el conjunto básico de privilegios es la recopilación de privilegios asignados al proceso de un usuario en el momento de inicio de sesión.

conjunto heredable El conjunto de privilegios que un proceso puede heredar a través de una llamada a `exec`.

conjunto permitido El conjunto de privilegios que están disponibles para que utilice un proceso.

conjunto vigente El conjunto de privilegios que actualmente están vigentes en un proceso.

consumidor En la función de estructura criptográfica de Oracle Solaris, un consumidor es un usuario de los servicios criptográficos prestados por los proveedores. Los consumidores pueden ser aplicaciones, usuarios finales u operaciones de núcleo. Kerberos, IKE e IPsec son ejemplos de consumidores. Para ver ejemplos de proveedores, consulte proveedor.

credencial Un paquete de información que incluye un ticket y una clave de sesión coincidente. Se utiliza para autenticar la identidad de un principal. Consulte también ticket, clave de sesión.

derechos Una alternativa al modelo de superusuario de todo o nada. La gestión de derechos de usuario y la gestión de derechos de procesos permiten a una organización dividir los privilegios de superusuario y asignarlos a usuarios o roles. Los derechos en Oracle Solaris se implementan como privilegios de núcleo, autorizaciones y la capacidad de ejecutar un proceso como un UID o GID determinado. Los derechos se pueden recopilar en perfil de derechos y rol.

DES Estándar de cifrado de datos. Método de cifrado de clave simétrica que se desarrolló en 1975 y que ANSI estandarizó en 1981 como ANSI X.3.92. DES utiliza una clave de 56 bits.

dominio 1. La red lógica gestionada por una única base de datos de Kerberos y un juego de centros de distribución de claves (KDC).

2. La tercera parte de un nombre de principal. Para el nombre de principal `jdoe/admin@CORP.EXAMPLE.COM`, el dominio es `CORP.EXAMPLE.COM`. Consulte también nombre de principal.

DSA Algoritmo de firma digital. Algoritmo de clave pública con un tamaño de clave variable que va de 512 a 4096 bits. DSS, el estándar del gobierno de los Estados Unidos, llega hasta los 1024 bits. DSA se basa en el algoritmo SHA1 para las entradas.

ECDSA Algoritmo de firma digital de curva elíptica. Un algoritmo de clave pública que se basa en matemáticas de curva elíptica. El tamaño de una clave ECDSA es significativamente menor que el tamaño de una clave pública DSA necesaria para generar una firma de la misma longitud.

elemento inicial Un iniciador numérico para generar números aleatorios. Cuando el iniciador comienza desde un origen aleatorio, el elemento inicial se denomina *elemento inicial aleatorio*.

escalada de privilegios Obtención de acceso a recursos que se encuentran fuera del rango de recursos que permitan los derechos asignados, incluidos los derechos que anulan los valores predeterminados. Como resultado, un proceso puede realizar operaciones no autorizadas.

evento asíncrono de auditoría Los eventos asíncronos constituyen la minoría de los eventos del sistema. Estos eventos no están asociados con ningún proceso; por lo tanto, no hay procesos disponibles para bloquear y reactivar más adelante. Los eventos de inicio del sistema y entrada y salida de la PROM son ejemplos de eventos asíncronos.

evento de auditoría no atribuible Un evento de auditoría cuyo iniciador no se puede determinar, como el evento AUE_BOOT.

evento síncrono de auditoría La mayoría de los eventos de auditoría. Estos eventos están asociados con un proceso en el sistema. Un evento no atribuible que está asociado con un proceso es un evento síncrono, como un error de inicio de sesión.

FQDN Siglas en inglés de Fully Qualified Domain Name, nombre de dominio completo. Por ejemplo, `central.example.com` (en lugar de simplemente `denver`).

frase de contraseña Una frase que se utiliza para verificar que una clave privada haya sido creada por el usuario de la frase de contraseña. Una buena frase de contraseña tiene una longitud de 10 a 30 caracteres, combina caracteres alfabéticos y numéricos, y evita el texto y los nombres simples. Se le pedirá la frase de contraseña para autenticar el uso de la clave privada para cifrar y descifrar comunicaciones.

GSS-API Generic Security Service Application Programming Interface. Una capa de red que proporciona apoyo para diversos servicios de seguridad modulares, incluido el servicio Kerberos. GSS-API

proporciona servicios de privacidad, integridad y autenticación de seguridad. Consulte también autenticación, integridad, privacidad.

host Un sistema al que se puede acceder a través de una red.

imagen de único sistema Una imagen de único sistema se utiliza en la auditoría Oracle Solaris para describir un grupo de sistemas auditados que utilizan el mismo servicio de nombres. Estos sistemas envían sus registros de auditoría a un servidor de auditoría central, donde los registros se pueden comparar como si procedieran de un sistema.

instancia La segunda parte de un nombre de principal; una instancia cualifica la primera parte del nombre de principal. En el caso de un principal de servicio, la instancia es obligatoria. La instancia es el nombre de dominio completo del host, como en `host/central.example.com`. Para los principales de usuario, una instancia es opcional. Sin embargo, tenga en cuenta que `jdoe` y `jdoe/admin` son principales únicos. Consulte también nombre primario, nombre de principal, principal de servicio, principal de usuario.

integridad Un servicio de seguridad que, además de la autenticación del usuario, permite validar los datos transmitidos mediante una suma de comprobación criptográfica. Consulte también autenticación y privacidad.

KDC Centro de distribución de claves. Un equipo que tiene tres componentes Kerberos V5:

- Base de datos de claves y principal
- Servicio de autenticación
- Servicio de otorgamiento de tickets

Cada dominio tiene un KDC maestro y debe tener uno o varios KDC esclavos.

KDC esclavo Una copia de un KDC maestro, que es capaz de realizar la mayoría de las funciones del maestro. Cada dominio, generalmente, tiene varios KDC esclavos (y un solo KDC maestro). Consulte también KDC, KDC maestro.

KDC maestro El KDC maestro en cada dominio, que incluye un servidor de administración Kerberos, `kadmind`, y un daemon de otorgamiento de tickets y autenticación, `krb5kdc`. Cada dominio debe tener al menos un KDC maestro y puede tener varios KDC duplicados, o esclavos, que proporcionan servicios de autenticación a los clientes.

Kerberos Un servicio de autenticación, el protocolo utilizado por ese servicio o el código utilizado para implementar ese servicio.

La implementación de Kerberos en Oracle Solaris que está estrechamente basada en la implementación de Kerberos V5.

Aunque son técnicamente diferentes, "Kerberos" y "Kerberos V5" suelen utilizarse de forma indistinta en la documentación de Kerberos.

En la mitología griega, Kerberos (también escrito Cerberus) era un mastín feroz de tres cabezas que protegía las puertas de Hades.

kvno Siglas en inglés de Key Version Number, número de versión de clave. Un número de secuencia que realiza un seguimiento de una clave determinada en orden de generación. El kvno más alto corresponde a la clave más reciente y actual.

límite definido El límite exterior que indica qué privilegios están disponibles para un proceso y sus procesos secundarios.

Lista de control de acceso Una lista de control de acceso (ACL) proporciona un nivel de seguridad de archivos más específico que la protección de archivos UNIX tradicionales. Por ejemplo, una ACL permite autorizar el acceso de lectura de grupo a un archivo, pero permitir que un solo miembro de ese grupo escriba en el archivo.

MAC 1. Consulte código de autenticación de mensajes (MAC).

2. También se denomina etiquetado. En la terminología de seguridad gubernamental, MAC significa control de acceso obligatorio (del inglés Mandatory Access Control). Etiquetas como Top Secret y Confidential son ejemplos de MAC. MAC se diferencia de DAC, que significa control de acceso discrecional (del inglés Discretionary Access Control). Los permisos UNIX son un ejemplo de DAC.

3. En hardware, la dirección única del sistema en una LAN. Si el sistema está en una Ethernet, la dirección MAC es la dirección Ethernet.

MD5 Una función de hash criptográfica iterativa utilizada para autenticar mensajes, incluso las firmas digitales. Rivest desarrolló esta función en 1991. Su uso está descartado.

mecanismo 1. Un paquete de software que especifica técnicas criptográficas para lograr la autenticación o confidencialidad de los datos. Ejemplos: clave pública Diffie-Hellman, Kerberos V5.

2. En la función de estructura criptográfica de Oracle Solaris, la implementación de un algoritmo para un propósito determinado. Por ejemplo, un mecanismo DES que se aplica a la autenticación, como CKM_DES_MAC, es un mecanismo distinto de un mecanismo DES que se aplica al cifrado, CKM_DES_CBC_PAD.

mecanismo de seguridad Consulte mecanismo.

minimización La instalación del sistema operativo mínimo necesario para ejecutar el servidor. Cualquier software que no se relacione directamente con el funcionamiento del servidor no se instala o se suprime después de la instalación.

modelo de privilegios Un modelo de seguridad más estricto en un sistema informático que el modelo de superusuario. En el modelo de privilegios, los procesos requieren un privilegio para ejecutarse. La administración del sistema se puede dividir en partes discretas que se basan en los privilegios que los administradores tienen en sus procesos. Los privilegios se pueden asignar al proceso de inicio de sesión de un administrador. O bien, los privilegios se pueden asignar para que estén vigentes para determinados comandos solamente.

modelo de superusuario El modelo de seguridad UNIX típico en un sistema informático. En el modelo de superusuario, un administrador tiene todo el control del sistema o ningún control (todo o nada). Generalmente, para administrar el equipo, un usuario se convierte en superusuario (`root`) y puede llevar a cabo todas las actividades administrativas.

motor de exploración Una aplicación de terceros, que reside en un host externo, que examina un archivo para ver si contiene virus conocidos.

nombre de principal 1. El nombre de un principal, con el formato *nombre primario/instancia@DOMINIO*. Consulte también, instancia, nombre primario, dominio.

2. (RPCSEC_GSS API) Consulte principal de cliente, principal de servidor.

nombre primario La primera parte de un nombre de principal. Consulte también instancia, nombre de principal, dominio.

NTP Siglas en inglés de Network Time Protocol, protocolo de hora de red. Software de la Universidad de Delaware que permite gestionar la sincronización precisa del tiempo o del reloj de la red, o de ambos, en un entorno de red. Puede usar NTP para mantener el desfase de reloj en un entorno de Kerberos. Consulte también desfase de reloj.

nueva autenticación El requisito para proporcionar una contraseña para realizar una operación informática. Normalmente, las operaciones de `sudo` requieren una nueva autenticación. Los perfiles de derechos autenticados pueden contener comandos que requieren una nueva autenticación. Consulte perfil de derechos autenticados.

objeto público Un archivo que es propiedad del usuario `root` y que todos pueden leer, como cualquier archivo en el directorio `/etc`.

PAM Siglas en inglés de Pluggable Authentication Module, módulo de autenticación conectable. Una estructura que permite que se utilicen varios mecanismos de autenticación sin que sea necesario recompilar los servicios que los utilizan. PAM permite inicializar la sesión de Kerberos en el momento del inicio de sesión.

perfil de derechos También se conoce como perfil. Una recopilación de sustituciones de seguridad que se puede asignar a un rol o a un usuario. Un perfil de derechos puede incluir autorizaciones, privilegios, comandos con atributos de seguridad y otros perfiles de derechos que se denominan perfiles complementarios.

perfil de derechos autenticados Un perfil de derechos que requiere el rol de usuario asignado o que se escriba una contraseña antes de ejecutar una operación desde el perfil. Este comportamiento es similar al comportamiento de `sudo`. El período en que la contraseña es válida y configurable.

pista de auditoría La recopilación de todos los archivos de auditoría de todos los hosts.

política Generalmente, un plan o curso de acción que influye sobre decisiones y acciones, o las determina. Para los sistemas informáticos, la política suele hacer referencia a la política

de seguridad. La política de seguridad de su sitio es el conjunto de reglas que definen la confidencialidad de la información que se está procesando y las medidas que se utilizan para proteger la información contra el acceso no autorizado. Por ejemplo, la política de seguridad puede requerir que se auditen los sistemas, que los dispositivos se asignen para su uso y que las contraseñas se cambien cada seis semanas.

Para la implementación de la política en áreas específicas del SO Oracle Solaris, consulte política de auditoría, política en la estructura criptográfica, política de dispositivos, política Kerberos, política de contraseñas y política de derechos.

política de auditoría La configuración global y por usuario que determina qué eventos de auditoría se registran. La configuración global que se aplica al servicio de auditoría, generalmente, afecta qué información opcional se incluye en la pista de auditoría. Dos valores, `cnt` y `ahlt`, afectan al funcionamiento del sistema cuando se completa la cola de auditoría. Por ejemplo, es posible que la política de auditoría requiera que un número de secuencia forme parte de cada registro de auditoría.

política de contraseñas Los algoritmos de cifrado que se pueden utilizar para generar contraseñas. También puede referirse a cuestiones más generales sobre las contraseñas, como la frecuencia con la que deben cambiarse las contraseñas, cuántos intentos de escribir la contraseña se permiten y otras consideraciones de seguridad. La política de seguridad requiere contraseñas. La política de contraseñas requiere que las contraseñas se cifren con el algoritmo AES y puede exigir requisitos adicionales relacionados con la seguridad de las contraseñas.

política de derechos La política de seguridad que está asociada a un comando. Actualmente, `solaris` es la política válida para Oracle Solaris. La política `solaris` reconoce privilegios y política de privilegio extendida, autorizaciones y atributos de seguridad`setuid`.

política de dispositivos Protección de dispositivos en el nivel de núcleo. La política de dispositivos se implementa como dos conjuntos de privilegios en un dispositivo. Un conjunto de privilegios controla el acceso de lectura al dispositivo. El segundo conjunto de privilegios controla el acceso de escritura al dispositivo. Consulte también política.

política de seguridad Consulte política.

política en la estructura criptográfica En la función de estructura criptográfica de Oracle Solaris, la política es la desactivación de mecanismos criptográficos existentes. Después de esto, los mecanismos no se pueden utilizar. La política en la estructura criptográfica puede impedir el uso de un mecanismo determinado, como `CKM_DES_CBC`, de un proveedor, como DES.

política Kerberos Un conjunto de reglas que rige el uso de contraseñas en el servicio Kerberos. Las políticas pueden regular los accesos de los principales, o los parámetros de tickets, como la duración.

política para tecnologías de clave pública En la estructura de gestión de claves (KMF), la política es la gestión del uso de certificados. La base de datos de políticas KMF puede limitar el uso de las claves y los certificados administrados por la biblioteca KMF.

política RBAC Consulte política de derechos.

políticas de red Los valores configurados por las utilidades de red para proteger el tráfico de red. Para obtener información sobre la seguridad de la red, consulte "Protección de la red en Oracle Solaris 11.2 ".

principal 1. Un cliente o usuario con un nombre único o una instancia de servidor o servicio que participa en una comunicación de red. Las transacciones de Kerberos implican interacciones entre principales (principales de servicio y principales de usuario) o entre principales y KDC. En otras palabras, un principal es una entidad única a la que Kerberos puede asignar tickets. Consulte también nombre de principal, principal de servicio, principal de usuario.

2. (RPCSEC_GSS API) Consulte principal de cliente, principal de servidor.

principal admin Un principal de usuario con un nombre del tipo *nombre de usuario*`/admin` (como en `jdoe/admin`). Un principal admin puede tener más privilegios (por ejemplo, para modificar las políticas) que un principal de usuario común. Consulte también nombre de principal, principal de usuario.

principal de cliente (RPCSEC_GSS API) Un cliente (un usuario o una aplicación) que utiliza los servicios de red RPCSEC_GSS seguros. Los nombres de principales de cliente se almacenan con el formato `rpc_gss_principal_t`.

principal de host Una instancia determinada de un principal de servicio en la que el principal (indicado por el nombre principal `host`) está configurado para proporcionar un rango de servicios de red, como `ftp`, `rcp` o `rlogin`. Un ejemplo de un principal de host principal es `host/central.example.com@EXAMPLE.COM`. Consulte también principal de servidor.

principal de servicio Un principal que proporciona autenticación Kerberos para un servicio o servicios. Para los principales de servicio, el nombre de principal es el nombre de un servicio, como `ftp` y su instancia es el nombre de host completo del sistema que proporciona el servicio. Consulte también principal de host, principal de usuario.

principal de servidor (RPCSEC_GSS API) Un principal que proporciona un servicio. El principal de servidor se almacena como una cadena ASCII con el formato *servicio*`@`*host*. Consulte también principal de cliente.

principal de usuario Un principal atribuido a un usuario determinado. El nombre primario de un principal de usuario es un nombre de usuario y su instancia opcional es un nombre que se utiliza para describir el uso que se pretende hacer de las credenciales correspondientes (por ejemplo, `jdoe` o `jdoe/admin`). También se conoce como instancia de usuario. Consulte también principal de servicio.

principio de privilegio mínimo Consulte privilegio mínimo.

privacidad Un servicio de seguridad en el que los datos transmitidos se cifran antes de enviarse. La privacidad también incluye la integridad de los datos y la autenticación de usuario. Consulte también autenticación, integridad y servicio.

privilegio 1. En general, una potencia o capacidad para realizar una operación en un sistema informático que supera las facultades de un usuario común. Los privilegios de superusuario son todos los derechos que se le otorgan al superusuario. Un usuario con privilegios o una aplicación con privilegios es un usuario o aplicación con derechos adicionales.

2. Un derecho discreto en un proceso de un sistema Oracle Solaris. Los privilegios ofrecen un control más específico de los procesos que `root`. Los privilegios se definen y se aplican en el núcleo. Los privilegios también se denominan *privilegios de proceso* o *privilegios de núcleo*. Para obtener una descripción completa de los privilegios, consulte la página del comando man `privileges`(5).

privilegio mínimo Un modelo de seguridad que ofrece a un proceso especificado sólo un subconjunto de poderes de superusuario. El modelo de privilegios básico asigna suficientes privilegios a los usuarios comunes para que puedan realizar tareas administrativas personales, como montar sistemas de archivos o cambiar la propiedad de los archivos. Por otro lado, los procesos se ejecutan sólo con esos privilegios, que son necesarios para completar la tarea, en lugar de con toda la capacidad de superusuario, es decir, todos los privilegios. Los daños debidos a errores de programación como desbordamiento de la memoria intermedia se pueden contener para un usuario que no es root, que no tiene acceso a capacidades críticas como la lectura o escritura en archivos de sistema protegidos o la detención del equipo.

protección La modificación de la configuración predeterminada del sistema operativo para eliminar las vulnerabilidades de seguridad inherentes al host.

protocolo de Diffie-Hellman También se lo denomina "criptografía de claves públicas". Se trata de un protocolo de claves criptográficas asimétricas que desarrollaron Diffie y Hellman en 1976. Este protocolo permite a dos usuarios intercambiar una clave secreta mediante un medio no seguro, sin ningún otro secreto. Kerberos utiliza el protocolo Diffie-Hellman.

proveedor En la función de estructura criptográfica de Oracle Solaris, un servicio criptográfico proporcionado a los consumidores. Las bibliotecas PKCS #11, los módulos criptográficos y los aceleradores de hardware son ejemplos de proveedores. Los proveedores se conectan a la estructura criptográfica y también se conocen como *complementos*. Para ver ejemplos de consumidores, consulte consumidor.

proveedor de hardware En la función de estructura criptográfica de Oracle Solaris, un controlador del dispositivo y su acelerador de hardware. Los proveedores de hardware descargan operaciones criptográficas costosas del sistema informático y, de esa manera, liberan los recursos de la CPU para otros usos. Consulte también proveedor.

proveedor de software En la función de estructura criptográfica de Oracle Solaris, un módulo de software de núcleo o una biblioteca PKCS #11 que proporciona servicios criptográficos. Consulte también proveedor.

QOP Calidad de protección. Un parámetro que se utiliza para seleccionar los algoritmos criptográficos que se utilizan junto con el servicio de integridad o de privacidad.

RBAC Control de acceso basado en roles, la función de gestión de derechos de usuario de Oracle Solaris. Consulte derechos.

reconocimiento de privilegios Programas, secuencias de comandos y comandos que activan y desactivan el uso de privilegios en su código. En un entorno de producción, los privilegios que estén activados deben proporcionarse al proceso, por ejemplo, solicitando a los usuarios del programa que utilicen un perfil de derechos que agrega los privilegios al programa. Para obtener una descripción completa de los privilegios, consulte la página del comando man `privileges`(5).

red privada virtual (VPN) Una red que proporciona comunicaciones seguras al utilizar el cifrado y el establecimiento de túneles para conectar usuarios a través de una red pública.

relación Una variable de configuración o un vínculo definidos en los archivos `kdc.conf` o `krb5.conf`.

resumen Consulte resumen de mensaje.

resumen de mensaje Un resumen de mensaje es un valor hash que se calcula a partir de un mensaje. El valor hash identifica el mensaje casi de manera exclusiva. Un resumen es útil para verificar la integridad de un archivo.

rol Una identidad especial para ejecutar aplicaciones con privilegios que sólo los usuarios asignados pueden asumir.

RSA Método para la obtención de firmas digitales y criptosistemas de claves públicas. Dicho método lo describieron sus creadores, Rivest, Shamir y Adleman, en 1978.

SEAM El nombre del producto para la primera versión de Kerberos en sistemas Solaris. Este producto se basa en la tecnología Kerberos V5 desarrollada en el Instituto Tecnológico de Massachusetts (MIT, Massachusetts Institute of Technology). SEAM ahora se denomina servicio Kerberos. Continúa difiriendo levemente de la versión del MIT.

separación de tareas Parte de la noción de privilegio mínimo. La separación de tareas impide que un usuario realice o apruebe todas las operaciones que permiten completar una transacción. Por ejemplo, en RBAC, puede separar la creación de un usuario de inicio de sesión de la asignación de sustituciones de seguridad. Un rol crea el usuario. Un rol individual puede asignar atributos de seguridad, como perfiles de derechos, roles y privilegios a los usuarios existentes.

servicio 1. Un recurso proporcionado a clientes de la red, a menudo, por más de un servidor. Por ejemplo, si ejecuta `rlogin` en el equipo `central.example.com`, ese equipo es el servidor que proporciona el servicio `rlogin`.

2. Un servicio de seguridad (ya sea de integridad o privacidad) que proporciona un nivel de protección más allá de la autenticación. Consulte también integridad y privacidad.

servicio de seguridad Consulte servicio.

servidor Un principal que proporciona un recurso a los clientes de la red. Por ejemplo, si ejecuta `ssh` en el sistema `central.example.com`, ese sistema es el servidor que proporciona el servicio `ssh`. Consulte también principal de servicio.

servidor de aplicaciones Consulte servidor de aplicaciones de red.

servidor de aplicaciones de red Un servidor que proporciona aplicaciones de red, como `ftp`. Un dominio puede contener varios servidores de aplicaciones de red.

sesgo de reloj La cantidad máxima de tiempo que pueden diferir los relojes del sistema interno de todos los hosts que participan en el sistema de autenticación Kerberos. Si el sesgo de reloj se excede entre cualquiera de los hosts participantes, las solicitudes se rechazan. El desfase de reloj se puede especificar en el archivo `krb5.conf`.

SHA1 Algoritmo de hash seguro. El algoritmo funciona en cualquier tamaño de entrada que sea inferior a 2^{64} para generar una síntesis del mensaje. El algoritmo SHA-1 es la entrada de DSA.

shell de perfil En gestión de derechos, un shell que permite que un rol (o un usuario) ejecute desde la línea de comandos cualquier aplicación con privilegios asignada a los perfiles de derechos del rol. Las versiones del shell de perfil corresponden a los shells disponibles en el sistema, como la versión de `pfbash` de `bash`.

shell seguro Un protocolo especial para el inicio de sesión remoto seguro y otros servicios de red seguros a través de una red no segura.

TGS Servicio de otorgamiento de tickets. La parte del KDC que es responsable de emitir tickets.

TGT Ticket de otorgamiento de tickets. Un ticket emitido por el KDC que permite que un cliente solicite tickets para otros servicios.

ticket Un paquete de información que se utiliza para transmitir de manera segura la identidad de un usuario a un servidor o servicio. Un ticket es válido únicamente para un solo cliente y un servicio determinado en un servidor específico. Un ticket contiene el nombre de principal del servicio, el nombre de principal del usuario, la dirección IP del host del usuario, un registro de hora y un valor que define la duración del ticket. Un ticket se crea con una clave de sesión aleatoria que utilizará el cliente y el servicio. Una vez que se ha creado un ticket, se puede volver a utilizar hasta que caduque. Un ticket sólo sirve para autenticar un cliente cuando se presenta junto con un autenticador nuevo. Consulte también autenticador, credencial, servicio y clave de sesión.

ticket de sustituto Un ticket que puede utilizar un servicio en nombre de un cliente para realizar una operación para el cliente. Por lo tanto, se dice que el servicio actúa como sustituto del cliente. Con el ticket, el servicio puede asumir la identidad del cliente. El servicio puede utilizar un ticket de sustituto para obtener un ticket de servicio para otro servicio, pero no puede obtener un ticket de otorgamiento de tickets. La diferencia entre un ticket de sustituto y un ticket reenviable

es que un ticket de sustituto únicamente es válido para una sola operación. Consulte también ticket reenviable.

ticket inicial Un ticket que se emite directamente (es decir, que no se basa en un ticket de otorgamiento de tickets existente). Algunos servicios, como las aplicaciones que cambian las contraseñas, posiblemente requieran que los tickets se marquen como `iniciales` para garantizar que el cliente pueda demostrar que conoce su clave secreta. Esta garantía es importante porque un ticket inicial indica que el cliente se ha autenticado recientemente (en lugar de basarse en un ticket de otorgamiento de tickets, que posiblemente haya existido durante mucho tiempo).

ticket no válido Un ticket posfechado que todavía no puede utilizarse. Un servidor de aplicaciones rechaza un ticket no válido hasta que se valide. Para validar un ticket no válido, el cliente debe presentarlo al KDC en una solicitud TGS, con el indicador `VALIDATE` definido, después de que haya pasado la hora de inicio. Consulte también ticket posfechado.

ticket posfechado Un ticket posfechado no es válido hasta que transcurra un tiempo especificado tras su creación. Un ticket de este tipo es útil, por ejemplo, para los trabajos por lotes que deben ejecutarse tarde por la noche, ya que si el ticket es robado, no se puede utilizar hasta que se ejecute el trabajo por lotes. Los tickets posfechados se emiten como `no válidos` y siguen teniendo ese estado hasta que: a) haya pasado su hora de inicio, y b) el cliente solicite la validación por parte del KDC. Generalmente, un ticket posfechado es válido hasta la hora de vencimiento del ticket de otorgamiento de tickets. Sin embargo, si el ticket posfechado se marca como `renovable`, su duración suele definirse para que coincida con la duración total del ticket de otorgamiento de tickets. Consulte también, ticket no válido, ticket renovable.

ticket reenviable Un ticket que un cliente puede utilizar para solicitar un ticket en un host remoto sin que sea necesario que el cliente complete todo el proceso de autenticación en ese host. Por ejemplo, si el usuario `david` obtiene un ticket reenviable mientras está en el equipo de `jennifer`, `david` puede iniciar sesión en su propio equipo sin tener que obtener un ticket nuevo (y, por lo tanto, autenticarse nuevamente). Consulte también ticket de sustituto.

ticket renovable Debido a que los tickets con duraciones muy largas constituyen un riesgo de seguridad, los tickets se pueden designar como `renovables`. Un ticket renovable tiene dos horas de vencimiento: a) la hora de vencimiento de la instancia actual del ticket, y b) la duración máxima de cualquier ticket. Si un cliente desea seguir utilizando un ticket, debe renovarlo antes del primer vencimiento. Por ejemplo, un ticket puede ser válido por una hora, pero todos los tickets tienen una duración máxima de 10 h. Si el cliente que tiene el ticket desea conservarlo durante más de una hora, debe renovarlo. Cuando un ticket alcanza la duración máxima, vence automáticamente y no se puede renovar.

tipo Históricamente, *tipo de seguridad* y *tipo de autenticación* tenían el mismo significado; ambos indicaban el tipo de autenticación (AUTH_UNIX, AUTH_DES, AUTH_KERB). RPCSEC_GSS también es un tipo de seguridad, aunque proporciona servicios de privacidad e integridad, además de autenticación.

tipo de seguridad Consulte tipo.

usuario con privilegios Un usuario al que se asignan derechos más allá de los derechos de usuario común en un sistema informático. Consulte también usuarios de confianza.

usuarios de confianza Usuarios con los que ha decidido que puede realizar tareas administrativas con determinado nivel de confianza. Normalmente, primero los administradores crean los inicios de sesión para usuarios de confianza y asignan derechos administrativos que coinciden con el nivel de confianza y la capacidad. Luego, estos usuarios ayudan a configurar y mantener el sistema. También denominados *usuarios con privilegios*.

Índice

D

L

M

N

O

P

R

S

T

U

V

Z