PREMIER MINISTRE

Liberté
Égalité
Fraternité

Secrétariat général de la défense et de la sécurité nationale

Agence nationale de la sécurité des systèmes d'information

Paris, le 12 Juillet 2024

N° 1252 **/ANSSI/SDE/PSS/CCN**

Référence **: ANSSI-CC-NOTE-27_v1.0**

NOTE D'APPLICATION

EXIGENCES POUR LA RECHERCHE DE VULNERABILITES PUBLIQUES ET DE VULNERABILITES GENERIQUES

Application : Le 1er Septembre 2024.

Diffusion : Publique.

Le sous-directeur « Expertise » adjoint de l'Agence nationale de la sécurité des systèmes d'information

Annaïg ANDRO

SUIVI DES MODIFICATIONS

Version	Date	Modifications
1.0	12 Juillet 2024	Création du document

En application du décret n°2002-535 du 18 avril 2002 modifié, la présente note a été soumise, lors de sa création, au comité directeur de la certification, qui a donné un avis favorable.

Cette note est également soumise pour avis lors de chaque modification majeure conformément au manuel qualité du centre de certification. Les évolutions mineures ne sont pas soumises au comité directeur de la certification.

La présente note est disponible en ligne sur le site institutionnel de l'ANSSI (www.cyber.gouv.fr).

TABLE DES MATIERES

1. Objet de la note

La présente note d'application vise à préciser la méthodologie de recherche de vulnérabilités publiques (CVE) et de vulnérabilités génériques, dans le contexte plus général de l'analyse de vulnérabilités d'un produit. Cette note est applicable aux évaluations selon la Certification de sécurité de premier niveau [CSPN] ou selon les Critères Communs [CC].

Cette méthodologie décrit ce qui est attendu des Centres d'évaluation de la sécurité des technologies de l'information (CESTI) et des développeurs/commanditaires.

2. Définitions

Terme	Définition
CVE	*Common Vulnerabilities and Exposures.* Dictionnaire des informations publiques relatives aux vulnérabilités de sécurité.
COTS	*Commercial-off-the-shelf*. Produit développé et commercialisé en série (dit aussi « sur étagère »). Un produit soumis à évaluation peut inclure des composants de type COTS, par exemple un processeur ou une librairie propriétaire.
PoC	*Proof-of-Concept.* Preuve de concept : Evaluation de la faisabilité .
RTE	Rapport technique d'évaluation.
TOE	Target Of Evaluation

Tableau 1 : Termes et définitions

3. Recherche de vulnérabilités publiques (CVE)

Ce chapitre décrit les étapes que l'évaluateur devra suivre pour rechercher des vulnérabilités publiques (CVE) sur le produit évalué et ses composants, ainsi que quelques points d'attention.

3.1. Attendus du commanditaire

E1. Le commanditaire doit établir l'inventaire exhaustif des composants tiers (COTS ou *open source*, ainsi que leurs versions) intégrés au produit évalué.

Remarque : ce point est exigé au titre de [NOTE-09] dans le cadre de la [CSPN], et au titre des exigences ALC_CMS.2[1] et supérieures, pour les [CC].

Remarque : Dans la mesure où le CESTI a besoin d'un inventaire correct et exploitable afin de mener sa recherche, le CESTI aura toute latitude pour demander des correctifs ou clarifications de cette liste au titre de la [NOTE-20].

[1] ALC_CMS.2.1C *The configuration list shall include the following: the TOE itself; the evaluation evidence required by the SARs; and the parts that comprise the TOE.*

ALC_CMS.2.2C *The configuration list shall uniquely identify the configuration items.*

ALC_CMS.2.3C *For each TSF relevant configuration item, the configuration list shall indicate the developer of the item.*

E2. Dans le cas de composants COTS, le commanditaire doit être en mesure de justifier que ces composants sont bien maintenus en condition de sécurité, et fournir au CESTI la liste des vulnérabilités connues sur ces composants.

Remarque : cela peut consister à transmettre à l'évaluateur les *security advisories* envoyées au commanditaire par le développeur du COTS.

3.2. Attendus de l'évaluateur

E3. L'évaluateur doit :

- **établir un inventaire des vulnérabilités publiques et génériques applicables au produit à évaluer, en suivant une méthodologie interne respectant les exigences de l'annexe A ;**
- **prendre en compte cet inventaire dans l'analyse de vulnérabilité du produit à évaluer (Phase 5 en [CSPN] ou AVA_VAN en [CC]).**

4. Références

Référence	Document
[CSPN]	Procédure – Critères pour l'évaluation en vue d'une Certification de sécurité de premier niveau, Référence : ANSSI-CSPN-CER-P-02, version en vigueur.
[NOTE-01]	Note d'application – Méthodologie d'évaluation en vue d'une Certification de sécurité de premier niveau – Contenu et structure du RTE, Référence : ANSSI-CSPN-NOTE-01, version en vigueur.
[NOTE-09]	Note d'application – Contenu et structure de la cible de sécurité CSPN, Référence : ANSSI-CSPN-NOTE-09, version en vigueur.
[NOTE-20]	Note d'application – Règles relatives à la mise en œuvre des évaluations sécuritaires, Référence : ANSSI-CC-NOTE-20, version en vigueur.
[CC]	*Common Criteria for Information Technology Security Evaluation*, version en vigueur.

ANNEXE A. Méthodologie de recherche de vulnérabilités publiques et génériques (table des matières)

A.1. Vérification de la liste des composants et de leurs versions

L'évaluateur doit vérifier indépendamment la liste des composants fournie par le commanditaire.

Dans les cas où une vérification exhaustive est impossible dans le temps de l'évaluation, l'évaluateur doit pouvoir justifier sa méthode d'échantillonnage. En outre, l'évaluateur devra être attentif à la méthode utilisée pour la vérification : s'il effectue un audit de configuration sur une version en cours de développement, rien ne garantit a priori que les versions seront les mêmes sur le produit final.

A.2. Recherche de vulnérabilités publiques

L'évaluateur devra être capable d'établir une liste exhaustive des vulnérabilités publiques concernant les composants de la TOE, ce qui implique:

- L'identification des sources (bases de données) de vulnérabilités publiques ;
- La recherche dans ces sources ;
- La contextualisation des résultats ;
- La recherche croisée avec le développeur.

A.2.1. Identification des sources de vulnérabilités publiques

L'évaluateur devra dresser une liste des sources qu'il a consultées pour sa recherche : bases de données de vulnérabilités en ligne, mais aussi les sites des développeurs (p.ex. pour les librairies *open source*), les forums de discussion de constructeurs (p.ex. pour les composants matériels), les *mailing lists* modérées ou semi-modérées, etc.

Attention : Dans le cas des COTS, les vulnérabilités ne sont généralement pas publiques. La liste des vulnérabilités est donc établie à partir des informations transmises par le développeur du COTS au commanditaire de l'évaluation, qui a normalement souscrit au maintien en condition de sécurité de ce COTS.

Le CESTI est également invité à dresser une liste de sources pour les preuves de concept ou *exploits* associées aux vulnérabilités. Certains *exploits* peuvent en effet être testés rapidement (sans requérir d'adaptation) et permettent de qualifier rapidement les CVE.

A.2.2. Recherche dans les sources

L'évaluateur devra savoir mener une recherche, manuelle ou scriptée, dans les sources qu'il a sélectionnées.

Attention : plusieurs problèmes classiques devront être gérés :

- Les versions vulnérables peuvent être non trivialement indiquées dans les sources (p.ex. sous forme de fourchette entre deux versions) ;
- Les composants vulnérables peuvent être non trivialement indiqués dans les sources (p.ex. vulnérabilité JVM indiquée comme affectant JRE ou Java) ;
- Une source peut ne pas avoir fait le travail de vérification qu'une vulnérabilité est bien applicable à des composants similaires (p.ex. une vulnérabilité OpenJDK pourrait

également être applicable à Oracle JDK, une vulnérabilité RHEL pourrait également être applicable à CentOS) ;
- etc.

Attention : Si certains types de sources (p.ex. bases de données) se prêtent aux recherches scriptées, les sources moins structurées nécessiteront une recherche manuelle (site de développeur, *mailing lists*, etc.).

A.2.3. Contextualisation des résultats

L'évaluateur devra savoir identifier si un composant n'est plus supporté. Dans ce cas la recherche de CVE n'est plus pertinente car il y a dans ce cas une plus grande probabilité que des vulnérabilités existent et qu'un *exploit* commercial soit disponible à bas coût[2], sans qu'elles soient connues du grand public ni ne fassent l'objet de correctifs.

Remarque : il peut être difficile de caractériser avec certitude si un projet *open source* est encore supporté ou non, en particulier pour de petites librairies devenues stables et ne générant plus beaucoup d'activité. L'évaluateur pourra définir des règles internes pour qualifier ce risque.

Pour les composants supportés, l'évaluateur devra définir des critères de tri pour pouvoir prioriser l'analyse de la liste des vulnérabilités.

Remarque : il peut être difficile (et inutile) de chercher à qualifier en détail toutes les CVE applicables : certaines n'induisent aucun problème de sécurité sur le produit visé, certaines induisent un problème qui n'est pas pertinent dans le contexte de la cible de sécurité, etc. Il est donc crucial que l'évaluateur définisse un moyen de définir quelles CVE nécessitent un examen attentif et lesquelles peuvent rapidement être exclues de l'analyse.

Attention : l'évaluateur peut être tenté de simplement trier « par valeur de sévérité (ou CVSS) ». C'est un critère qui n'est pas toujours pertinent. Par exemple :

- un score CVSS peut être bas à cause du critère *Privileges Required* (la vulnérabilité n'est exploitable que pour un rôle d'administrateur avec privilèges élevés). Mais si la cible de sécurité prévoit que cet utilisateur est un attaquant potentiel, alors la vulnérabilité est pourtant critique dans le contexte de l'évaluation.
- un score CVSS peut être bas à cause du critère *Exploit Code Maturity* (aucun exploit « largement diffusé » n'est connu). Cela ne doit pas être considéré comme une garantie, car :
    - d'une part, le score CVSS n'est pas nécessairement à jour et ne constitue pas une garantie quant à la connaissance des offres d'*exploit* commerciales. Si l'évaluateur n'a pas de connaissance de première main de ces offres, alors il peut considérer que des CVE ayant atteint une certaine ancienneté ont potentiellement des vecteurs d'attaque commercialement disponibles, en particulier si elles touchent des composants grand public et/ou attractifs pour des fournisseurs d'*exploits*. Dans ce cas, l'évaluateur doit clairement signaler l'impossibilité de conclure ;
    - d'autre part, l'absence d'exploit largement répandu peut être dû au fait que le composant est lui-même peu connu. Cela ne signifie pas en soi que la vulnérabilité est difficile à exploiter pour un attaquant qui viserait ce composant en particulier. Au contraire, la publication pour certains produits

[2] On pense en particulier aux offres logicielles commerciales en vente sur *des plateformes illégales.*

open source de leurs listes de composant permet d'identifier facilement ces vulnérabilités.

- un score CVSS peut être élevé en raison d'un risque important sur l'intégrité d'une donnée, mais la cible de sécurité peut considérer qu'il n'est pas nécessaire de protéger cette donnée en intégrité, auquel cas la vulnérabilité n'est simplement pas applicable.

Attention : la présence d'un composant exploitable n'induit pas une attaque sur la TOE dans tous les cas. Dans certains cas, les exécutables ou bibliothèques vulnérables sont difficilement accessibles par un attaquant. Le CESTI est ainsi autorisé à définir une méthode lui permettant de limiter l'analyse de CVE aux composants dont la surface d'attaque est disponible à un attaquant potentiel. Concernant les bibliothèques, une telle méthode peut en particulier reposer sur la profondeur de l'arbre d'appel. Par exemple, si le CESTI limite à une profondeur de deux bibliothèques, en partant d'un binaire accessible à l'attaquant, son analyse inclura uniquement :

- le binaire ;
- les bibliothèques appelées directement par le binaire ;
- les bibliothèques appelées par ces bibliothèques.

Remarque : il est recommandé la plus grande prudence à l'évaluateur s'il souhaite inclure des critères organisationnels pour le tri des CVE. Il pourrait par exemple considérer qu'une CVE nécessitant l'installation d'une application malveillante est peu probable car un « magasin » officiel[3] valide les applications en amont. Attention, cet argument n'a de sens que si l'évaluateur a une preuve tangible du fait que cette protection organisationnelle est réellement capable de détecter cette vulnérabilité précise.

A.2.4. Recherche croisée avec le développeur

En cas de grand nombre de vulnérabilités identifiées, l'évaluateur pourra demander au développeur de fournir lui-même une liste des vulnérabilités publiques qu'il aura identifiées sur son produit. Celles-ci devront être suffisamment caractérisées pour que l'évaluateur puisse y appliquer ses propres critères de tri.

A.3. Recherche de vulnérabilités génériques

L'évaluateur doit effectuer un travail de fond de recherche et développement lui permettant :

- de connaître et de recenser les vulnérabilités typiques de la catégorie de produit évalué (suivi des conférences de la sécurité, veille spécifique à la sécurité d'un domaine vertical donné, etc.), ou ;
- d'améliorer sa capacité à les exploiter (construction d'un catalogue d'outils *open-source*, COTS ou développés par l'évaluateur lui-même).

Par exemple, pour un produit donné, l'évaluateur doit être capable de construire une liste comme celle qui suit à titre de capitalisation interne. Un extrait pertinent de cette liste peut être intégré dans le RTE afin de justifier les vulnérabilités génériques considérées :

[3] ou un vérifieur de code avant chargement.

Catégorie	Exemple de vulnérabilité potentielle à rechercher
Crypto	Exemple : Le produit est de type X et destiné au marché Y – il doit donc respecter, pour des raisons d'interopérabilité, une spécification qui emploie un algorithme de cryptographie faible => Regarder ce point en priorité pour la recherche de vulnérabilités. Source : spécification *XXXX*.
Réseau	Exemple : Le produit est de type X – or de nombreux produits de type X présentent des protocoles d'administration propriétaires – vérifier si c'est le cas du produit à évaluer et demander des charges d'évaluation permettant de faire la rétro ingénierie et/ou les spécifications du protocole le cas échéant. Source : évaluations précédentes *XXXX* et *XXXX*.
OS	Exemple 1 : Le produit est développé par constructeur Y, qui utilise systématiquement un OS propriétaire. Cet OS a été étudié par des chercheurs qui ont découvert des faiblesses structurelles sur les aspects : - possible exécution de code sur la stack TCP/IP, - durcissement contre les exécutions de code. Source : conférence XXXX 2020 Exemple 2 : L'OS du produit est un *fork* d'un OS de type Y. Ces OS peuvent être sujets à des élévations de privilège local. Avant de dérouler des scénarios *ad-hoc*, les évaluateurs passeront systématiquement l'outil XXX afin de vérifier rapidement si une élévation de privilège générique est applicable. Source : outil XXX, disponible sur distribution Kali, cf. catalogue d'outils du CESTI.
Langage	Exemple : Le produit est codé avec les langages Y et Z, et il est possible (sans que ce soit certain à ce stade) que son interface utilisateur utilise les langages W ou W'. On cherchera en priorité les vulnérabilités que peuvent introduire ces langages : - à travers l'outil XXXX, en suivant la méthodologie interne XXXX, pour le langage X ; - via revue manuelle de code, en suivant la méthodologie interne XXXX, pour le langage Y ; - pour l'interface, tentative d'identifier le langage utilisé via le test XXX, puis rechercher en priorité tel type d'attaque (p.ex. recherche d'objets sérialisés à modifier en utilisant le catalogue de gadgets XXX, recherche de crash via fuzzing…). Source : évaluations précédentes XXXX et XXXX.

Tableau 2 : Exemples de capitalisation de vulnérabilités par type de produit