Conservation de la valeur probante

Date	Version
15/06/2018	4.0 (Release 7)

État du document

◉ En projet ○ Vérifié ○ Validé

Maîtrise du document

Responsabilité	Nom	Entité	Date
Rédaction	JSL	Équipe Vitam	20/03/2018
Vérification			
Validation		Équipe Vitam	15/06/18

Suivi des modifications

Version	Date	Auteur	Modifications
0.1	25/11/2016	JSL	Initialisation
1.0	28/11/2016	EVR	Revue et Synchronisation de version – Release Bêta 0.11.0
1.1	10/01/17	MR	Ajout de la licence
1.2	20/04/17	JSL	Ajout sur journal cycle de vie
1.3	13/07/17	MR	Mise à jour publication Release 4
1.4	15/11/17	JSL	Mise à jour publication Release 5
2.0	28/11/2017	MR	Finalisation du document pour publication de la V1 fonctionnelle
2.1	15/01/2018	JSL	Affinage du contenu des logs sécurisés
2.2	05/03/2018	JSL	Mise à jour Release 6
3.0	20/03/2018	MR	Finalisation du document pour publication de la V1 de production
3.1	05/06/2018	JSL	Prise en compte de l'évolution R7 de la sécurisation des journaux,pour soutenir les très grandes volumétries et faciliter l'extraction de relevé de valeur probante, évolution back portée en R6 pour mise en production
3.2	06/06/2018	MRE	Relecture
4.0	15/06/2018	MRE	Finalisation du document pour publication de la Release 7

Documents de référence

Document	Date de la version	Remarques
NF Z42-013 - Archivage électronique - Spécifications relatives à la conception et à l'exploitation de systèmes informatiques en vue d'assurer la conservation et l'intégrité des documents stockés dans ces systèmes	01/03/2009	
NF Z42-020 - Spécifications fonctionnelles d'un composant Coffre-Fort Numérique destiné à la conservation d'informations numériques dans des conditions de nature à en garantir leur intégrité dans le temps	07/2012	
GA Z42-019 – Guide d'application de la NF Z42-013 (Archivage électronique - Spécifications relatives à la conception et à l'exploitation de systèmes informatiques en vue d'assurer la conservation et l'intégrité des documents stockés dans ces systèmes)	06/2010	

Licence

La solution logicielle VITAM est publiée sous la licence CeCILL 2.1 ; la documentation associée (comprenant le présent document) est publiée sous Licence Ouverte V2.0.

Table des matières

1. Introduction

La conservation de la valeur probante est un sujet central d'un système d'archivage électronique. L'objectif est de rendre prouvable toute opération effectuée sur toute unité archivistique ou tout objet qui lui est associé. Toutefois, vu les volumétries envisagées dans les implémentations de la solution logicielle Vitam, il est illusoire de gérer cette sécurisation objet par objet, en mettant en œuvre des principes cryptographiques (signatures des objets, des actions unitaires, etc.…) ; cela induirait une gestion lourde et porterait même des risques d'écroulement de confiance en cas de corruption de quelques clés. La sécurité d'un SAE doit être systémique, c'est-à-dire reposer sur un faisceau d'éléments redondants dont la modification simultanée et cohérente est impossible, ou plus exactement non réalisable en pratique. Les journaux constituent un élément central de cette sécurité systémique.

Les normes NF Z42-013, NF Z42-020 mais aussi le guide d'application GA Z42-019 ont donné un cadre pour la conservation de cette valeur probante qui est pris en compte et complété dans la solution logicielle Vitam.

Ce document présente rapidement, d'un point de vue fonctionnel, la sécurisation des journaux proposée dans la première version de production (Release 7) de la solution logicielle Vitam. Ce document devra être enrichi au fur et à mesure de l'avancement des travaux sur la gestion de la preuve.

2. Journaux

La solution Vitam met en place trois types de journaux métiers, portant des événements signifiants contribuant à la conservation de la valeur probante :

- Le journal des opérations, qui a pour objectif d'enregistrer toutes les opérations effectuées par la solution logicielle ayant un impact significatif sur les unités archivistiques, groupes d'objets et objets pris en charge par celle-ci.
- Les journaux du cycle de vie, qui ont pour objectif d'enregistrer toutes les actions significatives effectuées par la solution logicielle sur chacune des unités archivistiques et sur chacun des groupes d'objets techniques et des objets qui les composent. Est considérée comme une action significative, toute action modifiant l'entité concernée ou apportant une information significative sur son cycle de vie. Ces journaux sont créés lors de la réception des unités archivistiques et des groupes d'objets.
- Le journal des écritures, qui a pour objectif de tracer les opérations d'écriture effectuées par la solution logicielle sur les offres de stockage. Il garantit de fait l'horodatage et l'intégrité de tout élément dans le système.

À noter :
Il y a un décalage dans les appellations des journaux par rapport à la norme NF Z42-013. La solution logicielle Vitam fournissant des journaux précis pour chaque élément d'archives et pas

seulement sur les SIP, il a été considéré que le journal fin devait garder l'appellation de « journal du cycle de vie des archives ». Par contre, les événements macro du système sont enregistrés dans le « journal des opérations ».
Pour une description fine des journaux, voir le document « Organisation de l'information ».

3. Preuve systémique

La preuve systémique vient de la conjonction de ces trois journaux avec des informations croisées qui permettent d'assurer la traçabilité de tout événement, et d'en apporter la preuve, parfois par plusieurs voies

A titre d'exemple, l'entrée d'un objet, avec les éléments métiers utiles, peut être prouvée via :

- le journal de cycle de vie du groupe d'objets qui contient l'empreinte, la date de création du journal et l'identifiant d'opération d'entrée pour cet objet ;
- le journal des opérations qui contient l'identifiant d'opération d'entrée, la date de l'opération et l'identité du service versant ;

Cette entrée d'un objet peut aussi être prouvée via le journal des écritures qui assure de l'écriture dans le stockage de cet objet (via son empreinte), même si ce journal est de nature intermédiaire entre le log technique et le journal métier.

Cette réflexion initiale devra être précisée au fur et à mesure de la réflexion sur les relevés de preuve à apporter et avec la détermination des événements précis que l'on voudra pouvoir prouver. Les éléments figurant actuellement dans les journaux doivent permettre par composition de prouver tous les types d'événements et l'état du système.

4. Sécurisation des journaux

4.1 Contexte de sécurisation

La sécurisation des journaux permet de renforcer l'enregistrement des événements et consiste à apporter de la sécurité cryptographique sur l'objet journal en tant que tel.

Voici quelques éléments pris en compte dans la conception :

1. Le guide GA Z42-019 demande explicitement (cf 2.2.8.1.b) un chaînage des journaux, celui-ci est mis en œuvre.
2. La NF Z42-013 demande aussi un horodatage au moins toutes les 24 heures. Un fichier spécifique est généré avec toutes les lignes présentes dans le journal depuis la dernière sécurisation. Ce fichier est horodaté sûrement avec un tampon RFC 3161. Cette opération devra être faite au moins une fois par 24 heures.
3. La sécurité d'un tampon d'horodatage même ancien (plus que sa durée de validité

cryptographique) peut être assurée par le chaînage et la vérification de la chaîne jusqu'à un tampon valide. Pour raccourcir ce parcours, le chaînage sera fait aussi avec des journaux du mois et de l'année précédente.

4 Il est utile de pouvoir prouver une ligne du journal sans devoir montrer les autres pour des raisons de poids de données à transmettre dans un relevé de preuve mais aussi pour des raisons de confidentialité. Un mécanisme d'arbre de Merkle[1] est mis en œuvre pour rendre prouvable indépendamment chaque ligne.

En fait, la conjonction du chaînage et de l'arbre de Merkle constituent des principes des blockchains dont l'usage grandissant permet d'assurer de son efficacité.

4.2 Procédure de sécurisation

La sécurisation du journal est opérée par la génération puis la sauvegarde sur l'offre de stockage d'un fichier de sécurisation selon la procédure suivante :

- Extraction de l'ensemble des éléments du journal à raison d'une ligne par élément et en partant de la ligne la plus ancienne non sécurisée ;
  → Écriture dans un fichier nommé « data.txt »;
- Construction et calcul de la racine de l'arbre de Merkle :
  → Écriture de l'arbre de Merkle sous forme d'un arbre binaire json (root, Left, Rigth) dans un fichier « merkleTree.json » ;
- Prise en compte des données de calcul du tampon d'horodatage (racine de l'arbre de Merkle | TSP(Journal(précédent) | TSP(Journal(J-1 mois) | TSP(Journal(J-1 an)) :
  → Écriture dans un fichier « computing_information.txt » des quatres éléments utilisés pour le calcul du tampon d'horodatage
- Génération du tampon d'horodatage :
  → Écriture dans un fichier « token.tsp » ;
- Ajout des informations générales :
  → Écriture des informations de nombre d'enregistrement, de date de début et de date de fin des événements de journaux pris en compte dans le fichier « additional_information .txt » ;
- Clôture de l'opération :
  → Agrégation de l'ensemble des fichiers dans un conteneur .zip sans compression, sauvegardé dans l'offre de stockage

Cette procédure est lancée régulièrement sur les différents journaux, tenant par tenant. La périodicité de sécurisation est définie au niveau de l'installation de la plateforme.

À noter, en V1 (Release 6) la période de sécurisation commençait 5 minutes (paramétrables) avant la date de fin de la dernière sécurisation réussie. Cette mécanique a été modifiée pour éviter d'engendrer des doublons de sécurisation.

1 Pour une explication de l'arbre de Merkle et de son utilisation pour la preuve d'une partie des éléments voir https://www.certificate-transparency.org/log-proofs-work

4.3 Mise en œuvre sur le journal des opérations

Le journal des opérations est une table où chaque enregistrement est une opération :

- Chaque opération est composée d'une série d'événements (le premier étant un événement parmi les autres, sauf que c'est le premier, appelé ici bloc maître) ;
- Chaque événement dispose de sa date d'événement ;
- La dernière date d'event est donc celle du dernier event dans le tableau des events ;
- De plus, à chaque écriture effective en base, une date technique de persistance est utilisée pour tracer l'écriture des events en base.

Pour le journal des opérations, les éléments pris en compte pour la construction du « data.txt » sont sélectionnés en fonction du moment de dernière modification en base des journaux à sécuriser, en prenant depuis le moment de dernière sécurisation jusqu'au moment présent une période de temps paramétrable[2] (par défaut 5 mn).

La question s'est posée de prendre comme élément à sécuriser soit les événements unitaires des opérations, soit les opérations elles-mêmes. Il a été choisi de prendre l'ensemble des opérations ayant fait l'objet d'un événement dans la période qui doit être sécurisée (date technique de persistance). Cela permet, pour les opérations réalisées sur une longue période, d'avoir un enregistrement complet de l'opération jusqu'à sa finalisation.

Le fichier correspondant à l'extraction du journal des opérations est construit, de ce fait, avec tous les éléments du journal des opérations de la période de sécurisation (à savoir toutes les opérations dont l'un des événements a eu lieu dans la période temps de sécurisation), trié par date du dernier événement. Chaque élément est enregistré au format JSON, mais à plat sur une ligne avec les sauts de ligne encodés.

À noter : comme la sécurisation est aussi une opération, en général[3], le fichier de sécurisation du journal des opérations se finit par une opération de sécurisation non finalisée (celle en cours).

4.4 Mise en œuvre sur les journaux de cycle de vie

Chaque ArchiveUnit et chaque DataObjectGroup a son propre journal du cycle de vie qui suit tous les événements qui lui sont propres. Cela constitue une masse très importante de fichiers, stockés comme les objets et les méta-données pour en garantir la conservation optimale. Ces fichiers contiennent de nombreuses informations qui peuvent même être signifiantes (par exemple mention des changements de méta-données et de leur contenu lors d'une opération de mise à jour). Il faut donc être sélectif dans ce qui sera sécurisé pour à la fois assurer suffisamment de traces sûres et ne pas avoir dans les logs des informations soumises à élimination ou à des secrets particuliers comme le secret de Défense.

2 Ce délai est pour tenir compte de la latence de la base NoSQL au cœur de Vitam.
3 Si d'autres opérations ont été journalisées après le déclenchement de la journalisation mais avant sa fin, il peut y avoir des événements concurrents qui s'intercalent, et la période prise en compte assure un léger recouvrement pour éviter toute perte dans un environnement fortement distribué et donc non strictement synchrone...

Pour la sécurisation il a donc été choisi de faire une ligne par journal de cycle de vie ayant été affecté par une opération (une ligne est donc un couple cycle de vie/opération). Cette ligne portera des informations sur l'opération ayant généré l'événement de cycle de vie, des informations issues du cycle de vie, le hachage des métadonnées et du cycle de vie en base, le hachage du couple métadonnées/journal du cycle de vie stocké sur disque et enfin dans le cas des DataObjectGroup, la liste des objets et de leur hachage.

Pour les journaux de cycle de vie comme pour les journaux d'opérations, les éléments pris en compte pour la construction du « data.txt » sont sélectionnés en fonction du moment de dernière modification en base des journaux à sécuriser, en prenant depuis le moment de dernière sécurisation jusqu'au moment présent une période de temps paramétrable[4] (par défaut 5 mn). Si le nombre d'éléments est supérieur à une limite paramétrable (par défaut 100 000), l'extraction est limitée à ce nombre d'éléments et une nouvelle sécurisation est lancée pour prendre en compte le reste, et ceci jusqu'à épuisement.

À ce jour les opérations laissant une trace dans le cycle de vie des ArchiveUnit et des ObjectGroup sont l'entrée, la mise à jour, la mise à jour des règles de gestion et l'audit (en cas d'échec seulement et en base seulement en attendant la réparation).

Par exemple, on aura pour une ligne DataObjectGroup la structure suivante (mise à plat sans retour chariot) :


```
{
  "hGlobalFStorage":
"b1217c6339486e1d5b77aa496fa9b8805f4270ad59d1fbc754940641aea4656df56f122a150382
34909b3b4192a027e68249b58ac6d81f44d571e43b516ea423",
  "hLFC":
"ewN6aTW5T+m7RzuvJmBM4vLfKWdh33qly5JF1Bd3ooKZF/Xctp98ism9h2bl0O2Pmcfzw
pih0u5Q3vQrGdHYkA==",
  "hLFCEvts":
"d6gmO+wjZLye3L38Hu4geVjrYw9DaJ5+ScPthAyFRd8+CT2W88MCd9LNc2mEh2tlVoWr
yO50nCjVtc24JI4ZyQ==",
  "hMetadata":
"OsYUJhP1AxvpnuToMhq9JFYua2j9c3Jz4tdjoO7v5UxMJC/au450Zmvek3QirUjMrbDX6Etf
KOiunrVWkgY/Dg==",
  "hOGDocsStorage": [{
                "hObject":
"46126752c92661048f734a9df8ddbcbab717e60400a3da9ea81c49d6018deb0587b841ac9f767
701cb1b4b67818bb6c8344af13592f2b8b71ca7d7abf03ba992",
                "id": "aeaaaaaaaahdpurfabzhsald2t3xohqaaaaq"
        }
  ],
```


4 Ce délais est pour tenir compte de la latence de la base NoSQL au cœur de Vitam.

```
  "lEvDTime": "2018-06-06T12:04:18.388",
  "lEvTypeProc": "INGEST",
  "lEvtIdProc": "aeeaaaaaachotoh2abksaald2t3ww5qaaaaq",
  "lfcId": "aebaaaaaaahdpurfabzhsald2t3xohyaaaaq",
  "ltEvtOutcome": "OK",
  "mdType": "OBJECTGROUP",
  "up": ["aeaqaaaaaahdpurfabzhsald2t4c4paaaaaq", "aeaqaaaaaahdpurfabzhsald2t3xoiiaaaba"],
  "version": 9
}
```


Par exemple, on aura pour une ligne ArchiveUnit la structure suivante (mise à plat sans retour chariot) :


```
{
"hGlobalFStorage":
"42c7c6ca6e089878d8b1d7b5fe07d149a1cb413973bbba04e09e8aeb05e368137814eed5eb043d1e75
4d2df86395e30c63728712030f6fc63ec41a78911a2500",
"hLFC":
"JsFAf/OekcIa74djhzh8zePPWbU9ErygsbQylrUpTrXvu1cuEahuMf1HzwXV9KU25FHesBU4LY0
mNTesjhSDvg==",
"hLFCEvts":
"GSKYDzjbZuHzyB8T82mIkawOB41BXSgDGgLjwRRPYJE1KqGUsF/aCW3CmoRB3bV21j+w2
d1lMvilNNwr1DfJhA==",
"hMetadata":
"VYYbdOT+LYEaynutm5GZjK00QJgGma3Ny8eE8fvCSEAzYwq2lwFn2lG+acuN8TqAnwGEArY
DG5qUNzMHvEN3WQ==",
"idOG": "aebaaaaaaahdpurfabzhsald2tjahyyaaaba",
"lEvDTime": "2018-06-06T11:18:54.056",
"lEvTypeProc": "INGEST",
"lEvtIdProc": "aeeaaaaaachotoh2abksaald2ti7s4qaaaaq",
"lfcId": "aeaqaaaaaahdpurfabzhsald2tjah2yaaaaq",
"ltEvtOutcome": "OK",
"mdType": "UNIT",
"up": ["aeaqaaaaaahdpurfabzhsald2tjah3iaaaba"],
"version": 5
}
```


Pour reprendre en détail chaque champ, on a :

- hGlobalFStorage: hachage du fichier stocké réunissant métadonnées et journal de cycle de vie et dans le cas des DataObjectGroup :
- hLFC : hachage du journal de cycle de vie dans sa forme en base
- hLFCEvts : hachage des lignes d'événement du journal de cycle de vie, sans les informations annexes techniques en base (pour mémoire, ce hachage permet de garantir, en cas d'investigation ciblée, une capacité de vérification d'un LFC après reconstitution sans connaître des informations techniques liées seulement à la gestion de la persistance et de la

base dans le logiciel)
- hMetadata : hachage des métadonnées dans leur forme en base
- hOGDocsStorage : liste des objets stockés attachés au DataObjectGroup
  - id : identifiant unique de l'objet stocké
  - hObject : hachage de l'objet stocké
- idOG : identifiant unique du DataObjectGroup attaché le cas échéant à l'ArchiveUnit
- lEvDTime : date et heure de l'événement généré dans le cycle de vie
- lEvTypeProc : nature de l'opération
- lEvtIdProc : identifiant unique de l'opération
- lfcId : identifiant unique du cycle de vie, qui est aussi l'identifiant unique de l'ArchiveUnit ou le DataObjectGroup correspondant
- ltEvtOutcome : résultat de l'événement dans le journal de cycle de vie
- mdType : type UNIT pour ArchiveUnit ou OBJECTGROUP pour DataObjectGroup
- up : liste des identifiants uniques des ArchiveUnit parentes de cet ArchiveUnit ou de ce DataObjectGroup
- version : numéro de version, incrémenté à chaque modification de l'ArchiveUnit ou du DataObjectGroup, dans les métadonnées

A noter suite à la V1 (Release 6), des champs supplémentaires ont été ajoutés pour faciliter la construction automatique des relevés de valeur probante (à mettre en œuvre en Release 8) en ayant toute l'information directement dans les journaux sécurisés à chaque étape. Par ailleurs, pour des raisons de gestion des volumétries, la sécurisation des journaux de cycle de vie des ArchiveUnit et des DataObjectGroup a été séparée en opérations distinctes et donc dans des fichiers séparés et chaînés sur deux files distinctes. Enfin un numéro de version permettant dans l'avenir d'identifier différents formats de journal sécurisé a été ajouté.

4.5 Mise en œuvre sur le journal des écritures

Chaque écriture sur les offres de stockage donne lieu à une ligne de journal dans un journal des écritures. Cette ligne comporte :
- la date de l'écriture
- le tenant concerné
- le type d'action (CREATE, DELETE...)
- le nom du fichier
- le hash du fichier
- la taille
- les modules offres sur lesquels l'écriture a eu lieu
- le résultat de l'écriture

Pour des raisons techniques et des raisons de séparation des mécanismes de sécurité, il a été choisi de ne pas s'appuyer sur la base pour ce journal, mais simplement sur des mécanismes de log locaux

propres à chaque serveur de stockage.

La sécurisation se fait en 2 étapes :

- Sauvegarde des journaux locaux dans les offres. Cette opération est réalisée sur chacun des serveurs de stockage.
- Sécurisation globale de tous journaux sauvegardés dans les offres.

Le fait de ne pas s'appuyer sur la base et la volumétrie de ces journaux a amené deux différences avec les autres types de journaux sécurisés :

- la sécurisation ne reprend pas comme ligne à protéger par l'arbre de Merkle les lignes des journaux d'écriture, mais construit une ligne par journal d'écriture à sécuriser contenant son hachage. Le journal sécurisé des écritures prend donc en compte l'ensemble des journaux d'écritures, et fait référence aux journaux des écritures eux-mêmes stockés par ailleurs.
- Le chaînage n'est fait qu'avec le précédent.

Hors ces différences, la structure est respectée et vérifiable de la même façon que les autres. Par exemple sur une ligne pour un fichier journal d'écriture on aura :


```
{
  "FileName":
"0_storage_logbook_20180306132033436_20180306132514628_aecaaaaaacfdgbvvaamrealb7
n6msayaaaaq.log",
  "Hash":
"kPvTdzVoWKk7QE4U+1y03qjzICz6HynE3Febm5OE0hY2eThlLyqJ5\/GaEesFqHb\/hSGA+
fJRjrqOAFanklBfUQ=="
}
```


Pour reprendre en détail chaque champ, on a :

- FileName : nom du fichier contenant le journal des écritures,
- Hash: hachage du fichier.

Ce journal est une sécurité supplémentaire par rapport aux journaux métiers standards. Il peut servir d'ultime recours pour s'assurer de la présence d'un fichier dans le système à un moment donné.