digicert®

Guide pratique des politiques de sécurité pour la signature de code : améliorez la sécurité, les contrôles et la gouvernance

GUIDE DE DÉFINITION DES POLITIQUES

Guide pratique des politiques de sécurité pour la signature de code : améliorez la sécurité, les contrôles et la gouvernance

Introduction

L'objectif de la signature de code

La signature de code ajoute un sceau infalsifiable aux logiciels. Ce dernier permet aux utilisateurs – y compris les navigateurs, les app stores et les systèmes d'exploitation – de vérifier l'identité du créateur du logiciel et d'attester que le code n'a pas été modifié depuis qu'il a été signé et publié.

Basée sur l'infrastructure à clé publique (PKI), la signature de code est un processus d'attestation non répudiable, établissant de fait l'autorité et l'intégrité du logiciel. En d'autres termes, la signature du code prouve, sans aucun doute, l'origine du logiciel et l'intégrité du code.

Pour garantir la confiance dans cette signature, les organisations doivent sécuriser leur cycle de développement logiciel (SDLC) et leur supply chain logicielle (SSC). La formalisation de vos règles et pratiques dans une politique de sécurité pour la signature de code établit une base de référence afin d'harmoniser la sécurité du code à l'échelle de toutes les équipes de développement.

L'importance d'une politique de sécurité pour la signature solide et documentée

Les réglementations et les standards de sécurité régissant le cycle de développement logiciel (SDLC) et l'infrastructure sous-jacente connaissent des changements rapides en raison des attaques dans la supply chain logicielle et le cycle de développement lui-même. Et de nouvelles réglementations entrent en vigueur chaque jour.

Qu'il s'agisse des exigences de base du CA/B Forum, de la directive NIS2 ou des standards PCI DSS pour les paiements électroniques, vous devrez créer une politique unique et cohérente pour répondre à toutes les réglementations qui s'appliquent à votre organisation. N'oubliez pas que ce n'est pas seulement le lieu d'implantation de votre entreprise qui détermine les réglementations applicables. Vos logiciels doivent aussi être conformes aux réglementations des pays où ils sont vendus et utilisés.

La signature du code est une étape cruciale dans le processus de compilation (build) et pour la protection des logiciels. Elle a donc un impact important sur les équipes de sécurité et de développement (DevOps). La sécurité/DevSecOps régit les actions de sécurité à mettre en œuvre et les raisons pour lesquelles elles sont nécessaires. Quant au DevOps, il doit traduire ces connaissances en étapes, en paramètres systèmes et en intégrations concrets pour atteindre ces objectifs. Souvent, ces mesures de sécurité sont considérées comme un obstacle dans le processus de déploiements. Or, l'intégration de vos politiques aux applications de signature et aux plateformes CI/CD permettra aux développeurs de signer le code correctement et en toute transparence, ce qui améliorera la rapidité et l'efficacité du processus de déploiement.

Ce guide pourra vous aider à réunir ces groupes et à définir les mesures de sécurité appropriées en fonction du risque pour le logiciel lui-même, des environnements de développement logiciel et de la maturité de votre sécurité.

Comment utiliser ce guide

1e partie : Rassembler les informations et l'expertise
Les parties prenantes de la signature de code sont réparties dans plusieurs organisations. La préparation et la collecte d'informations permettront d'accélérer la rédaction de la politique.

2e partie : Composantes communes de la politique
Il s'agit ici de comprendre les objectifs de chaque section de la politique, ainsi que les pratiques et procédures considérées comme importantes par les principaux experts en sécurité logicielle et en DevOps. Déterminer le niveau d'impact de chaque section sur vos processus actuels de mises à disposition de logiciels..

3e partie : Rédigez votre politique
Voici un exemple pour vous aider à créer la vôtre.

1e PARTIE : Collecte d'informations et d'expertise

Rassembler les compétences

Parce que cette politique est le point d'intersection entre la sécurité et le DevOps, il est important d'obtenir l'avis des membres de chaque département, ainsi que des équipes produit et ingénierie. Veillez également à recueillir les avis des différents domaines susceptibles d'être directement touchés par les modifications apportées à la sécurité et aux logiciels au sein de votre organisation. En matière de rédaction de politiques de signature de logiciels, les contributeurs suivants sont les plus souvent cités :

- RSSI
- DevSecOps/SecOps/InfoSec
- DevOps
- Equipes en charge de la sécurité des produits
- Chef produit
- Assurance qualité et Equipes d'infrastructure
- Conformité
- Gestion des risques
- Clients et partenaires

Collecte d'informations

Les équipes logicielles et leurs outils de développement

Analyse de code au sens large
La meilleure sécurité est une sécurité proactive. Mais les erreurs offrent aussi l'occasion d'apprendre. Menez régulièrement des analyses de code au sens large du terme sur toute défaillance ou erreur récente en matière de sécurité. Si vous n'avez connu aucun manquement ou défaillance vous-même, examinez ceux des autres. Les erreurs et les vulnérabilités passées peuvent vous aider à créer une politique de signature qui réduit l'exposition aux menaces et protège mieux votre code. Veillez à ce que votre politique crée des artefacts à utiliser dans le cadre d'investigations et de la correction des vulnérabilités.

Réglementations, directives et lois

Identifiez les règlements internes et externes qui doivent être respectés. Souvent, les autorités de normalisation édictent des réglementations, des référentiels et des directives supplémentaires visant à faciliter la mise en conformité. Recherchez les directives gouvernementales (EU NIS2 et US Executive Orders), les réglementations sectorielles (CA/B Forum et PCI) et les normes réglementaires (ISO, ENISA & NIST) applicables.

Prévention et remédiation

La signature de code permet non seulement d'empêcher l'altération des logiciels, mais aussi de générer des journaux (logs) et des artefacts auditables, particulièrement utiles pour remédier aux problèmes. SBOM signés, scripts , conteneurs, signatures, logs d'activité… tous contiennent des données utiles pour enquêter sur une attaque par malware ou sur le risque d'une nouvelle vulnérabilité exploitable.

2e PARTIE : Principales composantes d'une politique de signature de logiciels

Champ d'application

Les politiques sont rédigées dans un but précis. Indiquez clairement qui et quoi est soumis, ou non, conformément à la politique de signature du code. Exemples d'éléments :

Inclure

- **Les personnes :** développeurs internes et externes, signataires, réviseurs, sécurité des produits et auditeurs
- **Code :** produits que vous développez pour des clients externes, qu'ils soient gratuits ou payants (produits externes), systèmes développés en interne (produits internes)
- **Composants logiciels :** SBOM, macros, bibliothèques, conteneurs et scripts

Exclusion

Les applications tierces : les logiciels que vous utilisez pour la création de vos propres produits logiciels ne doivent pas être signés par vous (par exemple, les outils de productivité comme Microsoft Office ou les outils de développement comme JFrog). Tout logiciel tiers digne de confiance sera signé numériquement par ses créateurs. D'autres politiques devraient garantir que l'organisation n'utilise que des logiciels signés par un créateur vérifié.

Accès et privilèges

Authentification

L'authentification confirme que l'utilisateur qui tente d'accéder au système est bien la bonne personne et qu'il est autorisé à utiliser le système. En général, un système d'authentification est déjà en place et la politique stipule simplement que ce système doit être utilisé.

L'authentification multifacteur (MFA) ou l'authentification de serveur à serveur doit être utilisée pour accéder à la signature et l'autoriser, conformément aux exigences de base du CA/B Forum[1].

Contrôle d'accès basé sur les rôles (RBAC)

L'accès et les privilèges établissent les caractéristiques des personnes ou des serveurs habilités ou non à participer aux activités de signature de code, ainsi que les actions qu'ils sont autorisés à entreprendre et les ressources qu'ils sont autorisés à utiliser.

Le RBAC[2] est une bonne pratique de sécurité informatique qui consiste à définir des rôles et à leur attribuer des privilèges distincts, une pratique également connue sous le nom de séparation des tâches (SoD). Les rôles doivent être basés sur les tâches à accomplir (auteur, signataire, responsable sécurité, etc.) et non sur des groupes n'intervenant pas dans le processus (administrateur, Comex, etc.). Pour éviter les points de défaillance uniques (SPOF), tous les rôles doivent être attribués à plusieurs utilisateurs dans la mesure du possible.

Même au sein d'une petite équipe, l'octroi des accès et privilèges à différents individus peut constituer un défi logistique de taille. L'attribution de privilèges associés aux rôles facilite la gestion et le changement d'échelle des processus à mesure que l'équipe s'agrandit. Envisagez de confier des rôles au-delà des seuls signataires. Par exemple, qui doit pouvoir demander, émettre et révoquer des certificats ? Des rôles doivent-ils être attribués aux responsables PKI, à la sécurité ou au DevSecOps ?

Atténuer les risques avec le principe du moindre privilège[3]

Pour une sécurité plus forte, il convient de n'accorder à quiconque qu'un minimum d'accès et de privilèges. Cela est d'autant plus important pour les accès aux ressources critiques, les actions risquées et irréversibles, et les logiciels à haut risque. En voici quelques exemples :

- **Ressources critiques :** serveurs de compilation et HSM
- **Actions irréversibles :** suppression d'une paire de clés et révocation d'un certificat
- **Actions risquées :** exportation de paires de clés
- **Logiciels à haut risque :** code avec accès au cœur du système, tel que l'accès root dans Linux

Le risque de chaque produit ou projet logiciel doit être évalué sur la base du préjudice potentiel si le logiciel contenait du code malveillant ou des vulnérabilités exploitables susceptibles d'entraîner un accès non autorisé aux données, une perte de fonctionnalité du système ou d'autres défaillances.

La criticité des logiciels est un thème récurrent d'un point de vue réglementaire. Aux États-Unis, ce terme a une signification juridique qui découle du Décret 14028 publié en mai 2021. Le NIST fournit de nombreuses définitions et des documents complémentaires pour vous aider à déterminer le niveau de criticité.

Le concept de logiciel critique se retrouve dans d'autres textes, tels que la loi sur la cyber-résilience de l'Union européenne (EU Cyber Resilience Act) où les logiciels réseau et autres produits critiques sont soumis à davantage de règles et d'évaluations par un organisme certifié.

Pour les logiciels critiques et les actions risquées et irréversibles, les politiques de signature de code imposent souvent la signature de plusieurs approbateurs afin de s'assurer qu'aucune personne ne peut à elle seule effectuer la modification.

Une autre façon de limiter les privilèges consiste à restreindre l'accès des équipes aux seuls projets et produits sur lesquels elles travaillent. Par exemple, un développeur travaillant sur l'infrastructure ne devrait pas pouvoir signer le code d'une version publiée par l'équipe chargée des applications mobiles. Cela limite les privilèges non seulement au rôle joué par une personne, mais aussi au logiciel qu'elle produit.

Gestion des certificats et des paires de clés

Environnements logiciels

Des paires de clés et des certificats de signature de code uniques doivent être utilisés pour les environnements de production et de non-production. Les environnements de test accessibles à distance et traitant des données sensibles doivent être traités comme des environnements de production.

Les paires de clés et les certificats ont un rapport de 1:1. A minima, vous devez disposer d'un jeu de signatures pour la production et d'un autre pour la non-production. En théorie, il est possible de générer un nouveau certificat et une nouvelle paire de clés pour chaque version, ce que font d'ailleurs certaines équipes. Cette procédure présente l'avantage de limiter l'exposition d'une clé compromise.

[1] Les exigences du CA/B Forum en matière de MFA figurent dans le document suivant Baseline Requirements for the Issuance and Management of Publicly Trusted Code Signing Certificates Version 3.9.0- Sections 6.2.7.3 « Private key storage for Signing Services » et 6.5.1 « Specific computer security technical requirements ».

[2] Comme le souligne la définition du RBAC par le NIST (National Institute of Standards and Technology), de nombreux standards (telles que NIST SP800-95 et NIST SP 800-53) reposent sur la bonne mise en œuvre du RBAC.

[3] Voir NIST SP 800-53 AC-6 pour plus d'informations sur le principe du moindre privilège.

Stockage des certificats et des paires de clés

Depuis le 1er juin 2023, les politiques et les standards exigent que les clés de signature de code privé pour les certificats standard/de validation d'organisation (OV) et de validation étendue (EV) soient générées et stockées sur des dispositifs matériels sécurisés selon le CA/B Forum[4]. Cette exigence signifie que les autorités de certification publiques ne peuvent plus prendre en charge la génération de clés et l'installation de certificats par navigateur, ou tout autre processus incluant la création d'une CSR (Certificate Signing Request) et l'installation de votre certificat sur un ordinateur portable ou un serveur. La méthode de provisionnement est définie lors de la commande ou du renouvellement des certificats.

Dispositifs matériels sécurisés acceptés

- Jetons matériels (par exemple, jetons cryptographiques USB)
- Modules de sécurité matériels (HSM) – on-prem ou sur cloud – conformes à la norme FIPS 140-2 niveau 2 ou Common Criteria EAL 4+

Où devez-vous générer, stocker et utiliser les clés pour le code de non-production ? Idéalement, vous devriez utiliser un HSM conforme aux normes FIPS/CC. Toutefois, des mesures moins coûteuses, telles qu'un logiciel de gestion des secrets, peuvent être acceptées. Les pratiques dangereuses, telles que le stockage des clés sur des systèmes de fichiers non sécurisés ou dans le code source, ne sont jamais acceptables et les membres de l'équipe doivent savoir que cette pratique est formellement interdite[5].

Signature de logiciels internes : PKI publique vs PKI privée

La signature de code est appliquée de façon plus régulière lorsque le logiciel est destiné à être déployé dans des app stores, des systèmes d'exploitation et des navigateurs réglementés. Si un logiciel doit être signé pour pouvoir fonctionner, alors son code sera signé. Ces exigences ne sont ni ésotériques ni arbitraires. Il est prouvé que la signature protège les logiciels contre la corruption, l'intrusion et d'autres intentions malveillantes. Lorsque les plateformes imposent la signature, ce n'est nullement par idéal : elles ne font qu'exiger une solution pratique à une menace réelle.

Dans la plupart des cas, les organisations ne signent pas le code de leurs logiciels internes car elles font confiance à leurs équipes de développeurs et à la sécurité qui entoure leurs systèmes. Pour elles, la menace rôde de l'autre côté des pare-feu, une fois que le code est envoyé dans le monde extérieur. Le bon sens voudrait pourtant qu'au sein d'une organisation, les logiciels transmis entre les équipes ou déployés sur des serveurs internes pour un usage interne soient protégés.

En réalité, les logiciels internes sont également vulnérables aux attaques. Si quelqu'un accède au système, le code non signé constitue une proie facile qui peut être utilisée contre l'infrastructure de l'ensemble de l'organisation. La signature interne n'est pas une posture dogmatique, c'est une mesure pratique qui protège votre logiciel et votre organisation tout autant que la protection déployée pour empêcher des acteurs malveillants de l'exploiter.

Les logiciels utilisables ou accessibles en dehors de votre organisation doivent être signés à l'aide d'une paire de clés reconnue par une autorité de certification et facilement vérifiable à l'aide de navigateurs et de listes PKI standard.

Les logiciels conçus uniquement pour l'usage interne de votre organisation peuvent utiliser une PKI interne, également connue sous le nom de PKI privée, dans laquelle les certificats sont rattachés à une racine de confiance, tout en étant émis par votre entreprise par l'intermédiaire d'une autorité de certification intermédiaire (ICA). Il s'agit d'une situation courante pour les logiciels développés et utilisés uniquement en interne.

Les certificats et les paires de clés de PKI privée doivent être contrôlés, gérés et sécurisés, tout comme leurs pendants de la PKI publique. Certains systèmes de gestion du cycle de vie des certificats (CLM) peuvent gérer à la fois des éléments PKI publics et privés[6]. Les administrateurs doivent également inclure le certificat public de la racine privée dans les listes de racines de confiance des systèmes internes.

Inventaire et cycle de vie des certificats

Maintenez un inventaire de tous les certificats de signature de code, surveillez les dates d'expiration et générez de nouveaux certificats de manière proactive. En pratique, à l'exception des très petites structures, la gestion de l'inventaire et des certificats n'est efficace que si elle effectuée à l'aide d'un système CLM qui automatise le processus. Si vous utilisez un nombre minimal de certificats et de clés, par exemple un pour les tests et un pour la production, veillez à ce qu'ils soient sauvegardés dans un endroit sûr[7].

Les certificats doivent être révoqués et réémis selon les besoins et en fonction de la fréquence et/ou de la durée de vie prévue des versions des logiciels. L'automatisation du renouvellement des certificats permet d'éviter que la mise en production d'un logiciel ne soit perturbée par un certificat expiré. Le groupe de travail Code Signing du CA/B Forum a discuté de la possibilité de raccourcir la durée de vie d'un certificat de signature de code d'environ 3 ans à 460 jours (environ 1 an et 3 mois). En automatisant dès maintenant le renouvellement des certificats de signature de code, vous pourrez négocier cette transition de façon fluide et transparente.

Les certificats et les clés doivent faire l'objet d'une rotation et/ou d'un remplacement réguliers afin de maintenir des niveaux de sécurité appropriés. En changeant régulièrement les clés et les certificats, on limite le préjudice causé par la perte ou le vol d'une clé. Certains systèmes CLM sont dotés d'une fonctionnalité de rotation automatique des clés.

[4] Voir les exigences de base du CA/Browser Forum pour l'émission et la gestion de certificats de signature de code publiquement approuvés.
[5] Scénario catastrophe : ce qui s'est passé lorsque Kali Linux a perdu la clé de signature de son dépôt de code.
[6] DigiCert propose une solution pour les PKI publiques et privées.
[7] Voir note de bas de page 4.

Chiffrement des clés

Pour une sécurité maximale, chiffrez les clés au niveau le plus élevé possible, en fonction de ce que les ressources qui utilisent ces clés peuvent déchiffrer. Votre politique doit clairement indiquer l'algorithme à utiliser et le niveau minimum acceptable de chiffrement des clés pour chaque type de certificat ou cas d'usage (par exemple, mise en production d'applications mobiles)

Le NIST a fixé l'échéance de 2030 pour la suppression des algorithmes de chiffrement anciens et encore largement utilisés, tels que RSA, ECDSA, EdDSA, DH et ECDH. Ces algorithmes sont en effet vulnérables à l'informatique quantique et seront totalement interdits d'ici 2035.

Avec les renouvellements automatiques basés sur les profils, la mise à niveau vers un chiffrement plus robuste peut être définie via le modèle et se produire automatiquement et systématiquement au fur et à mesure que chaque clé est remplacée. D'après les prévisions des experts et analystes, les entreprises devront mettre à jour leur système de chiffrement plus d'une fois au cours des cinq prochaines années.

Processus de signature du code

Revues de sécurité

Le code source du logiciel doit faire l'objet d'une revue de sécurité, conformément à la politique de sécurisation du cycle de développement logiciel (SDLC)[8].

Chaque version et composant associé doit ainsi être analysé pour détecter des indicateurs de malveillance et des vulnérabilités connues. Les malwares et autre logiciels vulnérables sont tout aussi inacceptables dans les environnements de test et hors production que dans les environnements de production. C'est pourquoi les logiciels doivent faire l'objet d'analyses de détection de ce type de problèmes au cours du cycle de développement et du processus CI/CD.

Toutes les vulnérabilités ne justifient pas l'arrêt d'une version logicielle ou même d'une livraison d'un code. Nombre d'entre elles sont bénignes et d'autres sont difficiles, voire impossibles à exploiter. Vous devez définir un seuil de vulnérabilité pour savoir quand suspendre une compilation ou une livraison de code.

Ne signez pas un logiciel s'il contient des malwares confirmés ou une vulnérabilité exploitable.

Processus automatisés

La meilleure pratique consiste à automatiser le processus de signature du code dans le cadre du SDLC et du pipeline CI/CD. Utilisez une autorité de certification (AC) agréée pour les certificats de signature du code de production. Dans l'idéal, les développeurs devraient utiliser une clé de signature unique pour signer tout code qu'ils soumettent. Les signatures individuelles doivent être vérifiées avant de signer la version dans son ensemble.

Les utilisateurs multiples ne sont pas autorisés à partager les clés de signature de code en production, sauf si le système de journalisation des activités permet d'enregistrer et d'identifier les actions d'utilisateurs uniques, y compris les utilisateurs des services.

Les organisations qui automatisent le processus peuvent également bénéficier de l'avantage sécurité d'une rotation fréquente des clés et des certificats. En l'absence d'automatisation, une politique de rotation des clés dite « paresseuse » peut constituer l'approche la plus pratique.

Horodatage

Les versions qui resteront valables après l'expiration du certificat de signature de code doivent inclure un horodatage provenant d'un service approuvé afin de prouver l'heure de la signature.

À partir du 15 avril 2025, les autorités d'horodatage (TSA) ont renforcé leur sécurité en matière de stockage des certificats et des clés, et le niveau minimum d'algorithme de hash a été relevé à SHA-2. Les certificats d'horodatage doivent également inclure l'utilisation étendue de la clé (EKU) pour l'horodatage. Veillez à ce que vos horodatages soient conformes aux nouvelles exigences du CA/B Forum.

Signature des artefacts de la version et de la nomenclature logicielle (SBOM)

Les artefacts logiciels, incluant les informations relatives aux développeurs, la SBOM, les scripts et les documents de VEX (Vulnerability Exploitability eXchange), doivent être signés et stockés afin de faciliter la détection de toute altération des assertions et des archives.

Certaines organisations signent différents artefacts logiciels avec des clés différentes, de sorte qu'une clé compromise n'entraîne des problèmes que pour un seul élément du pack logiciel.

Les artefacts logiciels doivent être stockés de manière sécurisée et peuvent exiger de suivre certaines méthodes pour le partage avec les clients ou les autorités de gouvernance. Les SBOM, en particulier, peuvent faire partie d'un ensemble, d'une version soumise, du licencing ou d'un audit.

Audit et journalisation

Maintenez un journal d'audit complet comprenant les résultats des revues d'audit, des scans de sécurité et de toute action sur la clé de signature du code, ainsi que des changements d'accès et de privilèges.

Les activités doivent être associées à des utilisateurs et à des comptes de service uniques. Cela peut se faire par le biais de clés uniques pour chaque signataire ou via des logs de signature du logiciel si un groupe partage les clés.

Une procédure standard doit être mise en place pour définir le protocole de partage des journaux avec les auditeurs et les formats de fichiers à utiliser.

[8] Voir OWASP pour obtenir de l'aide dans l'élaboration d'un SDLC sécurisé et d'autres politiques de sécurité utiles.

La crypto-agilité, gage de préparation et de réponse au changement

Préparation et réponse aux incidents et changements imprévus

Avec l'augmentation du nombre et de la fréquence des attaques contre la supply chain logicielle et le SDLC, il est indispensable de se préparer à l'avance au changement.

Définissez et, dans la mesure du possible, automatisez les procédures de documentation, de révocation et de réémission des certificats et des paires de clés compromis.

Après avoir suivi votre procédure de révocation/réémission, vous devrez re-signer et redéployer le code et les artefacts logiciels qui avaient été signés avec la clé compromise.

Les entreprises qui automatisent ces processus utilisent des alias de certificats et de paires de clés et n'utilisent pas les identifiants réels. Cela permet à leur système CI/CD de continuer à fonctionner sans interruption ni ajustement des paramètres.

Mettez en place des processus de renforcement des clés de chiffrement soit sous la forme d'une mise à niveau unique, soit selon un calendrier établi en fonction de l'expiration des certificats, soit en masse. Ces méthodes permettront de répondre aux incidents de petite ou de grande ampleur, ainsi qu'aux calendriers imposés par les différentes réglementations.

Préparation et réponse aux changements prévus

Les réglementations sont en constante évolution. Cependant, elles sont souvent introduites progressivement ou laissent un certain laps de temps pour apporter des changements au système. Définissez et, dans la mesure du possible, automatisez les procédures de documentation et de modification progressive. Par exemple, effectuez des mises à jour lorsque les certificats expirent et doivent être réémis.

Il faut se préparer à deux changements majeurs : des cycles de vie plus courts et un renforcement des algorithmes de chiffrement.

La réduction des cycles de vie fait l'objet de discussions depuis un certain temps, et le vote récent approuvant le raccourcissement de la durée de vie des certificats TLS est un signe avant-coureur de ce que nous réserve l'avenir.

Alors que la cryptographie post-quantique (PQC) est sur le point de casser la plupart des algorithmes de chiffrement utilisés aujourd'hui, nous assisterons à la fois à l'apparition de nouveaux algorithmes complexes et au retrait des algorithmes existants.

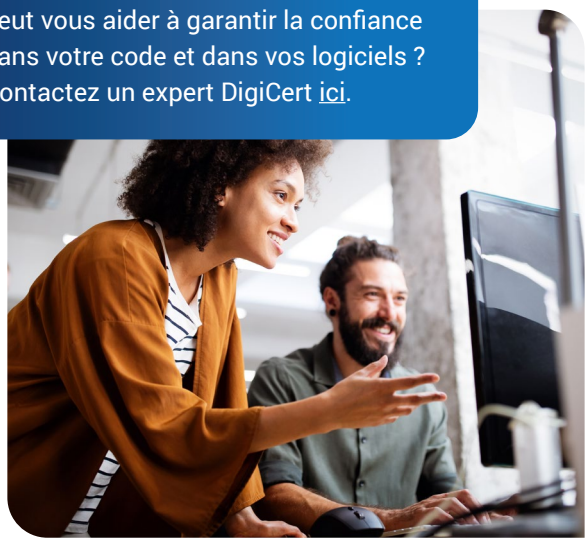
Mettez en place des processus de renforcement des clés de chiffrement soit sous la forme d'une mise à niveau unique, soit selon un calendrier établi en fonction de l'expiration des certificats, soit en masse. Ces méthodes permettront de répondre aux ajustements de petite ou de grande ampleur, ainsi qu'aux calendriers imposés par les différentes réglementations.

Révisions et mises à jour

Révisez vos politiques chaque année et mettez-les à jour si nécessaire afin de garantir leur efficacité et leur alignement sur les standards sectoriels et les changements organisationnels.

3e PARTIE : Rédigez votre politique de signature de vos logiciels

Prenez une longueur d'avance en téléchargeant ce modèle de politique à adapter à vos spécificités. Il comprend des espaces réservés à la formulation des concepts abordés dans ce guide. Utilisez-les ensemble pour élaborer une politique bien pensée pour la signature de vos logiciels.
Télécharger le modèle de politique ici.



À propos de DigiCert

Leader mondial de la confiance numérique, DigiCert apporte aux entreprises et aux particuliers les outils qui leur permettront d'échanger et de communiquer de façon sereine et sécurisée dans l'univers du digital. Sa plateforme DigiCert® ONE assure aux organisations une visibilité centralisée et un contrôle inégalé sur leurs besoins en certificats publics et privés pour sécuriser tout leur environnement : site web, accès et communications d'entreprise, logiciels, identités, contenus et appareils. DigiCert est le fournisseur de confiance numérique privilégié par les grandes entreprises du monde entier grâce à son logiciel maintes fois distingué et à son leadership dans les domaines des normes, du support et des opérations. Pour plus d'informations, rendez-vous sur le site digicert.com/fr.