digicert®

Guía práctica para crear una política de firma de código: mejore la seguridad, el control y la gobernanza

GUÍA DE POLÍTICAS

Guía práctica para crear una política de firma de código: mejore la seguridad, el control y la gobernanza

Introducción

El objetivo de la firma de código

La firma de código añade al software un sello a prueba de manipulaciones y permite a los usuarios de software —incluidos los navegadores, las tiendas de aplicaciones y los sistemas operativos— verificar la identidad del creador del software y establecer que este no ha sido alterado desde que se firmó y se publicó.

La firma de código, que se basa en la infraestructura de clave pública (PKI), es un proceso de certificación no repudiable que establece la autoridad e integridad del software. En otras palabras, la firma del código demuestra, sin lugar a dudas, de dónde procede el software y que el código es original.

Para garantizar la confianza en la firma, las empresas deben proteger su ciclo de vida de desarrollo de software (SDLC) y su cadena de suministro de software (SSC). Reflejar las normas y prácticas de su empresa en una política de firma de código establece una base de referencia para que todos los equipos de desarrollo trabajen con las mismas directrices en materia de seguridad del código.

La importancia de contar con una política de firma sólida y documentada

Las normativas y estándares de seguridad que se aplican al SDLC y a la infraestructura de software están experimentando rápidos cambios debido a los ataques a la cadena de suministro de software y al propio SDLC. Y cada día se añaden nuevas normativas.

Tanto si se rige por los requisitos básicos del CA/B Forum, por la Directiva NIS2 o por los estándares sectoriales estipulados por la PCI DSS, es necesario elaborar una política única y coherente que contemple todas las normativas aplicables a su organización. Recuerde que no es solo el lugar donde está ubicada su empresa lo que determina la normativa aplicable, sino que su software debe cumplir también la normativa de los países en los que se venda y utilice.

La firma de código es un paso crucial en el proceso de creación y protección del software que afecta en gran medida a los equipos de seguridad y operaciones de desarrollo (DevOps). La seguridad y las DevSecOps determinan las acciones de seguridad que deben realizarse y por qué son necesarias. DevOps necesita traducir ese conocimiento en pasos concretos, configuraciones de sistemas e integraciones para alcanzar estos objetivos. A menudo, estas medidas de seguridad se ven como un impedimento para la entrega de software. Sin embargo, integrar los principios de la política en las aplicaciones de firma y las plataformas de CI/CD permite a los desarrolladores firmar el código correctamente y sin problemas, lo que aumenta la velocidad y la eficiencia del proceso de publicación.

Esta guía ayuda a coordinar estos grupos y generar las acciones de seguridad adecuadas en función del riesgo relativo al producto de software, los entornos de desarrollo de software y el grado de madurez de la seguridad.

Cómo utilizar esta guía

1.ª parte: Recopilación de información y conocimientos
En la firma de código, entran en juego distintas partes interesadas de varios departamentos diferentes. La preparación y la recopilación de información ayudarán a agilizar la redacción de la política.

2.ª parte: Componentes comunes de la política
Asegúrese de entender los objetivos de cada sección de la política y las prácticas y procedimientos que los principales expertos en DevOps y seguridad del software consideran importantes. Evalúe el nivel de impacto de cada sección en sus procesos de publicación actuales.

3.ª parte: Redacción de la política
En esta parte encontrará un modelo que le ayudará a empezar a crear su propia política.

1.ª PARTE: Recopilación de información y conocimientos

Recopilación de conocimientos

En esta política se solapan la seguridad y las DevOps, por lo que es importante conocer la opinión de responsables de cada departamento, así como de los equipos de Producto e Ingeniería. También es conveniente recabar comentarios de profesionales que trabajen en otras áreas relacionadas que puedan verse directamente afectadas por los cambios que se produzcan en su empresa en lo relativo a la seguridad y al software. A la hora de redactar una política de firma de software, estas son algunas de las fuentes de información a las que se recurre con más frecuencia:

- CISO
- DevSecOps/SecOps/InfoSec
- DevOps
- Seguridad de los productos
- Gestión de productos
- Control de calidad y fiabilidad del sitio
- Cumplimiento
- Gestión de riesgos
- Clientes y socios

Recopilación de información

Equipos de software y sus herramientas de desarrollo

El valor de las investigaciones forenses
La mejor seguridad es la proactiva, pero los errores ofrecen la oportunidad de aprender lecciones muy valiosas. Realice investigaciones forenses periódicas de cualquier fallo de seguridad o error reciente. Si no ha experimentado ningún fallo o infracción en primera persona, examine los de los demás. Los errores y vulnerabilidades pueden ayudarle a crear una política de firma que minimice las amenazas a las que se expone y proteja mejor su código. Asegúrese de que su política crea artefactos que se puedan utilizar en las investigaciones y la corrección de vulnerabilidades.

Reglamentos, directivas y leyes

Identifique los reglamentos internos y externos que deben cumplirse. A menudo, los organismos de normalización proporcionan reglamentos, marcos y orientaciones adicionales sobre cómo cumplir la normativa. Fíjese en las directivas gubernamentales (como la NIS2 de la UE o las órdenes ejecutivas de EE. UU.), las normativas del sector (como las del CA/B Forum o la PCI) y los estándares normativos (ISO, ENISA y NIST).

Prevención y corrección

La firma de código no solo ayuda a prevenir la manipulación del software, sino que también genera registros y artefactos auditables que resultan útiles a la hora de corregir los problemas. Los SBOM firmados, los scripts de compilación, los contenedores, los registros de firmas y actividad, etc., contienen datos útiles para la investigación de un ataque de malware o del riesgo relativo a una nueva vulnerabilidad explotable.

2.ª PARTE: Componentes comunes de una política de firma de software

Ámbito de aplicación

Las políticas se redactan para servir a un propósito concreto. Establezca claramente quién y qué está o no sujeto a la política de firma de código. Por ejemplo:

Incluya

- **Personas:** desarrolladores, firmantes, revisores, responsables de seguridad del producto y auditores internos y externos
- **Código:** productos que crea para clientes externos, tanto gratuitos como de pago (productos externos), sistemas propios (productos internos)
- **Artefactos de software:** SBOM, macros, bibliotecas, contenedores y scripts de compilación

Excluya

Aplicaciones de terceros: el software que utilice para crear sus productos de software no debe estar firmado por su empresa (por ejemplo, herramientas de productividad como Microsoft Office o herramientas de desarrollo como JFrog). Cualquier software de terceros digno de confianza estará firmado digitalmente por sus propios creadores. Debe haber otras políticas que garanticen que la empresa utilice únicamente software firmado por un creador verificado.

Accesos y privilegios

Autenticación

La autenticación confirma que el usuario que intenta acceder al sistema es la persona esperada y que está autorizado a utilizar dicho sistema. Por lo general, ya existe un mecanismo de autenticación y la política únicamente establece que este debe utilizarse.

Se debe utilizar la autenticación multifactor (MFA) o la autenticación de servidor a servidor para acceder y autorizar la firma según los requisitos básicos del CA/B Forum.[1]

Control de acceso basado en roles (RBAC)

El acceso y los privilegios establecen las características de las personas o servidores que deben (o no deben) participar en las actividades de firma de código, así como las acciones que están autorizados a realizar y los recursos que están autorizados a utilizar.

RBAC[2] es una práctica recomendada de seguridad informática que incluye la definición de roles y la asignación de privilegios diferenciados, lo que también se conoce como separación de tareas. Los roles deben basarse en las tareas que hay que realizar (p. ej., «Remitente», «Firmante» y «Responsable de seguridad»), y no en grupos no relacionados (como «Administrador» o «Directivo»). Para evitar puntos únicos de fallo, siempre que sea posible, los roles deben asignarse a más de un usuario.

Incluso en un equipo pequeño, asignar accesos y privilegios a usuarios individuales puede suponer todo un reto desde el punto de vista logístico. La asignación de privilegios mediante roles facilita el mantenimiento y favorece la escalabilidad a medida que crece el equipo. Considere la posibilidad de tener otros roles además del de firmante. Por ejemplo: ¿quién debería poder solicitar, emitir y revocar certificados? ¿Hay algún rol que debieran desempeñar los responsables de PKI, seguridad o DevSecOps?

Mitigación del riesgo con el principio del mínimo privilegio[3]

Para tener un control más estricto de la seguridad, no se debe conceder a nadie más que el conjunto mínimo de accesos y privilegios. Esto es más importante en el caso del acceso a recursos básicos, acciones de riesgo e irreversibles y software de alto riesgo. Algunos ejemplos son:

- **Recursos básicos:** servidores de compilación y HSM
- **Acciones irreversibles:** eliminación de pares de claves y revocación de certificados
- **Acciones de riesgo:** exportación de pares de claves
- **Software de alto riesgo:** código con acceso de bajo nivel al sistema, como el acceso de root de Linux

El riesgo de cada producto o proyecto de software debe evaluarse en función de los daños que podrían producirse en el supuesto de que el software contuviese código malicioso o vulnerabilidades explotables que pudieran provocar un acceso no autorizado a los datos, la pérdida de funcionalidad del sistema o algún otro fallo.

La criticidad del software es un tema recurrente en las normativas. El término tiene un significado jurídico en EE. UU., definido por la Orden Ejecutiva 14028, promulgada en mayo de 2021. El NIST proporciona numerosas definiciones y numerosos materiales complementarios para ayudarle a determinar la criticidad.

El concepto de «software crítico» también puede encontrarse en otras normativas, como la Ley de Ciberresiliencia de la UE, según las cuales el software de red y otros productos críticos están sujetos a más normas y evaluaciones externas por organismos certificados.

En el caso del software crítico y las acciones de riesgo e irreversibles, las políticas de firma de código suelen recurrir a varios autorizadores para garantizar que el cambio no pueda realizarlo una única persona.

Otra forma de limitar los privilegios es autorizar a los equipos a acceder únicamente a los proyectos y productos en los que estén trabajando. Por ejemplo, un desarrollador que trabaje en infraestructura no debería poder firmar código para una publicación del equipo de aplicaciones móviles. Esto limita los privilegios no solo al papel que alguien desempeña, sino también al software que produce.

Gestión de certificados y pares de claves

Entornos de software

En los entornos de producción y de no producción, deben utilizarse pares de claves y certificados de firma de código únicos. Los entornos de prueba a los que se puede acceder en remoto y que procesan datos confidenciales deben tratarse como entornos de producción.

A cada par de claves le corresponde un certificado. Como mínimo, necesita un conjunto para la firma en los entornos de producción y otro para la firma en los de no producción. En teoría, se podría generar un certificado y par de claves nuevos para cada compilación, y algunos equipos lo hacen. Este procedimiento tiene la ventaja de limitar la exposición que resultaría de la existencia de una clave comprometida.

[1] Los requisitos del CA/B Forum sobre la MFA se encuentran en Baseline Requirements for the Issuance and Management of Publicly Trusted Code Signing Certificates Version 3.9.0, secciones 6.2.7.3 (Private key storage for Signing Services) y 6.5.1 (Specific computer security technical requirements).

[2] Según la definición de «RBAC» del NIST (Instituto Nacional de Normas y Tecnología de EE. UU.), muchas normas de cumplimiento normativo (como NIST SP 800-95 y NIST SP 800-53) se basan en la correcta implementación de RBAC.

[3] Véase el apartado AC-6 de NIST SP 800-53 para obtener más información sobre el principio del mínimo privilegio.

Almacenamiento de certificados y pares de claves

Desde el 1 de junio de 2023, las políticas y normas exigen que las claves privadas de firma de código tanto para los certificados con validación estándar/de empresa (OV) como para los certificados con validación extendida (EV) se generen y almacenen en dispositivos de hardware seguros según lo estipulado por el CA/B Forum.[4] Este requisito significa que las autoridades de certificación (CA) públicas ya no pueden admitir la generación de claves y la instalación de certificados basadas en el navegador, ni ningún otro proceso que incluya la creación de una CSR (solicitud de firma de certificado) y la instalación de su certificado en un portátil o servidor. El método de aprovisionamiento se establece en el momento de la solicitud o renovación de los certificados.

Dispositivos de hardware seguros aceptados

- Tokens de hardware (p. ej., tokens USB criptográficos)
- Módulos de seguridad de hardware (HSM) —basados en la nube o locales— conformes con el estándar FIPS 140-2 de Nivel 2 o el Common Criteria EAL 4+

¿Dónde debería generar, almacenar y utilizar las claves para el código que no está destinado a la producción? Lo ideal sería utilizar un HSM con certificación FIPS/CC. Aun así, pueden aceptarse medidas menos costosas, como un sistema de gestión de secretos basado en software. Ciertas prácticas de riesgo como el almacenamiento de claves en sistemas de archivos inseguros o en el código fuente nunca son aceptables, y los miembros del equipo deberían saber que nunca deben recurrir a ellas.[5]

PKI pública vs. PKI privada: firma de software interno

La firma de código es más coherente cuando el software se va a implementar en tiendas de aplicaciones, sistemas operativos y navegadores regulados. Si el software requiere un proceso de firma para ejecutarse, ese código se firma. Estos requisitos no son esotéricos ni arbitrarios. Está demostrado que la firma protege el software de la corrupción, la intrusión y otras acciones malintencionadas. Cuando las plataformas aplican la firma, lo que están exigiendo es una solución práctica a una amenaza, y no un ideal pretencioso.

En la mayoría de los casos, la firma de software interno se ignora porque las empresas confían en los equipos de desarrollo conocidos y en la seguridad que rodea a sus sistemas. Las amenazas están al acecho fuera del perímetro que protegen los cortafuegos, una vez que el código se implementa en entornos externos. Por lógica parecería que, dentro de una empresa, el software que pasa de un equipo a otro o se implementa en servidores internos para uso interno está protegido.

Sin embargo, la realidad es que el software interno también es vulnerable a los ataques. Si alguien consigue acceder al sistema, el código sin firmar se convierte en un blanco fácil que puede utilizarse contra la infraestructura de toda la empresa. De acuerdo con las mejores prácticas, la firma interna no es un ideal, sino una cuestión práctica que protege su software y a su empresa al mismo nivel que los mecanismos de protección implementados para impedir que los actores maliciosos lo exploten.

El software que se utiliza o al que se accede fuera de su empresa debe firmarse con un par de claves respaldado por una CA que pueda verificarse fácilmente mediante navegadores y listas de PKI estándar.

El software creado únicamente para uso interno de su empresa puede utilizar una PKI interna, también conocida como PKI privada, en la que los certificados están encadenados a una raíz de confianza, pero es la empresa la que los emite a través de una autoridad de certificación intermediaria (ICA) interna. Esta es una situación habitual en el caso del software que se desarrolla y se utiliza únicamente a nivel interno.

Los certificados y pares de claves utilizados en la PKI privada deben controlarse, gestionarse y protegerse, al igual que sus homólogos de PKI pública. Algunos sistemas de gestión del ciclo de vida de los certificados (CLM) pueden gestionar elementos tanto de PKI pública como de PKI privada.[6] Los administradores también deben incluir el certificado público de la raíz privada en las listas de raíces de confianza de los sistemas internos.

Inventario y ciclo de vida de los certificados

Lleve un inventario de todos los certificados de firma de código, supervise las fechas de caducidad y genere certificados nuevos de forma proactiva. En la práctica, en todas las empresas —salvo en las de tamaño muy reducido—, la gestión de inventarios y certificados solo puede realizarse eficazmente con un sistema de CLM que automatice el proceso. Si utiliza un número mínimo de certificados y claves (por ejemplo, uno para las pruebas y otro para producción), asegúrese de guardar una copia de seguridad en una ubicación segura.[7]

Los certificados deben revocarse y volver a emitirse según sea necesario y en función de la frecuencia o la vida útil prevista de las versiones de software. Automatizar la renovación de certificados garantiza que las versiones de software no se vean afectadas por certificados caducados. El grupo de trabajo sobre firma de código del CA/B Forum ha estado debatiendo la reducción de la validez de los certificados de firma de código, que pasaría de unos 3 años a 460 días (esto es, un año y tres meses). Automatizar ahora la renovación de los certificados de firma de código le preparará para gestionar este futuro cambio con mayor facilidad.

[4] Véanse los requisitos básicos del CA/Browser Forum para la emisión y gestión de certificados de firma de código de confianza pública.

[5] Historia de un infortunio: qué pasó cuando Kali Linux perdió la clave de firma de su repositorio.

[6] DigiCert tiene una solución para PKI pública y privada.

[7] Véase la nota 4.

Los certificados y las claves deben rotarse o sustituirse periódicamente para mantener los niveles de seguridad. Cambiarlos con regularidad limita el daño que puede causar la pérdida o el robo de una clave. Algunos sistemas de CLM disponen de funciones que permiten rotar las claves automáticamente.

Cifrado de claves

Para obtener la máxima seguridad, cifre las claves al nivel más alto posible, en función de lo que puedan descifrar los recursos que utilizan dichas claves. Su política debe establecer claramente qué algoritmo utilizar, así como el nivel mínimo aceptable de cifrado de claves para cada tipo de certificado o caso de uso (por ejemplo, la versión de producción de una aplicación móvil).

El NIST ha fijado el año 2030 como fecha límite para la retirada de algoritmos de cifrado heredados y ampliamente utilizados, como RSA, ECDSA, EdDSA, DH y ECDH. Estos algoritmos son vulnerables a la informática cuántica y quedarán totalmente desautorizados en 2035.

Con las renovaciones automáticas basadas en perfiles, puede establecerse una actualización del cifrado a través de la plantilla y, a continuación, producirse de forma automática y sistemática a medida que se sustituye cada clave. Los expertos y analistas del sector prevén que las empresas tendrán que actualizar su cifrado más de una vezen los próximos 5 años.

Proceso de firma de código

Revisiones de seguridad

El código fuente del software debe superar una revisión de seguridad de acuerdo con la política del ciclo de vida de desarrollo de software (SDLC) seguro.[8]

Cada versión y cada componente asociado deben analizarse para detectar posibles indicadores maliciosos y vulnerabilidades conocidas. El malware y el software vulnerable son tan inaceptables en los entornos de prueba y otros entornos ajenos a la producción como en producción. Por este motivo, el software debe analizarse en busca de estos problemas en el ciclo de desarrollo y en el proceso de CI/CD.

No todas las vulnerabilidades justificarían pausar una compilación, o ni siquiera el envío de código, ya que muchas son de gravedad baja y otras son difíciles o imposibles de explotar. Es necesario establecer un umbral de gravedad para las vulnerabilidades para saber cuándo hay que detener una compilación o publicación.

No firme ningún software que contenga malware confirmado o alguna vulnerabilidad explotable.

Procesos automatizados

Lo recomendado según las mejores prácticas es automatizar el proceso de firma de código como parte del SDLC y del ciclo de CI/CD. Utilice una autoridad de certificación (CA) aprobada para los certificados de firma de código de producción. Lo ideal es que los desarrolladores utilicen una clave de firma única para firmar cualquier código que envíen. Las firmas individuales deben verificarse antes de firmar una versión.

No deben utilizarse las mismas claves de firma de código de producción para diferentes usuarios, a menos que los logs de actividad puedan registrar e identificar las acciones por usuarios únicos, incluidos los usuarios del servicio.

Las empresas que automatizan el proceso también pueden aprovechar las ventajas de seguridad que ofrece la rotación frecuente de claves y certificados. Si no se recurre a la automatización, lo más práctico podría ser establecer una política según la cual las claves se roten cuando sea estrictamente necesario, y no de manera periódica.

Sellos de tiempo

Las versiones que sigan siendo válidas una vez que el certificado de firma de código haya caducado deben incluir un sello de tiempo de un servicio de sellado de tiempo aprobado para demostrar el momento en el que se realizó la firma.

Desde el 15 de abril de 2025, las autoridades de sellado de tiempo (TSA) aumentaron su seguridad con respecto al almacenamiento de certificados y claves, así como el nivel mínimo de algoritmo hash, que ahora ha de ser mayor o igual que SHA-2. Los certificados de sello de tiempo también deben incluir el EKU (uso extendido de clave) para el sellado de tiempo. Asegúrese de que sus sellos de tiempo cumplan los nuevos requisitos del CA/B Forum.

Firma de artefactos de la versión y listas de materiales de software (SBOM)

Los artefactos de la versión, incluida la rama de código del software, el SBOM, los scripts de compilación y los documentos de intercambio de explotabilidad de vulnerabilidades, deben firmarse y almacenarse para ayudar a detectar posibles manipulaciones de aserciones y archivos.

Algunas empresas utilizan claves distintas para firmar los diferentes artefactos, de modo que, si una clave se viera comprometida, esto solo ocasionaría problemas para uno de los elementos del paquete de compilación.

Los artefactos deben almacenarse de forma segura, y también podría ser necesario disponer de métodos seguros para compartirlos con los clientes u órganos de gobierno. Los SBOM, en particular, pueden formar parte de un paquete de propuesta comercial, una presentación de producto o una auditoría.

Auditoría y registro

Mantenga un registro de auditoría exhaustivo que incluya pruebas de las revisiones de código, análisis de seguridad y cambios en las acciones, los accesos y los privilegios relativos a las claves de firma de código.

Las actividades deben estar asociadas a usuarios y cuentas de servicio únicos. Esto puede hacerse mediante claves únicas para cada firmante o mediante registros de firma de software en el caso de que un grupo comparta las claves.

Debería existir un procedimiento estándar que estipule cómo se deben compartir los registros con los auditores y qué formatos de archivo se deben utilizar.

[8] Visite el sitio web de OWASP para obtener ayuda con el desarrollo de un SDLC seguro y consultar otras políticas de seguridad útiles.

La agilidad criptográfica ofrece preparación y respuesta ante el cambio

Preparación y respuesta ante incidentes y cambios inesperados

Debido al aumento del número y la frecuencia de los ataques a la cadena de suministro de software y al SDLC, es imprescindible estar preparado para responder a los cambios con poca antelación.

Defina y, en la medida de lo posible, automatice los procedimientos de documentación, revocación y reemisión de certificados y pares de claves comprometidos.

Una vez completado el proceso de revocación/reemisión, deberá volver a firmar e implementar el código y los artefactos de software que se hayan firmado con la clave comprometida.

Las empresas que automatizan estos procesos utilizan alias para los certificados y pares de claves, en lugar de sus ID reales. Esto permite que su sistema de CI/CD siga funcionando sin interrupciones y sin tener que realizar ajustes en la configuración.

Establezca procesos para actualizar el cifrado de claves como un evento puntual, de manera programada a medida que caducan los certificados y en bloque. Estos métodos abordan incidentes de poco y mucho calibre, así como actualizaciones programadas en función de la normativa.

Preparación y respuesta ante los cambios previstos

Las normativas cambian constantemente; sin embargo, suelen introducirse gradualmente o dar un margen de tiempo para realizar cambios en el sistema. Defina y, en la medida de lo posible, automatice los procedimientos para documentar y realizar los cambios progresivamente. Por ejemplo, realice actualizaciones cuando los certificados caduquen y haya que volver a emitirlos.

Hay dos cambios importantes para los que hay que estar preparados: la reducción del período de validez de los certificados y los cambios en los algoritmos de cifrado.

Hace tiempo que se habla de acortar los ciclos de vida, y la propuesta aprobada recientemente para reducir el período de validez de los certificados TLS es el principal indicador de que esto acabará ocurriendo.

Ahora que la criptografía postcuántica (PQC) está a punto de romper la mayoría de los algoritmos de cifrado utilizados hoy en día, seremos testigos tanto de la aparición de nuevos algoritmos complejos como de la eliminación de los actuales.

Establezca procesos para actualizar el cifrado de claves como un evento puntual, de manera programada a medida que caducan los certificados y en bloque. Estos métodos abordan ajustes de poco y gran calado, así como actualizaciones programadas en función de la entrada en vigor de nuevas normativas.

Revisión y actualización

Revise las políticas anualmente y actualícelas según sea necesario para garantizar su eficacia y su adecuación a las normas del sector y a los cambios que se produzcan en la empresa.

3.ª PARTE: Redacción de la política de firma de software
Comience descargando esta política editable, que indica los lugares en los que incluir los conceptos que hemos tratado en esta guía. Utilice ambos documentos conjuntamente para redactar una política de firma de software bien concebida.

Descargue el modelo de política aquí.

DigiCert® Software Trust Manager

¿Le gustaría saber más sobre cómo DigiCert puede ayudarle a incorporar la confianza en el código y el software? Póngase en contacto con un experto de DigiCert aquí.

Acerca de DigiCert

DigiCert es el proveedor número uno del mundo de confianza digital. Gracias a él, tanto los usuarios individuales como las empresas pueden utilizar Internet con la tranquilidad de saber que su presencia en el mundo digital está protegida. La plataforma de confianza digital DigiCert® ONE protege los sitios web, los accesos y comunicaciones empresariales, el software, las identidades, el contenido y los dispositivos para que las empresas respondan a toda una gama de necesidades en materia de confianza pública y privada con una visibilidad y un control centralizados. Su galardonado software y su liderazgo en el sector de los estándares, la asistencia y las operaciones convierten a DigiCert en el proveedor de confianza digital al que recurren las grandes empresas de todo el mundo. Para obtener más información, visite digicert.com/es.

© 2025 DigiCert, Inc. Todos los derechos reservados. DigiCert es una marca registrada de DigiCert, Inc. en los Estados Unidos y otros lugares. El resto de las marcas comerciales y marcas registradas pertenecen a sus respectivos propietarios.