



VITAM - Documentation d'installation

Version 3.0.1

VITAM

mars 20, 2020

Table des matières

1	Introduction	1
1.1	Objectif de ce document	1
2	Rappels	2
2.1	Information concernant les licences	2
2.2	Documents de référence	2
2.2.1	Documents internes	2
2.2.2	Référentiels externes	3
2.3	Glossaire	3
3	Prérequis à l'installation	6
3.1	Expertises requises	6
3.2	Pré-requis plate-forme	6
3.2.1	Base commune	6
3.2.2	PKI	7
3.2.3	Systèmes d'exploitation	8
3.2.3.1	Déploiement sur environnement CentOS	8
3.2.3.2	Déploiement sur environnement Debian	8
3.2.3.3	Présence d'un agent antivirus	9
3.2.4	Matériel	9
3.2.5	Librairie de cartouches pour offre froide	9
3.3	Questions préparatoires	9
3.4	Récupération de la version	10
3.4.1	Utilisation des dépôts <i>open-source</i>	10
3.4.1.1	<i>Repository</i> pour environnement CentOS	10
3.4.1.1.1	Cas de <i>griffins</i>	11
3.4.1.2	<i>Repository</i> pour environnement Debian	11
3.4.1.2.1	Cas de <i>griffins</i>	11
3.4.2	Utilisation du package global d'installation	11
4	Procédures d'installation / mise à jour	13
4.1	Vérifications préalables	13
4.2	Procédures	13
4.2.1	Cinématique de déploiement	13
4.2.2	Cas particulier d'une installation multi-sites	14
4.2.2.1	Procédure d'installation	14
4.2.2.1.1	vitam_site_name	14

4.2.2.1.2	primary_site	14
4.2.2.1.3	consul_remote_sites	15
4.2.2.1.4	vitam_offers	15
4.2.2.1.5	vitam_strategy	16
4.2.2.1.6	other_strategies	17
4.2.2.1.7	plateforme_secret	19
4.2.2.1.8	consul_encrypt	19
4.2.2.2	Procédure de réinstallation	20
4.2.2.3	Flux entre Storage et Offer	20
4.2.2.3.1	Avant la génération des keystores	21
4.2.2.3.2	Après la génération des keystores	21
4.2.3	Configuration du déploiement	21
4.2.3.1	Fichiers de déploiement	21
4.2.3.2	Informations <i>plate-forme</i>	22
4.2.3.2.1	Inventaire	22
4.2.3.2.2	Fichier vitam_security.yml	30
4.2.3.2.3	Fichier offers_opts.yml	31
4.2.3.2.4	Fichier cots_vars.yml	35
4.2.3.3	Déclaration des secrets	41
4.2.3.3.1	vitam	41
4.2.3.3.2	Cas des extras	45
4.2.3.3.3	Commande ansible-vault	45
4.2.3.3.3.1	Générer des fichiers <i>vaultés</i> depuis des fichier en clair	46
4.2.3.3.3.2	Ré-encoder un fichier <i>vaulté</i>	46
4.2.4	Gestion des certificats	46
4.2.4.1	Cas 1 : Configuration développement / tests	46
4.2.4.1.1	Procédure générale	46
4.2.4.1.2	Génération des CA par les scripts Vitam	46
4.2.4.1.3	Génération des certificats par les scripts Vitam	47
4.2.4.2	Cas 2 : Configuration production	47
4.2.4.2.1	Procédure générale	47
4.2.4.2.2	Génération des certificats	47
4.2.4.2.2.1	Certificats serveurs	47
4.2.4.2.2.2	Certificat clients	48
4.2.4.2.2.3	Certificats d'horodatage	48
4.2.4.2.3	Intégration de certificats existants	48
4.2.4.2.4	Intégration de certificats clients de VITAM	50
4.2.4.2.4.1	Intégration d'une application externe (cliente)	50
4.2.4.2.4.2	Intégration d'un certificat personnel (<i>personae</i>)	50
4.2.4.2.5	Cas des offres objet	50
4.2.4.2.6	Absence d'usage d'un <i>reverse</i>	50
4.2.4.3	Intégration de CA pour une offre <i>Swift</i> ou <i>s3</i>	51
4.2.4.4	Génération des magasins de certificats	51
4.2.5	Paramétrages supplémentaires	51
4.2.5.1	<i>Tuning</i> JVM	51
4.2.5.2	Installation des <i>griffins</i> (greffons de préservation)	51
4.2.5.3	Rétention liée aux logback	52
4.2.5.3.1	Cas des access_log	52
4.2.5.4	Paramétrage de l'antivirus (ingest-externe)	53
4.2.5.5	Paramétrage des certificats externes (*-externe)	53
4.2.5.6	Placer « hors Vitam » le composant ihm-demo	53
4.2.5.7	Paramétrer le <i>secure_cookie</i> pour ihm-demo	54
4.2.5.8	Paramétrage de la centralisation des logs VITAM	54
4.2.5.8.1	Gestion par VITAM	54

4.2.5.8.2	Redirection des logs sur un SIEM tiers	54
4.2.5.9	Passage des identifiants des référentiels en mode <i>esclave</i>	55
4.2.5.10	Paramétrage du batch de calcul pour l'indexation des règles héritées	55
4.2.5.11	Durées minimales permettant de contrôler les valeurs saisies	56
4.2.5.12	Fichiers complémentaires	56
4.2.5.13	Paramétrage de l'Offre Froide (librairies de cartouches)	77
4.2.5.14	Sécurisation SELinux	80
4.2.6	Procédure de première installation	81
4.2.6.1	Déploiement	81
4.2.6.1.1	Cas particulier : utilisation de ClamAv en environnement Debian	81
4.2.6.1.2	Fichier de mot de passe	82
4.2.6.1.3	Mise en place des repositories VITAM (optionnel)	82
4.2.6.1.4	Génération des <i>hostvars</i>	83
4.2.6.1.4.1	Cas 1 : Machines avec une seule interface réseau	83
4.2.6.1.4.2	Cas 2 : Machines avec plusieurs interfaces réseau	83
4.2.6.1.4.3	Vérification de la génération des <i>hostvars</i>	83
4.2.6.1.5	Déploiement	83
4.2.7	Elements <i>extras</i> de l'installation	84
4.2.7.1	Configuration des <i>extras</i>	84
4.2.7.2	Déploiement des <i>extras</i>	86
4.2.7.2.1	ihm-recette	86
4.2.7.2.2	<i>Extras</i> complet	87
5	Procédures de mise à jour de la configuration	88
5.1	Cas d'une modification du nombre de tenants	88
5.2	Cas d'une modification des paramètres JVM	88
5.3	Cas de la mise à jour des <i>griffins</i>	89
6	Post installation	90
6.1	Validation du déploiement	90
6.1.1	Sécurisation du fichier <i>vault_pass.txt</i>	90
6.1.2	Validation manuelle	90
6.1.3	Validation via Consul	90
6.1.4	Post-installation : administration fonctionnelle	91
6.2	Sauvegarde des éléments d'installation	91
6.3	Troubleshooting	91
6.3.1	Erreur au chargement des <i>index template</i> kibana	91
6.3.2	Erreur au chargement des tableaux de bord Kibana	92
6.4	Retour d'expérience / cas rencontrés	92
6.4.1	Crash rsyslog, code killed, signal : BUS	92
6.4.2	Mongo-express ne se connecte pas à la base de données associée	92
6.4.3	Elasticsearch possède des shard non alloués (état « UNASSIGNED »)	92
6.4.4	Elasticsearch possède des shards non initialisés (état « INITIALIZING »)	93
6.4.5	Elasticsearch est dans l'état « <i>read-only</i> »	93
6.4.6	MongoDB semble lent	94
6.4.7	Les shards de MongoDB semblent mal équilibrés	94
6.4.8	L'importation initiale (profil de sécurité, certificats) retourne une erreur	95
6.4.9	Problème d'ingest et/ou d'access	95
7	Montée de version	96
8	Annexes	97
8.1	Vue d'ensemble de la gestion des certificats	97
8.1.1	Liste des suites cryptographiques & protocoles supportés par VITAM	97
8.1.2	Vue d'ensemble de la gestion des certificats	98

8.1.3	Description de l'arborescence de la PKI	98
8.1.4	Description de l'arborescence du répertoire <code>deployment/environments/certs</code> . . .	100
8.1.5	Description de l'arborescence du répertoire <code>deployment/environments/keystores</code>	101
8.1.6	Fonctionnement des scripts de la PKI	101
8.2	Spécificités des certificats	101
8.2.1	Cas des certificats serveur	102
8.2.1.1	Généralités	102
8.2.1.2	Noms DNS des serveurs https VITAM	102
8.2.2	Cas des certificats client	103
8.2.3	Cas des certificats d'horodatage	103
8.2.4	Cas des certificats des services de stockage objets	103
8.3	Cycle de vie des certificats	103
8.4	Ansible & SSH	105
8.4.1	Authentification du compte utilisateur utilisé pour la connexion SSH	105
8.4.1.1	Par clé SSH avec passphrase	105
8.4.1.2	Par login/mot de passe	105
8.4.1.3	Par clé SSH sans passphrase	105
8.4.2	Authentification des hôtes	105
8.4.3	Elévation de privilèges	105
8.4.3.1	Par sudo avec mot de passe	106
8.4.3.2	Par su	106
8.4.3.3	Par sudo sans mot de passe	106
8.4.3.4	Déjà Root	106

1.1 Objectif de ce document

Ce document a pour but de fournir à une équipe d'exploitants de la solution logicielle *VITAM* les procédures et informations utiles et nécessaires pour l'installation de la solution logicielle.

Il s'adresse aux personnes suivantes :

- Les architectes techniques des projets désirant intégrer la solution logicielle *VITAM* ;
- Les exploitants devant installer la solution logicielle *VITAM*.

CHAPITRE 2

Rappels

2.1 Information concernant les licences

La solution logicielle *VITAM* est publiée sous la licence [CeCILL 2.1](#)¹ ; la documentation associée (comprenant le présent document) est publiée sous [Licence Ouverte V2.0](#)².

Les clients externes java de solution *VITAM* sont publiés sous la licence [CeCILL-C](#)³ ; la documentation associée (comprenant le présent document) est publiée sous [Licence Ouverte V2.0](#)⁴.

2.2 Documents de référence

2.2.1 Documents internes

Tableau 1 – Documents de référence VITAM

Nom	Lien
<i>DAT</i>	http://www.programmevitam.fr/ressources/DocCourante/html/archi
<i>DIN</i>	http://www.programmevitam.fr/ressources/DocCourante/html/installation
<i>DEX</i>	http://www.programmevitam.fr/ressources/DocCourante/html/exploitation
<i>DMV</i>	http://www.programmevitam.fr/ressources/DocCourante/html/migration
Release notes	https://github.com/ProgrammeVitam/vitam/releases/latest

https://cecill.info/licences/Licence_CeCILL_V2.1-fr.html

<https://www.etalab.gouv.fr/wp-content/uploads/2017/04/ETALAB-Licence-Ouverte-v2.0.pdf>

https://cecill.info/licences/Licence_CeCILL-C_V1-fr.html

<https://www.etalab.gouv.fr/wp-content/uploads/2017/04/ETALAB-Licence-Ouverte-v2.0.pdf>

2.2.2 Référentiels externes

2.3 Glossaire

API *Application Programming Interface*

AU *Archive Unit*, unité archivistique

BDD Base De Données

BDO *Binary DataObject*

CA *Certificate Authority*, autorité de certification

CAS Content Adressable Storage

CCFN Composant Coffre Fort Numérique

CN Common Name

COTS Component Off The shelf ; il s'agit d'un composant « sur étagère », non développé par le projet *VITAM*, mais intégré à partir d'un binaire externe. Par exemple : MongoDB, ElasticSearch.

CRL *Certificate Revocation List* ; liste des identifiants des certificats qui ont été révoqués ou invalidés et qui ne sont donc plus dignes de confiance. Cette norme est spécifiée dans les RFC 5280 et RFC 6818.

CRUD *create, read, update, and delete*, s'applique aux opérations dans une base de données MongoDB

DAT Dossier d'Architecture Technique

DC Data Center

DEX Dossier d'EXploitation

DIN Dossier d'INstallation

DIP *Dissemination Information Package*

DMV Documentation de Montées de Version

DNS *Domain Name System*

DNSSEC *Domain Name System Security Extensions* est un protocole standardisé par l'IETF permettant de résoudre certains problèmes de sécurité liés au protocole DNS. Les spécifications sont publiées dans la RFC 4033 et les suivantes (une version antérieure de DNSSEC n'a eu aucun succès). [Définition DNSSEC](#)⁵

DSL *Domain Specific Language*, langage dédié pour le requêtage de VITAM

DUA Durée d'Utilité Administrative

EBIOS Méthode d'évaluation des risques en informatique, permettant d'apprécier les risques Sécurité des systèmes d'information (entités et vulnérabilités, méthodes d'attaques et éléments menaçants, éléments essentiels et besoins de sécurité. . .), de contribuer à leur traitement en spécifiant les exigences de sécurité à mettre en place, de préparer l'ensemble du dossier de sécurité nécessaire à l'acceptation des risques et de fournir les éléments utiles à la communication relative aux risques. Elle est compatible avec les normes ISO 13335 (GMITS), ISO 15408 (critères communs) et ISO 17799

EAD Description archivistique encodée

ELK *Elasticsearch Logstash Kibana*

FIP *Floating IP*

GOT Groupe d'Objet Technique

IHM Interface Homme Machine

IP *Internet Protocol*

IsaDG Norme générale et internationale de description archivistique

JRE *Java Runtime Environment* ; il s'agit de la machine virtuelle Java permettant d'y exécuter les programmes compilés pour.

https://fr.wikipedia.org/wiki/Domain_Name_System_Security_Extensions

JVM *Java Virtual Machine* ; Cf. *JRE*

LAN *Local Area Network*, réseau informatique local, qui relie des ordinateurs dans une zone limitée

LFC *LiFe Cycle*, cycle de vie

LTS *Long-term support*, support à long terme : version spécifique d'un logiciel dont le support est assuré pour une période de temps plus longue que la normale.

M2M *Machine To Machine*

MitM L'attaque de l'homme du milieu (HDM) ou *man-in-the-middle attack* (MITM) est une attaque qui a pour but d'intercepter les communications entre deux parties, sans que ni l'une ni l'autre ne puisse se douter que le canal de communication entre elles a été compromis. Le canal le plus courant est une connexion à Internet de l'internaute lambda. L'attaquant doit d'abord être capable d'observer et d'intercepter les messages d'une victime à l'autre. L'attaque « homme du milieu » est particulièrement applicable dans la méthode d'échange de clés Diffie-Hellman, quand cet échange est utilisé sans authentification. Avec authentification, Diffie-Hellman est en revanche invulnérable aux écoutes du canal, et est d'ailleurs conçu pour cela. [Explication](#)⁶

MoReq *Modular Requirements for Records System*, recueil d'exigences pour l'organisation de l'archivage, élaboré dans le cadre de l'Union européenne.

NoSQL Base de données non-basée sur un paradigme classique des bases relationnelles. [Définition NoSQL](#)⁷

NTP *Network Time Protocol*

OAIS *Open Archival Information System*, acronyme anglais pour Systèmes de transfert des informations et données spatiales – Système ouvert d'archivage d'information (SOAI) - Modèle de référence.

OOM Aussi appelé *Out-Of-Memory Killer* ; mécanisme de la dernière chance incorporé au noyau Linux, en cas de dépassement de la capacité mémoire

OS *Operating System*, système d'exploitation

OWASP *Open Web Application Security Project*, communauté en ligne de façon libre et ouverte à tous publiant des recommandations de sécurisation Web et de proposant aux internautes, administrateurs et entreprises des méthodes et outils de référence permettant de contrôler le niveau de sécurisation de ses applications Web

PDMA Perte de Données Maximale Admissible ; il s'agit du pourcentage de données stockées dans le système qu'il est acceptable de perdre lors d'un incident de production.

PKI Une infrastructure à clés publiques (ICP) ou infrastructure de gestion de clés (IGC) ou encore Public Key Infrastructure (PKI), est un ensemble de composants physiques (des ordinateurs, des équipements cryptographiques logiciels ou matériel type HSM ou encore des cartes à puces), de procédures humaines (vérifications, validation) et de logiciels (système et application) en vue de gérer le cycle de vie des certificats numériques ou certificats électroniques. [Définition PKI](#)⁸

PCA Plan de Continuité d'Activité

PRA Plan de Reprise d'Activité

REST *REpresentational State Transfer* : type d'architecture d'échanges. Appliqué aux services web, en se basant sur les appels http standard, il permet de fournir des API dites « RESTful » qui présentent un certain nombre d'avantages en termes d'indépendance, d'universalité, de maintenabilité et de gestion de charge. [Définition REST](#)⁹

RGAA Référentiel Général d'Accessibilité pour les Administrations

RGI Référentiel Général d'Interopérabilité

RPM *Red Hat Package Manager* ; il s'agit du format de packets logiciels nativement utilisé par les distributions CentOS (entre autres)

SAE Système d'Archivage Électronique

SEDA Standard d'Échange de Données pour l'Archivage

https://fr.wikipedia.org/wiki/Attaque_de_l'homme_du_milieu

<https://fr.wikipedia.org/wiki/NoSQL>

https://fr.wikipedia.org/wiki/Infrastructure_%C3%A0_cl%C3%A9s_publicques

https://fr.wikipedia.org/wiki/Representational_state_transfer

SGBD *Système de Gestion de Base de Données*

SGBDR *Système de Gestion de Base de Données Relationnelle*

SIA *Système d'Informations Archivistique*

SIEM *Security Information and Event Management*

SIP *Submission Information Package*

SSH *Secure SHell*

Swift *OpenStack Object Store project*

TLS *Transport Layer Security*

TNR *Tests de Non-Régression*

TTL *Time To Live*, indique le temps pendant lequel une information doit être conservée, ou le temps pendant lequel une information doit être gardée en cache

UDP *User Datagram Protocol*, protocole de datagramme utilisateur, un des principaux protocoles de télécommunication utilisés par Internet. Il fait partie de la couche transport du modèle OSI

UID *User IDentification*

VITAM *Valeurs Immatérielles Transférées aux Archives pour Mémoire*

VM *Virtual Machine*

WAF *Web Application Firewall*

WAN *Wide Area Network*, réseau informatique couvrant une grande zone géographique, typiquement à l'échelle d'un pays, d'un continent, ou de la planète entière

Prérequis à l'installation

3.1 Expertises requises

Les équipes en charge du déploiement et de l'exploitation de la solution logicielle *VITAM* devront disposer en interne des compétences suivantes :

- connaissance d'ansible en tant qu'outil de déploiement automatisé ;
- connaissance de Consul en tant qu'outil de découverte de services ;
- maîtrise de MongoDB et ElasticSearch par les administrateurs de bases de données.

3.2 Pré-requis plate-forme

Les pré-requis suivants sont nécessaires :

3.2.1 Base commune

- Tous les serveurs hébergeant la solution logicielle *VITAM* doivent être synchronisés sur un serveur de temps (protocole *NTP*, pas de *stratum* 10)
- Disposer de la solution de déploiement basée sur ansible

Le déploiement est orchestré depuis un poste ou serveur d'administration ; les pré-requis suivants doivent y être présents :

- packages nécessaires :
 - **ansible** (version **2.9** minimale et conseillée ; se référer à la [documentation ansible](http://docs.ansible.com/ansible/latest/intro_installation.html)¹⁰ pour la procédure d'installation)
 - **openssh-client** (client SSH utilisé par ansible)

¹⁰http://docs.ansible.com/ansible/latest/intro_installation.html

- **JRE OpenJDK 11** et **openssl** (du fait de la génération de certificats / *stores*, l'utilitaire `keytool` est nécessaire)
- un accès ssh vers un utilisateur d'administration avec élévation de privilèges vers les droits `root`, `vitam`, `vitamdb` (les comptes `vitam` et `vitamdb` sont créés durant le déploiement) sur les serveurs cibles.
- Le compte utilisé sur le serveur d'administration doit avoir confiance dans les serveurs sur lesquels la solution logicielle *VITAM* doit être installée (fichier `~/.ssh/known_hosts` correctement renseigné)

Note : Se référer à la [documentation d'usage](#)¹¹ pour les procédures de connexion aux machines-cibles depuis le serveur ansible.

Prudence : Les adresses *IP* des machines sur lesquelles la solution logicielle *VITAM* sera installée ne doivent pas changer d'adresse IP au cours du temps. En cas de changement d'adresse IP, la plateforme ne pourra plus fonctionner.

Prudence : Dans le cadre de l'installation des packages « extra », il est nécessaire, pour les partitions hébergeant des conteneurs docker (mongo-express, head), qu'elles aient un accès internet (installation du paquet officiel `docker`, récupération des images).

Prudence : Dans le cadre de l'installation des packages « extra », il est nécessaire, pour les partitions hébergeant le composant `ihm-recette`, qu'elles aient un accès internet (installation du *repository* et installation du *package* `git-lfs` ; récupération des *TNR* depuis un dépôt git).

Avertissement : Dans le cas d'une installation du composant `vitam-offer` en `filesystem-hash`, il est fortement recommandé d'employer un système de fichiers `xfs` pour le stockage des données. Se référer au *DAT* pour connaître la structuration des *filesystems* dans la solution logicielle *VITAM*. En cas d'utilisation d'un autre type, s'assurer que le filesystem possède/gère bien l'option `user_xattr`.

Avertissement : Dans le cas d'une installation du composant `vitam-offer` en `tape-library`, il est fortement recommandé d'installer au préalable sur les machines cible associées les paquets pour les commandes `mt`, `mtx` et `dd`. Ces composants doivent également apporter le groupe système `tape`. Se reporter également à *Librairie de cartouches pour offre froide* (page 9).

3.2.2 PKI

La solution logicielle *VITAM* nécessite des certificats pour son bon fonctionnement (cf. *DAT* pour la liste des secrets et *Vue d'ensemble de la gestion des certificats* (page 97) pour une vue d'ensemble de leur usage.) La gestion de ces certificats, par le biais d'une ou plusieurs *PKI*, est à charge de l'équipe d'exploitation. La mise à disposition des certificats et des chaînes de validation *CA*, placés dans les répertoires de déploiement adéquats, est un pré-requis à tout déploiement en production de la solution logicielle *VITAM*.

Voir aussi :

http://docs.ansible.com/ansible/latest/intro_getting_started.html

Veuillez vous référer à la section [Vue d'ensemble de la gestion des certificats](#) (page 97) pour la liste des certificats nécessaires au déploiement de la solution VITAM, ainsi que pour leurs répertoires de déploiement.

3.2.3 Systèmes d'exploitation

Seules deux distributions Linux suivantes sont supportées à ce jour :

- CentOS 7
- Debian 10 (buster)

SELinux doit être configuré en mode `permissive` ou `disabled`. Toutefois depuis la release R13, la solution logicielle *VITAM* prend désormais en charge l'activation de SELinux sur le périmètre du composant worker et des processus associés aux *griffins* (greffons de préservation).

Note : En cas de changement de mode SELinux, redémarrer les machines pour la bonne prise en compte de la modification avant de lancer le déploiement.

Prudence : En cas d'installation initiale, les utilisateurs et groupes systèmes (noms et *UID*) utilisés par VITAM (et listés dans le *DAT*) ne doivent pas être présents sur les serveurs cible. Ces comptes sont créés lors de l'installation de VITAM et gérés par VITAM.

3.2.3.1 Déploiement sur environnement CentOS

- Disposer d'une plate-forme Linux CentOS 7 installée selon la répartition des services souhaités. En particulier, ces serveurs doivent avoir :
 - une configuration de temps synchronisée (ex : en récupérant le temps à un serveur centralisé)
 - Des autorisations de flux conformément aux besoins décrits dans le *DAT*
 - une configuration des serveurs de noms correcte (cette configuration sera surchargée lors de l'installation)
 - un accès à un dépôt (ou son miroir) CentOS 7 (base et extras) et EPEL 7
- Disposer des binaires VITAM : paquets *RPM* de VITAM (vitam-product) ainsi que les paquets d'éditeurs tiers livrés avec VITAM (vitam-external)
- Disposer, si besoin, des binaires pour l'installation des *griffins*

3.2.3.2 Déploiement sur environnement Debian

- Disposer d'une plate-forme Linux Debian « buster » installée selon la répartition des services souhaitée. En particulier, ces serveurs doivent avoir :
 - une configuration de temps synchronisée (ex : en récupérant le temps à un serveur centralisé)
 - Des autorisations de flux conformément aux besoins décrits dans le *DAT*
 - une configuration des serveurs de noms correcte (cette configuration sera surchargée lors de l'installation)
 - un accès à un dépôt (ou son miroir) Debian (base et extras) et buster-backports
 - un accès internet, car le dépôt docker sera ajouté
- Disposer des binaires VITAM : paquets deb de VITAM (vitam-product) ainsi que les paquets d'éditeurs tiers livrés avec VITAM (vitam-external)
- Disposer, si besoin, des binaires pour l'installation des *griffins*

Avertissement : Pour l'installation des *packages* mongoDB, il est nécessaire de mettre à disposition le *package* `libcurl3` présent en *stretch* uniquement (le *package* `libcurl4` sera désinstallé).

Avertissement : Le *package* `curl` est installé depuis les dépôts *stretch*.

3.2.3.3 Présence d'un agent antivirus

Dans le cas de partitions sur lesquelles un agent antivirus est déjà configuré (typiquement, *golden image*), il est recommandé de positionner une exception sur l'arborescence `/vitam` et les sous-arborescences, hormis la partition hébergeant le composant `ingest-external` (emploi d'un agent antivirus en prérequis des *ingest* ; se reporter à *Rétention liée aux logback* (page 52)).

3.2.4 Matériel

Les prérequis matériel sont définis dans le *DAT* ; à l'heure actuelle, le minimum recommandé pour la solution Vitam est 2 CPUs. Il également est recommandé de prévoir (paramétrage par défaut à l'installation) 512Mo de RAM disponible par composant applicatif *VITAM* installé sur chaque machine (hors *elasticsearch* et *mongo*).

Concernant l'espace disque, à l'heure actuelle, aucun pré-requis n'a été défini ; cependant, sont à prévoir par la suite des espaces de stockage conséquents pour les composants suivants :

- offer
- solution de centralisation des logs (*cluster elasticsearch* de log)
- workspace
- worker (temporairement, lors du traitement de chaque fichier à traiter)
- *cluster elasticsearch* des données *VITAM*

L'arborescence associée sur les partitions associées est : `/vitam/data/<composant>`

3.2.5 Librairie de cartouches pour offre froide

Des prérequis sont à réunir pour utiliser l'offre froide de stockage « tape-library » définie dans le *DAT*.

- La librairie de cartouches doit être opérationnelle et chargée en cartouches.
- La librairie et les lecteurs doivent déjà être disponibles sur la machine devant supporter une instance de ce composant. La commande `ls SCSI -g` peut permettre de vérifier si des périphériques sont détectés.

3.3 Questions préparatoires

La solution logicielle *VITAM* permet de répondre à différents besoins.

Afin d'y répondre de la façon la plus adéquate et afin de configurer correctement le déploiement *VITAM*, il est nécessaire de se poser en amont les questions suivantes :

- **Questions techniques :**
 - Topologie de déploiement et dimensionnement de l'environnement ?
 - Espace de stockage (volumétrie métier cible, technologies d'offres de stockage, nombre d'offres, etc.) ?

- Sécurisation des flux http (récupération des clés publiques des services versants, sécurisation des flux d'accès aux offres, etc.) ?
- **Questions liées au métier :**
 - Nombre de tenants souhaités (hormis les tenants 0 et 1 qui font respectivement office de tenant « blanc » et de tenant d'administration) ?
 - Niveau de classification (la plate-forme est-elle « Secret Défense » ?)
 - Modalités d'indexation des règles de gestion des unités archivistiques (autrement dit, sur quels tenants le recalcul des `inheritedRules` doit-il être fait complètement / partiellement) ?
 - Greffons de préservations (*griffins*) nécessaires ?
 - Fréquence de calcul de l'état des fonds symboliques souhaitée ?
 - Définition des habilitations (profil de sécurité, contextes applicatifs, ...) ?
 - Modalités de gestion des données de référence (maître/esclave) pour chaque tenant ?

Par la suite, les réponses apportées vous permettront de configurer le déploiement par la définition des paramètres ansible.

3.4 Récupération de la version

3.4.1 Utilisation des dépôts *open-source*

Les scripts de déploiement de la solution logicielle *VITAM* sont disponibles dans le [dépôt github VITAM](#)¹², dans le répertoire `deployment`.

Les binaires de la solution logicielle *VITAM* sont disponibles sur des dépôts *VITAM* publics indiqués ci-dessous par type de *package* ; ces dépôts doivent être correctement configurés sur la plate-forme cible avant toute installation.

3.4.1.1 *Repository* pour environnement CentOS

Sur les partitions cibles, configurer le fichier `/etc/yum.repos.d/vitam-repositories.repo` (remplacer `<branche_vitam>` par le nom de la branche de support à installer) comme suit

```
[programmevitam-vitam-rpm-release-product]
name=programmevitam-vitam-rpm-release-product
baseurl=http://download.programmevitam.fr/vitam_repository/<vitam_version>/rpm/vitam-
↳product/
gpgcheck=0
repo_gpgcheck=0
enabled=1

[programmevitam-vitam-rpm-release-external]
name=programmevitam-vitam-rpm-release-external
baseurl=http://download.programmevitam.fr/vitam_repository/<vitam_version>/rpm/vitam-
↳external/
gpgcheck=0
repo_gpgcheck=0
enabled=1
```

Note : remplacer `<vitam_version>` par la version à déployer.

<https://github.com/ProgrammeVitam/vitam>

3.4.1.1.1 Cas de *griffins*

Un dépôt supplémentaire est à paramétrer pour pouvoir dérouler l'installation des *griffins*

```
[programmevitam-vitam-griffins]
name=programmevitam-vitam-griffins
baseurl=http://download.programmevitam.fr/vitam_griffins/<version_griffins>/rpm/
gpgcheck=0
repo_gpgcheck=0
enabled=1
```

Note : remplacer <version_griffins> par la version à déployer.

3.4.1.2 *Repository* pour environnement Debian

Sur les partitions cibles, configurer le fichier `/etc/apt/sources.list.d/vitam-repositories.list` comme suit

```
deb [trusted=yes] http://download.programmevitam.fr/vitam_repository/<vitam_version>/
↳deb/vitam-product/ ./
deb [trusted=yes] http://download.programmevitam.fr/vitam_repository/<vitam_version>/
↳deb/vitam-external/ ./
```

Note : remplacer <vitam_version> par la version à déployer.

3.4.1.2.1 Cas de *griffins*

Un dépôt supplémentaire est à paramétrer pour pouvoir dérouler l'installation des *griffins*

```
deb [trusted=yes] http://download.programmevitam.fr/vitam_griffins/<version_griffins>/
↳deb/ ./
```

Note : remplacer <version_griffins> par la version à déployer.

3.4.2 Utilisation du package global d'installation

Note : Le *package* global d'installation n'est pas présent dans les dépôts publics.

Le *package* global d'installation contient les livrables binaires (dépôts CentOS, Debian, Maven)

Sur la machine « *ansible* » dédiée au déploiement de la solution logicielle *VITAM*, décompresser le package (au format `tar.gz`).

Pour l'installation des *griffins*, il convient de récupérer, puis décompresser, le package associé (au format `zip`).

Sur le *repository* « *VITAM* », récupérer également depuis le fichier d'extension `tar.gz` les binaires d'installation (`rpm` pour CentOS ; `deb` pour Debian) et les faire prendre en compte par le *repository*.

Sur le *repository* « *griffins* », récupérer également depuis le fichier d'extension `zip` les binaires d'installation (rpm pour CentOS ; deb pour Debian) et les faire prendre en compte par le *repository*.

Procédures d'installation / mise à jour

4.1 Vérifications préalables

Tous les serveurs cibles doivent avoir accès aux dépôts de binaires contenant les paquets de la solution logicielle *VITAM* et des composants externes requis pour l'installation. Les autres éléments d'installation (*playbook* ansible, ...) doivent être disponibles sur la machine ansible orchestrant le déploiement de la solution.

4.2 Procédures

4.2.1 Cinématique de déploiement

La cinématique de déploiement d'un site *VITAM* est représentée dans le schéma suivant :

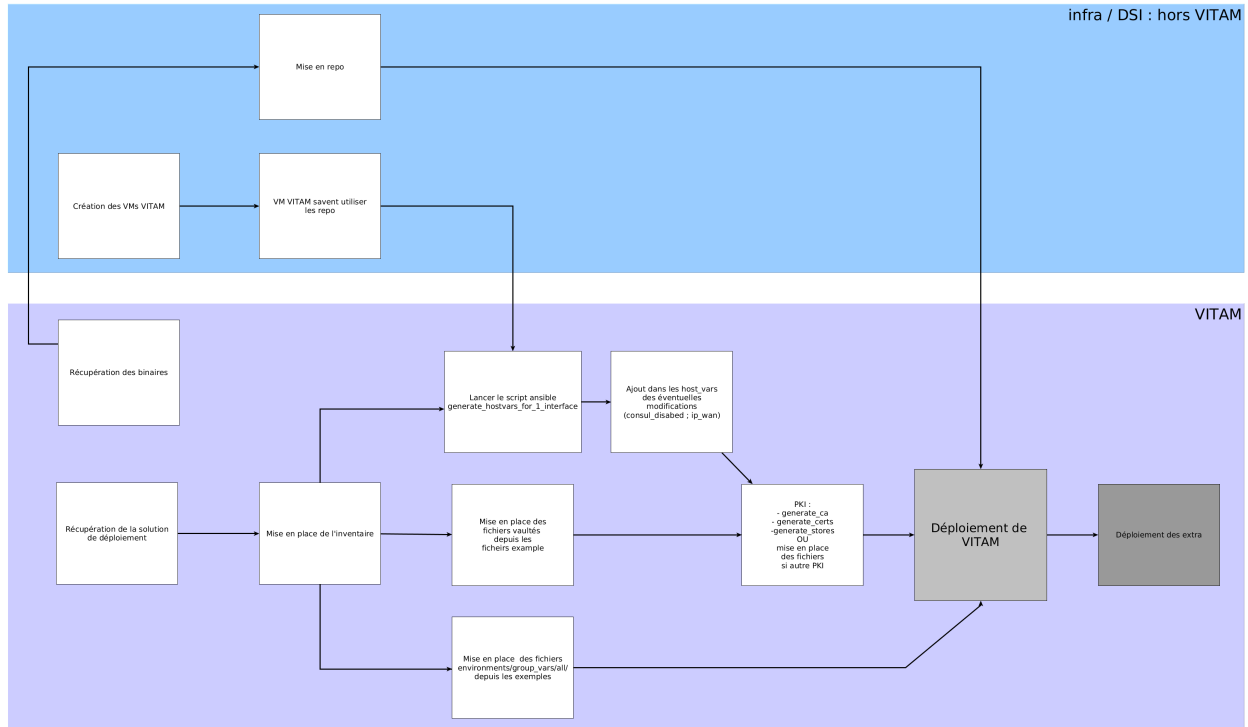


Fig. 1 – Cinématique de déploiement

4.2.2 Cas particulier d'une installation multi-sites

4.2.2.1 Procédure d'installation

Dans le cadre d'une installation multi-sites, il est nécessaire de déployer la solution logicielle *VITAM* sur le site secondaire dans un premier temps, puis déployer le site *production*.

Il faut paramétrer correctement un certain nombre de variables ansible pour chaque site :

4.2.2.1.1 vitam_site_name

Fichier : `deployment/environments/hosts.<my_env>`

Cette variable sert à définir le nom du site. Elle doit être différente sur chaque site.

4.2.2.1.2 primary_site

Fichier : `deployment/environments/hosts.<my_env>`

Cette variable sert à définir si le site est primaire ou non. Sur VITAM installé en mode multi site, un seul des sites doit avoir la valeur *primary_site* à *true*. Sur les sites secondaires (*primary_site* : *false*), certains composants ne seront pas démarrés et apparaîtront donc en orange sur l'*IHM* de consul. Certains timers systemd seront en revanche démarrés pour mettre en place la reconstruction au fil de l'eau, par exemple.

4.2.2.1.3 consul_remote_sites

Fichier : `deployment/environments/group_vars/all/cots_vars.yml`

Cette variable sert à référencer la liste des *Consul Server* des sites distants, à celui que l'on configure.

Exemple de configuration pour une installation avec 3 sites.

Site 1 :

```
consul_remote_sites:
  - dc2:
    wan: ["dc2-host-1", "dc2-host-2", "dc2-host-3"]
  - dc3:
    wan: ["dc3-host-1", "dc3-host-2", "dc3-host-3"]
```

Site 2 :

```
consul_remote_sites:
  - dc1:
    wan: ["dc1-host-1", "dc1-host-2", "dc1-host-3"]
  - dc3:
    wan: ["dc3-host-1", "dc3-host-2", "dc3-host-3"]
```

Site 3 :

```
consul_remote_sites:
  - dc1:
    wan: ["dc1-host-1", "dc1-host-2", "dc1-host-3"]
  - dc2:
    wan: ["dc2-host-1", "dc2-host-2", "dc2-host-3"]
```

Il faut également prévoir de déclarer, lors de l'installation de chaque site distant, la variable `ip_wan` pour les partitions hébergeant les serveurs Consul (groupe ansible `hosts_consul_server`) et les offres de stockage (groupe ansible `hosts_storage_offer_default`, considérées distantes par le site primaire). Ces ajouts sont à faire dans `environments/host_vars/<nom partition>`.

Exemple

```
ip_service: 172.17.0.10
ip_admin: 172.19.0.10
ip_wan: 10.2.64.3
```

Ainsi, à l'usage, le composant `storage` va appeler les services `offer`. Si le service est « hors domaine » (déclaration explicite `<service>.<datacenterdistant>.service.<domaineconsul>`), un échange d'information entre « datacenters » Consul est réalisé et la valeur de `ip_wan` est fournie pour l'appel au service distant.

4.2.2.1.4 vitam_offers

Fichier : `deployment/environments/group_vars/all/offer_opts.yml`

Cette variable référence toutes les offres disponibles sur la totalité des sites VITAM.

Exemple :

```
vitam_offers:
  offer-fs-1:
```

(suite sur la page suivante)

```

    provider: filesystem-hash
offer-fs-2:
    provider: filesystem-hash
offer-fs-3:
    provider: filesystem-hash

```

4.2.2.1.5 vitam_strategy

Fichier : `deployment/environments/group_vars/all/offer_opts.yml`

Cette variable référence la stratégie de stockage de plateforme *default* sur le site courant. Si l'offre se situe sur un site distant, il est nécessaire de préciser le nom du site sur lequel elle se trouve comme dans l'exemple ci-dessous. Il est fortement conseillé de prendre comme offre référente une des offres locale au site. Les sites secondaires doivent uniquement écrire sur leur(s) offre(s) locale(s).

Exemple pour le site 1 (site primaire) :

```

vitam_strategy:
  - name: offer-fs-1
    referent: true
    # status: INACTIVE (valeur par défaut: ACTIVE)
    # id: idoffre # OPTIONAL, if used, MUST BE UNIQUE & same on each site
    # asyncRead: true # ONLY ENABLE WHEN tape-library
    # vitam_site_name: prod-dc2 # Optional, needed only when call to distant site_
    ↳ (indicate distant vitam_site_name)
  - name: offer-fs-2
    referent: false
    vitam_site_name: site2
    # status: INACTIVE (valeur par défaut: ACTIVE)
    # id: idoffre # OPTIONAL, if used, MUST BE UNIQUE & same on each site
    # asyncRead: true # ONLY ENABLE WHEN tape-library
    # vitam_site_name: prod-dc2 # Optional, needed only when call to distant site_
    ↳ (indicate distant vitam_site_name)
  - name: offer-fs-3
    referent: false
    vitam_site_name: site3
    # status: INACTIVE (valeur par défaut: ACTIVE)
    # id: idoffre # OPTIONAL, if used, MUST BE UNIQUE & same on each site
    # asyncRead: true # ONLY ENABLE WHEN tape-library
    # vitam_site_name: prod-dc2 # Optional, needed only when call to distant site_
    ↳ (indicate distant vitam_site_name)

```

Exemple pour le site 2 (site secondaire) :

```

vitam_strategy:
  - name: offer-fs-2
    referent: true
    # id: idoffre # OPTIONAL, if used, MUST BE UNIQUE & same on each site
    # status: INACTIVE (valeur par défaut: ACTIVE)
    # asyncRead: true # ONLY ENABLE WHEN tape-library
    # vitam_site_name: prod-dc2 # Optional, needed only when call to distant site_
    ↳ (indicate distant vitam_site_name)

```

Exemple pour le site 3 (site secondaire) :

```

vitam_strategy:
- name: offer-fs-3
  referent: true
  # id: idoffre # OPTIONAL, if used, MUST BE UNIQUE & same on each site
  # status: INACTIVE (valeur par défaut: ACTIVE)
  # asyncRead: true # ONLY ENABLE WHEN tape-library
  # vitam_site_name: prod-dc2 # Optional, needed only when call to distant site_
↪(indicate distant vitam_site_name)

```

4.2.2.1.6 other_strategies

Fichier: deployment/environments/group_vars/all/offer_opts.yml

Cette variable référence les stratégies de stockage additionnelles sur le site courant. **Elles ne sont déclarées et utilisées que dans le cas du multi-stratégies.** Si l'offre se situe sur un site distant, il est nécessaire de préciser le nom du site sur lequel elle se trouve comme dans l'exemple ci-dessous. Les sites secondaires doivent uniquement écrire sur leur(s) offre(s) locale(s).

Les offres correspondant à l'exemple other_strategies sont les suivantes :

```

vitam_offers:
  offer-fs-1:
    provider: filesystem-hash
  offer-fs-2:
    provider: filesystem-hash
  offer-fs-3:
    provider: filesystem-hash
  offer-s3-1:
    provider: amazon-s3-v1
  offer-s3-2:
    provider: amazon-s3-v1
  offer-s3-3:
    provider: amazon-s3-v1

```

Exemple pour le site 1 (site primaire) :

```

other_strategies:
  metadata:
    - name: offer-fs-1
      referent: true
      # status: INACTIVE (valeur par défaut: ACTIVE)
      # id: idoffre # OPTIONAL, if used, MUST BE UNIQUE & same on each site
      # asyncRead: true # ONLY ENABLE WHEN tape-library
      # vitam_site_name: site1 # Optional, needed only when call to distant site_
↪(indicate distant vitam_site_name)
    - name: offer-fs-2
      referent: false
      vitam_site_name: site2
      # status: INACTIVE (valeur par défaut: ACTIVE)
      # id: idoffre # OPTIONAL, if used, MUST BE UNIQUE & same on each site
      # asyncRead: true # ONLY ENABLE WHEN tape-library
    - name: offer-fs-3
      referent: false
      vitam_site_name: site3
      # status: INACTIVE (valeur par défaut: ACTIVE)
      # id: idoffre # OPTIONAL, if used, MUST BE UNIQUE & same on each site

```

(suite sur la page suivante)

(suite de la page précédente)

```

# asyncRead: true # ONLY ENABLE WHEN tape-library
- name: offer-s3-1
referent: false
# status: INACTIVE (valeur par défaut: ACTIVE)
# id: idoffre # OPTIONAL, if used, MUST BE UNIQUE & same on each site
# asyncRead: true # ONLY ENABLE WHEN tape-library
# vitam_site_name: site1 # Optional, needed only when call to distant site_
→ (indicate distant vitam_site_name)
- name: offer-s3-2
referent: false
vitam_site_name: site2
# status: INACTIVE (valeur par défaut: ACTIVE)
# id: idoffre # OPTIONAL, if used, MUST BE UNIQUE & same on each site
# asyncRead: true # ONLY ENABLE WHEN tape-library
- name: offer-s3-3
referent: false
vitam_site_name: site3
# status: INACTIVE (valeur par défaut: ACTIVE)
# id: idoffre # OPTIONAL, if used, MUST BE UNIQUE & same on each site
# asyncRead: true # ONLY ENABLE WHEN tape-library
binary:
- name: offer-s3-1
referent: false
# status: INACTIVE (valeur par défaut: ACTIVE)
# id: idoffre # OPTIONAL, if used, MUST BE UNIQUE & same on each site
# asyncRead: true # ONLY ENABLE WHEN tape-library
# vitam_site_name: site1 # Optional, needed only when call to distant site_
→ (indicate distant vitam_site_name)
- name: offer-s3-2
referent: false
vitam_site_name: site2
# status: INACTIVE (valeur par défaut: ACTIVE)
# id: idoffre # OPTIONAL, if used, MUST BE UNIQUE & same on each site
# asyncRead: true # ONLY ENABLE WHEN tape-library
- name: offer-s3-3
referent: false
vitam_site_name: site3
# status: INACTIVE (valeur par défaut: ACTIVE)
# id: idoffre # OPTIONAL, if used, MUST BE UNIQUE & same on each site
# asyncRead: true # ONLY ENABLE WHEN tape-library

```

Exemple pour le site 2 (site secondaire) :

```

other_strategies:
  metadata:
    - name: offer-fs-2
      referent: true
      # id: idoffre # OPTIONAL, if used, MUST BE UNIQUE & same on each site
      # status: INACTIVE (valeur par défaut: ACTIVE)
      # asyncRead: true # ONLY ENABLE WHEN tape-library
      # vitam_site_name: site2 # Optional, needed only when call to distant site_
→ (indicate distant vitam_site_name)
    - name: offer-s3-2
      referent: false
      # status: INACTIVE (valeur par défaut: ACTIVE)
      # id: idoffre # OPTIONAL, if used, MUST BE UNIQUE & same on each site
      # asyncRead: true # ONLY ENABLE WHEN tape-library

```

(suite sur la page suivante)

(suite de la page précédente)

```

    # vitam_site_name: site2 # Optional, needed only when call to distant site_
↪(indicate distant vitam_site_name)
    binary:
      - name: offer-s3-2
        referent: false
        # status: INACTIVE (valeur par défaut: ACTIVE)
        # id: idoffre # OPTIONAL, if used, MUST BE UNIQUE & same on each site
        # asyncRead: true # ONLY ENABLE WHEN tape-library
        # vitam_site_name: site2 # Optional, needed only when call to distant site_
↪(indicate distant vitam_site_name)

```

Exemple pour le site 3 (site secondaire) :

```

other_strategies:
  metadata:
    - name: offer-fs-3
      referent: true
      # id: idoffre # OPTIONAL, if used, MUST BE UNIQUE & same on each site
      # status: INACTIVE (valeur par défaut: ACTIVE)
      # asyncRead: true # ONLY ENABLE WHEN tape-library
      # vitam_site_name: site3 # Optional, needed only when call to distant site_
↪(indicate distant vitam_site_name)
    - name: offer-s3-3
      referent: false
      # status: INACTIVE (valeur par défaut: ACTIVE)
      # id: idoffre # OPTIONAL, if used, MUST BE UNIQUE & same on each site
      # asyncRead: true # ONLY ENABLE WHEN tape-library
      # vitam_site_name: site3 # Optional, needed only when call to distant site_
↪(indicate distant vitam_site_name)
    binary:
      - name: offer-s3-3
        referent: false
        # status: INACTIVE (valeur par défaut: ACTIVE)
        # id: idoffre # OPTIONAL, if used, MUST BE UNIQUE & same on each site
        # asyncRead: true # ONLY ENABLE WHEN tape-library
        # vitam_site_name: site3 # Optional, needed only when call to distant site_
↪(indicate distant vitam_site_name)

```

4.2.2.1.7 plateforme_secret

Fichier: deployment/environments/group_vars/all/vault-vitam.yml

Cette variable stocke le *secret de plateforme* qui doit être commun à tous les composants de la solution logicielle *VITAM* de tous les sites. La valeur doit donc être identique pour chaque site.

4.2.2.1.8 consul_encrypt

Fichier: deployment/environments/group_vars/all/vault-vitam.yml

Cette variable stocke le *secret de plateforme* qui doit être commun à tous les *Consul* de tous les sites. La valeur doit donc être identique pour chaque site.

4.2.2.2 Procédure de réinstallation

En prérequis, il est nécessaire d'attendre que tous les *workflows* et reconstructions (sites secondaires) en cours soient terminés.

Ensuite :

- Arrêter vitam sur le site primaire.
- Arrêter les sites secondaires.
- Redéployer vitam sur les sites secondaires.
- Redéployer vitam sur le site primaire

4.2.2.3 Flux entre Storage et Offer

Dans le cas d'appel en **https** entre les composants **Storage** et **Offer**, il faut modifier `deployment/environments/group_vars/all/vitam_vars.yml` et indiquer `https_enabled: true` dans `storageofferdefault`.

Il convient également d'ajouter :

- **Sur le site primaire**
 - Dans le truststore de Storage : la **CA** ayant signé le certificat de l'Offer du site secondaire
- **Sur le site secondaire**
 - Dans le truststore de Offer : la **CA** ayant signé le certificat du Storage du site primaire
 - Dans le grantedstore de Offer : le certificat du storage du site primaire

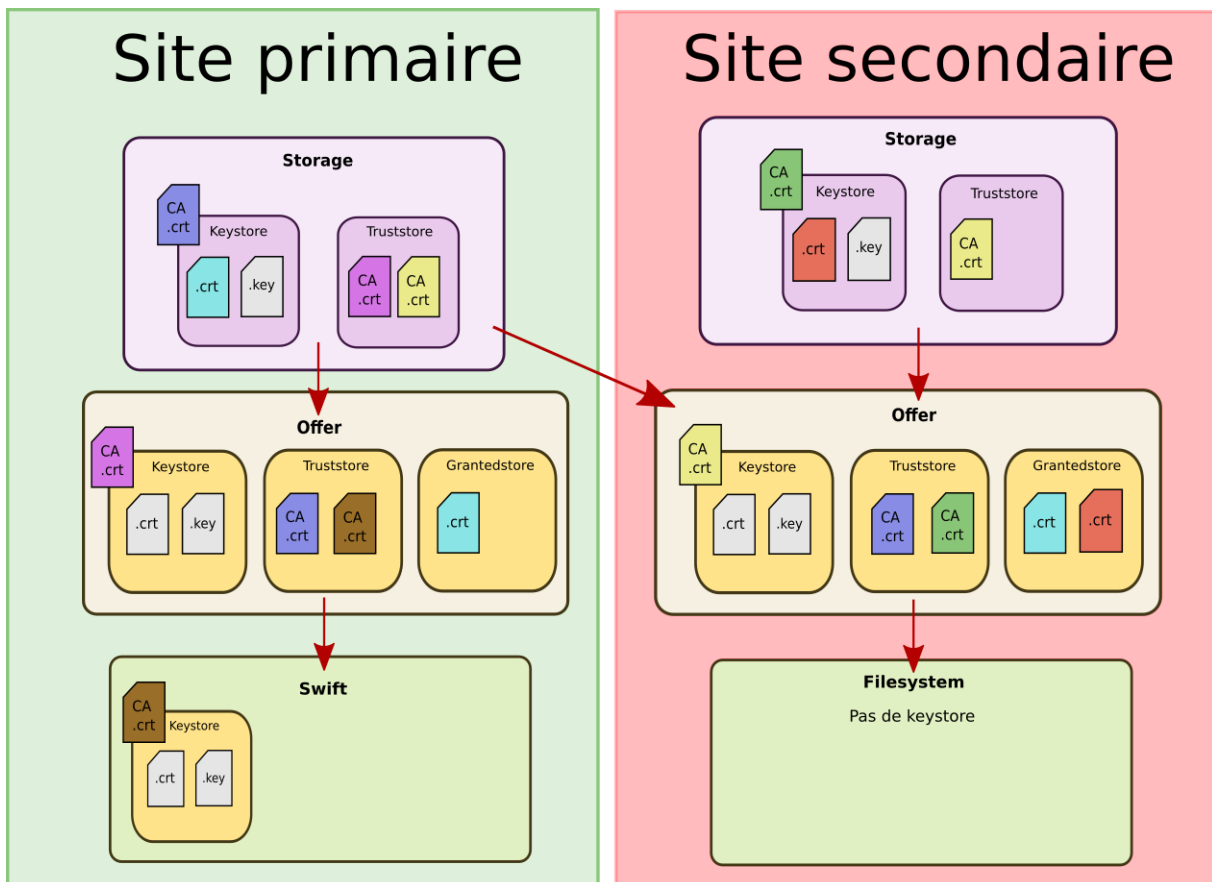


Fig. 2 – Vue détaillée des certificats entre le storage et l'offre en multi-site

Il est possible de procéder de 2 manières différentes :

4.2.2.3.1 Avant la génération des keystores

Avertissement : Pour toutes les copies de certificats indiquées ci-dessous, il est important de ne jamais les écraser, il faut donc renommer les fichiers si nécessaire.

Déposer les [CA](#) du client storage du site 1 `environments/certs/client-storage/ca/*` dans le client storage du site 2 `environments/certs/client-storage/ca/`.

Déposer le certificat du client storage du site 1 `environments/certs/client-storage/clients/storage/*` dans le client storage du site 2 `environments/certs/client-storage/clients/storage/`.

Déposer les [CA](#) du serveur offer du site 2 `environments/certs/server/ca/*` dans le répertoire des [CA](#) serveur du site 1 `environments/certs/server/ca/*`

4.2.2.3.2 Après la génération des keystores

Via le script `deployment/generate_stores.sh`, il convient donc d'ajouter les [CA](#) et certificats indiqués sur le schéma ci-dessus.

Ajout d'un certificat : `keytool -import -keystore -file <certificat.crt> -alias <alias_certificat>`

Ajout d'une [CA](#) : `keytool -import -trustcacerts -keystore -file <ca.crt> -alias <alias_certificat>`

4.2.3 Configuration du déploiement

Voir aussi :

L'architecture de la solution logicielle, les éléments de dimensionnement ainsi que les principes de déploiement sont définis dans le [DAT](#).

4.2.3.1 Fichiers de déploiement

Les fichiers de déploiement sont disponibles dans la version [VITAM](#) livrée, dans le sous-répertoire `deployment`. Concernant l'installation, ils se déclinent en 2 parties :

- les *playbook* ansible de déploiement, présents dans le sous-répertoire `ansible-vitam`, qui est indépendant de l'environnement à déployer ; ces fichiers ne sont normalement pas à modifier pour réaliser une installation.
- l'arborescence d'inventaire ; des fichiers d'exemples sont disponibles dans le sous-répertoire `environments`. Cette arborescence est valable pour le déploiement d'un environnement, et doit être dupliquée lors de l'installation d'environnements ultérieurs. Les fichiers contenus dans cette arborescence doivent être adaptés avant le déploiement, comme expliqué dans les paragraphes suivants.

4.2.3.2 Informations *plate-forme*

4.2.3.2.1 Inventaire

Pour configurer le déploiement, il est nécessaire de créer, dans le répertoire `environments`, un nouveau fichier d'inventaire (par la suite, ce fichier sera communément appelé `hosts.<environnement>`). Ce fichier devra se conformer à la structure présente dans le fichier `hosts.example` (et notamment respecter scrupuleusement l'arborescence des groupes *ansible*). Les commentaires dans ce fichier fournissent les explications permettant l'adaptation à l'environnement cible :

```

1  # Group definition ; DO NOT MODIFY
2  [hosts]
3
4  # Group definition ; DO NOT MODIFY
5  [hosts:children]
6  vitam
7  reverse
8  hosts_dev_tools
9  ldap
10
11
12  ##### Tests environments specifics #####
13
14  # EXTRA : Front reverse-proxy (test environments ONLY) ; add machine name after
15  [reverse]
16  # optional : after machine, if this machine is different from VITAM machines, you can
17  ↪ specify another become user
18  # Example
19  # vitam-centos-01.vitam ansible_ssh_user=centos
20
21  ##### Extra VITAM applications #####
22
23  [ldap] # Extra : OpenLDAP server
24  # LDAP server !!! NOT FOR PRODUCTION !!! Test only
25
26  [library]
27  # TODO: Put here servers where this service will be deployed : library
28
29  [hosts_dev_tools]
30  # TODO: Put here servers where this service will be deployed : mongo-express,
31  ↪ elasticsearch-head
32
33  [elasticsearch:children] # EXTRA : elasticsearch
34  hosts_elasticsearch_data
35  hosts_elasticsearch_log
36
37  ##### VITAM services #####
38
39  # Group definition ; DO NOT MODIFY
40  [vitam:children]
41  zone_external
42  zone_access
43  zone_applicative
44  zone_storage
45  zone_data
46  zone_admin
47  library

```

(suite sur la page suivante)

(suite de la page précédente)

```

46
47
48 ##### Zone externe
49
50
51 [zone_external:children]
52 hosts_ihm_demo
53 hosts_ihm_recette
54
55
56 [hosts_ihm_demo]
57 # TODO: Put here servers where this service will be deployed : ihm-demo. If you own_
58 ↪ another frontend, it is recommended to leave this group blank
59 # If you don't need consul for ihm-demo, you can set this var after each hostname :
60 # consul_disabled=true
61
62 [hosts_ihm_recette]
63 # TODO: Put here servers where this service will be deployed : ihm-recette (extra_
64 ↪ feature)
65
66 ##### Zone access
67
68 # Group definition ; DO NOT MODIFY
69 [zone_access:children]
70 hosts_ingest_external
71 hosts_access_external
72
73 [hosts_ingest_external]
74 # TODO: Put here servers where this service will be deployed : ingest-external
75
76 [hosts_access_external]
77 # TODO: Put here servers where this service will be deployed : access-external
78
79 ##### Zone applicative
80
81 # Group definition ; DO NOT MODIFY
82 [zone_applicative:children]
83 hosts_ingest_internal
84 hosts_processing
85 hosts_batch_report
86 hosts_worker
87 hosts_access_internal
88 hosts_metadata
89 hosts_functional_administration
90 hosts_logbook
91 hosts_workspace
92 hosts_storage_engine
93 hosts_security_internal
94
95
96 [hosts_security_internal]
97 # TODO: Put here servers where this service will be deployed : security-internal
98
99
100 [hosts_logbook]

```

(suite sur la page suivante)

(suite de la page précédente)

```

101 # TODO: Put here servers where this service will be deployed : logbook
102
103
104 [hosts_workspace]
105 # TODO: Put the server where this service will be deployed : workspace
106 # WARNING: put only one server for this service, not more !
107
108
109 [hosts_ingest_internal]
110 # TODO: Put here servers where this service will be deployed : ingest-internal
111
112
113 [hosts_access_internal]
114 # TODO: Put here servers where this service will be deployed : access-internal
115
116
117 [hosts_metadata]
118 # TODO: Put here servers where this service will be deployed : metadata
119
120
121 [hosts_functional_administration]
122 # TODO: Put here servers where this service will be deployed : functional-
    ↪ administration
123
124
125 [hosts_processing]
126 # TODO: Put the server where this service will be deployed : processing
127 # WARNING: put only one server for this service, not more !
128
129
130 [hosts_storage_engine]
131 # TODO: Put here servers where this service will be deployed : storage-engine
132
133
134 [hosts_batch_report]
135 # TODO: Put here servers where this service will be deployed : batch-report
136
137
138 [hosts_worker]
139 # TODO: Put here servers where this service will be deployed : worker
140 # Optional parameter after each host : vitam_worker_capacity=<integer> ; please refer_
    ↪ to your infrastructure for defining this number ; default is ansible_processor_
    ↪ vcpus value (cpu number in /proc/cpuinfo file)
141
142
143 ##### Zone storage
144
145 [zone_storage:children] # DO NOT MODIFY
146 hosts_storage_offer_default
147 hosts_mongodb_offer
148
149 [hosts_storage_offer_default]
150 # TODO: Put here servers where this service will be deployed : storage-offer-default
151 # LIMIT : only 1 offer per machine
152 # LIMIT and 1 machine per offer when filesystem or filesystem-hash provider
153 # Possibility to declare multiple machines with same provider only when provider is_
    ↪ s3 or swift.
154 # Mandatory param for each offer is offer_conf and points to offer_opts.yml & vault-
    ↪ vitam.yml (with same tree)

```

(suite sur la page suivante)

(suite de la page précédente)

```

153 # for swift
154 # hostname-offre-1.vitam offer_conf=offer-swift-1
155 # hostname-offre-2.vitam offer_conf=offer-swift-1
156 # for filesystem
157 # hostname-offre-2.vitam offer_conf=offer-fs-1
158 # for s3
159 # hostname-offre-3.vitam offer_conf=offer-s3-1
160 # hostname-offre-4.vitam offer_conf=offer-s3-1
161
162 [hosts_mongodb_offer:children]
163 hosts_mongos_offer
164 hosts_mongoc_offer
165 hosts_mongod_offer
166
167 [hosts_mongos_offer]
168 # WARNING : DO NOT COLLOCATE WITH [hosts_mongos_data]
169 # TODO: put here servers where this service will be deployed : mongos cluster for
↳ storage offers
170 # Mandatory param : mongo_cluster_name : name of the cluster (should exist in the
↳ offer_conf configuration)
171 # The recommended practice is to install the mongos instance on the same servers as
↳ the mongoc instances
172 # Example (for a more complete one, see the one in the group hosts_mongos_data) :
173 # vitam-mongo-swift-offer-01    mongo_cluster_name=offer-swift-1
174 # vitam-mongo-swift-offer-02    mongo_cluster_name=offer-swift-1
175 # vitam-mongo-fs-offer-01       mongo_cluster_name=offer-fs-1
176 # vitam-mongo-fs-offer-02       mongo_cluster_name=offer-fs-1
177 # vitam-mongo-s3-offer-01       mongo_cluster_name=offer-s3-1
178 # vitam-mongo-s3-offer-02       mongo_cluster_name=offer-s3-1
179
180 [hosts_mongoc_offer]
181 # WARNING : DO NOT COLLOCATE WITH [hosts_mongoc_data]
182 # TODO: put here servers where this service will be deployed : mongoc cluster for
↳ storage offers
183 # Mandatory param : mongo_cluster_name : name of the cluster (should exist in the
↳ offer_conf configuration)
184 # Optional param : mandatory for 1 node of the shard, some init commands will be
↳ executed on it
185 # Optional param : mongo_arbiter=true : the node will be only an arbiter ; do not add
↳ this paramter on a mongo_rs_bootstrap node
186 # Recommended practice in production: use 3 instances
187 # Example :
188 # vitam-mongo-swift-offer-01    mongo_cluster_name=offer-swift-1
↳ mongo_rs_bootstrap=true
189 # vitam-mongo-swift-offer-02    mongo_cluster_name=offer-swift-1
190 # vitam-swift-offer             mongo_cluster_name=offer-swift-1
↳ mongo_arbiter=true
191 # vitam-mongo-fs-offer-01       mongo_cluster_name=offer-fs-1
↳ mongo_rs_bootstrap=true
192 # vitam-mongo-fs-offer-02       mongo_cluster_name=offer-fs-1
193 # vitam-fs-offer                mongo_cluster_name=offer-fs-1
↳ mongo_arbiter=true
194 # vitam-mongo-s3-offer-01       mongo_cluster_name=offer-s3-1
↳ mongo_rs_bootstrap=true
195 # vitam-mongo-s3-offer-02       mongo_cluster_name=offer-s3-1
196 # vitam-s3-offer                mongo_cluster_name=offer-s3-1
↳ mongo_arbiter=true

```

(suite sur la page suivante)

(suite de la page précédente)

```

197
198 [hosts_mongod_offer]
199 # WARNING : DO NOT COLLOCATE WITH [hosts_mongod_data]
200 # TODO: put here servers where this service will be deployed : mongod cluster for
    ↳ storage offers
201 # Mandatory param : mongo_cluster_name : name of the cluster (should exist in the
    ↳ offer_conf configuration)
202 # Mandatory param : id of the current shard, increment by 1 from 0 to n
203 # Optional param : mandatory for 1 node of the shard, some init commands will be
    ↳ executed on it
204 # Optional param : mongo_arbiter=true : the node will be only an arbiter ; do not add
    ↳ this paramter on a mongo_rs_bootstrap node
205 # Optional param : mongod_memory=x ; this will force the wiredtiger cache size to x
    ↳ (unit is GB) ; can be usefull when colocalization with elasticsearch
206 # Optional param : is_small=true ; this will force the priority for this server to be
    ↳ lower when electing master ; hardware can be downgraded for this machine
207 # Recommended practice in production: use 3 instances per shard
208 # Example :
209 # vitam-mongo-swift-offer-01    mongo_cluster_name=offer-swift-1    mongo_shard_id=0
    ↳                                     mongo_rs_bootstrap=true
210 # vitam-mongo-swift-offer-02    mongo_cluster_name=offer-swift-1    mongo_shard_id=0
211 # vitam-swift-offer             mongo_cluster_name=offer-swift-1    mongo_shard_id=0
    ↳                                     mongo_arbiter=true
212 # vitam-mongo-fs-offer-01       mongo_cluster_name=offer-fs-1       mongo_shard_id=0
    ↳                                     mongo_rs_bootstrap=true
213 # vitam-mongo-fs-offer-02       mongo_cluster_name=offer-fs-1       mongo_shard_id=0
214 # vitam-fs-offer                mongo_cluster_name=offer-fs-1       mongo_shard_id=0
    ↳                                     mongo_arbiter=true
215 # vitam-mongo-s3-offer-01       mongo_cluster_name=offer-s3-1       mongo_shard_id=0
    ↳                                     mongo_rs_bootstrap=true
216 # vitam-mongo-s3-offer-02       mongo_cluster_name=offer-s3-1       mongo_shard_id=0
    ↳                                     is_small=true # PSsmin, this machine needs less hardware
217 # vitam-s3-offer                mongo_cluster_name=offer-s3-1       mongo_shard_id=0
    ↳                                     mongo_arbiter=true
218
219 ##### Zone data
220
221 # Group definition ; DO NOT MODIFY
222 [zone_data:children]
223 hosts_elasticsearch_data
224 hosts_mongodb_data
225
226 [hosts_elasticsearch_data]
227 # TODO: Put here servers where this service will be deployed : elasticsearch-data
    ↳ cluster
228 # 2 params available for huge environments (parameter to be declared after each
    ↳ server) :
229 #     is_data=true/false
230 #     is_master=true/false
231 #     for site/room balancing : is_balancing=<whatever> so replica can be applied on
    ↳ all sites/rooms ; default is vitam_site_name
232 #     other options are not handled yet
233 # defaults are set to true, if undefined. If defined, at least one server MUST be is_
    ↳ data=true
234 # Examples :
235 # server1 is_master=true is_data=false
236 # server2 is_master=false is_data=true

```

(suite sur la page suivante)

(suite de la page précédente)

```

237 # More explanation here : https://www.elastic.co/guide/en/elasticsearch/reference/5.6/
    ↳modules-node.html
238
239
240 # Group definition ; DO NOT MODIFY
241 [hosts_mongodb_data:children]
242 hosts_mongos_data
243 hosts_mongoc_data
244 hosts_mongod_data
245
246 [hosts_mongos_data]
247 # WARNING : DO NOT COLLOCATE WITH [hosts_mongos_offer]
248 # TODO: Put here servers where this service will be deployed : mongos cluster
249 # Mandatory param : mongo_cluster_name=mongo-data ("mongo-data" is mandatory)
250 # The recommended practice is to install the mongos instance on the same servers as
    ↳the mongoc instances
251 # Example :
252 # vitam-mdbs-01      mongo_cluster_name=mongo-data
253 # vitam-mdbs-02      mongo_cluster_name=mongo-data
254 # vitam-mdbs-03      mongo_cluster_name=mongo-data
255
256 [hosts_mongoc_data]
257 # WARNING : DO NOT COLLOCATE WITH [hosts_mongoc_offer]
258 # TODO: Put here servers where this service will be deployed : mongoc cluster
259 # Mandatory param : mongo_cluster_name=mongo-data ("mongo-data" is mandatory)
260 # Optional param : mandatory for 1 node of the shard, some init commands will be
    ↳executed on it
261 # Recommended practice in production: use 3 instances
262 # Example :
263 # vitam-mdbc-01      mongo_cluster_name=mongo-data                mongo_rs_
    ↳bootstrap=true
264 # vitam-mdbc-02      mongo_cluster_name=mongo-data
265 # vitam-mdbc-03      mongo_cluster_name=mongo-data
266
267 [hosts_mongod_data]
268 # WARNING : DO NOT COLLOCATE WITH [hosts_mongod_offer]
269 # TODO: Put here servers where this service will be deployed : mongod cluster
270 # Each replica_set should have an odd number of members (2n + 1)
271 # Reminder: For Vitam, one mongodb shard is using one replica_set
272 # Mandatory param : mongo_cluster_name=mongo-data ("mongo-data" is mandatory)
273 # Mandatory param : id of the current shard, increment by 1 from 0 to n
274 # Optional param : mandatory for 1 node of the shard, some init commands will be
    ↳executed on it
275 # Optional param : mongod_memory=x ; this will force the wiredtiger cache size to x
    ↳(unit is GB) ; can be usefull when colocalization with elasticsearch
276 # Recommended practice in production: use 3 instances per shard
277 # Example:
278 # vitam-mdbd-01      mongo_cluster_name=mongo-data      mongo_shard_id=0  mongo_rs_
    ↳bootstrap=true
279 # vitam-mdbd-02      mongo_cluster_name=mongo-data      mongo_shard_id=0
280 # vitam-mdbd-03      mongo_cluster_name=mongo-data      mongo_shard_id=0
281 # vitam-mdbd-04      mongo_cluster_name=mongo-data      mongo_shard_id=1  mongo_rs_
    ↳bootstrap=true
282 # vitam-mdbd-05      mongo_cluster_name=mongo-data      mongo_shard_id=1
283 # vitam-mdbd-06      mongo_cluster_name=mongo-data      mongo_shard_id=1
284
285 ##### Zone admin

```

(suite sur la page suivante)

(suite de la page précédente)

```

286
287 # Group definition ; DO NOT MODIFY
288 [zone_admin:children]
289 hosts_cerebro
290 hosts_consul_server
291 hosts_kibana_data
292 log_servers
293 hosts_elasticsearch_log
294
295 [hosts_cerebro]
296 # TODO: Put here servers where this service will be deployed : vitam-elasticsearch-
    ↳ cerebro
297
298 [hosts_consul_server]
299 # TODO: Put here servers where this service will be deployed : consul
300
301 [hosts_kibana_data]
302 # TODO: Put here servers where this service will be deployed : kibana (for data_
    ↳ cluster)
303
304 [log_servers:children]
305 hosts_kibana_log
306 hosts_logstash
307
308
309 [hosts_kibana_log]
310 # TODO: Put here servers where this service will be deployed : kibana (for log_
    ↳ cluster)
311
312 [hosts_logstash]
313 # TODO: Put here servers where this service will be deployed : logstash
314 # IF you connect VITAM to external SIEM, DO NOT FILL THE SECTION
315
316
317 [hosts_elasticsearch_log]
318 # TODO: Put here servers where this service will be deployed : elasticsearch-log_
    ↳ cluster
319 # IF you connect VITAM to external SIEM, DO NOT FILL THE SECTION
320
321 ##### Global vars #####
322
323 [hosts:vars]
324
325 # =====
326 # VITAM
327 # =====
328
329 # Declare user for ansible on target machines
330 ansible_ssh_user=
331 # Can target user become as root ? ; true is required by VITAM (usage of a sudoer is_
    ↳ mandatory)
332 ansible_become=true
333 # How can ansible switch to root ?
334 # See https://docs.ansible.com/ansible/latest/user_guide/become.html
335
336 # Related to Consul ; apply in a table your DNS server(s)
337 # Example : dns_servers=["8.8.8.8","8.8.4.4"]

```

(suite sur la page suivante)

(suite de la page précédente)

```

338 # If no recursors, use : dns_servers=
339 dns_servers=
340
341 # Vitam tenants to create
342 vitam_tenant_ids=[0,1,2]
343 vitam_tenant_admin=1
344
345 ### Logback configuration ###
346 # Days before deleting logback log files (java & access logs for vitam components)
347 days_to_delete_logback_logfiles=
348
349 # Define local Consul datacenter name
350 # CAUTION !!! Only alphanumeric characters when using s3 as offer backend !!!
351 vitam_site_name=prod-dcl
352 # On offer, value is the prefix for all containers' names. If upgrading from R8, you
353   ↳ MUST UNCOMMENT this parameter AS IS !!!
354 #vitam_prefix_offer=""
355 # EXAMPLE : vitam_site_name = prod-dcl
356 # check whether on primary site (true) or secondary (false)
357 primary_site=true
358
359 # =====
360 # EXTRA
361 # =====
362 # Environment (defines title in extra on reverse homepage). Variable is DEPRECATED !
363 #environnement=
364
365 ### vitam-itest repository ###
366 vitam_tests_branch=master
367 vitam_tests_gitrepo_protocol=
368 vitam_tests_gitrepo_baseurl=
369 vitam_tests_gitrepo_url=
370
371 # Used when VITAM is behind a reverse proxy (provides configuration for reverse proxy
372   ↳ && displayed in header page)
373 vitam_reverse_external_dns=
374 # For reverse proxy use
375 reverse_proxy_port=443
376 vitam_reverse_external_protocol=https
377 # http_proxy env var to use ; has to be declared even if empty
378 http_proxy_environnement=

```

Pour chaque type de *host*, indiquer le(s) serveur(s) défini(s), pour chaque fonction. Une colocalisation de composants est possible (Cf. le paragraphe idoine du *DAT*)

Note : Concernant le groupe *hosts_consul_server*, il est nécessaire de déclarer au minimum 3 machines.

Avertissement : Il n'est pas possible de colocaliser les clusters MongoDB *data* et *offer*.

Avertissement : Il n'est pas possible de colocaliser *kibana-data* et *kibana-log*.

Note : Pour les composants considérés par l'exploitant comme étant « hors *VITAM* » (typiquement, le composant ihm-demo), il est possible de désactiver la création du service Consul associé. Pour cela, après chaque hostname impliqué, il faut rajouter la directive suivante : `consul_disabled=true`.

Prudence : Concernant la valeur de `vitam_site_name`, seuls les caractères alphanumériques et le tiret (« - ») sont autorisés.

Note : Il est possible de multi-instancier le composant « storage-offer-default » dans le cas d'un *provider* de type objet (s3, swift). Il faut ajouter `offer_conf=<le nom>`.

4.2.3.2.2 Fichier `vitam_security.yml`

La configuration des droits d'accès à VITAM est réalisée dans le fichier `environments /group_vars/all/vitam_security.yml`, comme suit :

```
1 ---
2
3 hide_passwords_during_deploy: true
4
5 ### Admin context name and tenants ###
6 admin_context_name: "admin-context"
7 admin_context_tenants: "{{ vitam_tenant_ids }}"
8 # Indicate context certificates relative paths under {{ inventory_dir }}/certs/client-
9 ↪external/clients
10 # vitam-admin-int is mandatory for internal use (PRONOM upload)
11 admin_context_certs: [ "ihm-demo/ihm-demo.crt", "ihm-recette/ihm-recette.crt",
12 ↪"reverse/reverse.crt", "vitam-admin-int/vitam-admin-int.crt" ]
13 # Indicate here all the personal certificates relative paths under {{ inventory_dir }}
14 ↪/certs/client-vitam-users/clients
15 admin_personal_certs: [ "userOK.crt" ]
16
17 # Admin security profile name
18 admin_security_profile: "admin-security-profile"
19
20 admin_basic_auth_user: "adminUser"
21
22 # SELinux state, can be: enforcing, permissive, disabled
23 selinux_state: "disabled"
24 # SELinux Policy, can be: targeted, minimum, mls
25 selinux_policy: "targeted"
26 # If needed, reboot the VM to enable SELinux
27 selinux_reboot: True
28 # Relabel the entire filesystem ?
29 selinux_relabel: False
```

Note : Pour la directive `admin_context_certs` concernant l'intégration de certificats *SIA* au déploiement, se reporter à la section *Intégration d'une application externe (cliente)* (page 50).

Note : Pour la directive `admin_personal_certs` concernant l'intégration de certificats personnels (*personae*) au déploiement, se reporter à la section *Intégration d'un certificat personnel (personae)* (page 50).

4.2.3.2.3 Fichier `offers_opts.yml`

Indication : Fichier à créer depuis `offers_opts.yml.example` et à paramétrer selon le besoin.

La déclaration de configuration des offres de stockage associées se fait dans le fichier `environments / group_vars/all/offers_opts.yml` :

```

1  # This is the default vitam strategy ('default'). It is mandatory and must_
   ↳define a referent offer.
2  # This list of offers is ordered. It can and has to be completed if more_
   ↳offers are necessary
3  # Strategy order (1st has to be the preferred one)
4  vitam_strategy:
5    - name: offer-fs-1
6      referent: true
7    # status: ACTIVE # status : enable (value=ACTIVE, default value) or_
   ↳disable (value=INACTIVE) this offer
8    # vitam_site_name: prod-dc2 # optional, should be related to vitam_site_
   ↳name if local ; remote vitam_site_name if distant
9    # - name: offer-swift-1
10   # Example distant:
11   # - name: distant
12   # referent: true
13   # status: INACTIVE
14   # vitam_site_name: distant-dc2
15   # distant: true # Only add this parameter when distant offer (not on same_
   ↳platform)
16
17   # WARNING : multi-strategy is a BETA functionality
18   # More strategies can be added but are optional
19   # Strategy name must only use [a-z][a-z0-9-]* pattern
20   # Any strategy must contain at least one offer
21   # This list of offers is ordered. It can and has to be completed if more_
   ↳offers are necessary
22   # Every strategy can define at most one referent offer.
23   # other_strategies:
24   # metadata:
25   #   - name: offer-fs-1
26   #     referent: true
27   #   - name: offer-fs-2
28   #     referent: false
29   # binary:
30   #   - name: offer-fs-2
31   #     referent: false
32   #   - name: offer-s3-1
33   #     referent: false
34
35   # DON'T forget to add associated passwords in vault-vitam.yml with same tree_
   ↳when using provider openstack-swift*
```

(suite sur la page suivante)

(suite de la page précédente)

```

36 # ATTENTION !!! Each offer has to have a distinct name, except for clusters.
37 ↳ binding a same physical storage
38 # WARNING : for offer names, please only use [a-z][a-z0-9-]* pattern
39 vitam_offers:
40   offer-tape-1:
41     provider: tape-library
42     tapeLibraryConfiguration:
43       maxTarEntrySize: 100000
44       maxTarFileSize: 1000000
45       # Enable overriding non empty cartridges
46       # WARNING : FOR DEV/TEST ONLY. DO NOT ENABLE IN PRODUCTION.
47       forceOverrideNonEmptyCartridges: false
48       # Archive (Tar) file expire time for retention in local FS
49       archiveRetentionCacheTimeoutInMinutes: 30
50
51     useSudo: false
52     topology:
53       buckets:
54         -
55           name: test
56           tenants: [0]
57           tarBufferingTimeoutInMinutes: 60
58         -
59           name: admin
60           tenants: [1]
61           tarBufferingTimeoutInMinutes: 60
62         -
63           name: prod
64           tenants: [2,3,4,5,6,7,8,9]
65           tarBufferingTimeoutInMinutes: 60
66     tapeLibraries:
67       -
68         name: TAPE_LIB_1
69         robots:
70           -
71             device: /dev/tape/by-id/scsi-1QUANTUM_10F73224E6664C84A1D00000
72             mtxPath: "/usr/sbin/mtx"
73             timeoutInMilliseconds: 3600000
74         drives:
75           -
76             index: 0
77             device: /dev/tape/by-id/scsi-1IBM_ULT3580-TD6_1235308739-nst
78             mtPath: "/bin/mt"
79             ddPath: "/bin/dd"
80             tarPath: "/bin/tar"
81             timeoutInMilliseconds: 3600000
82             readWritePriority: BACKUP
83           -
84             index: 1
85             device: /dev/tape/by-id/scsi-1IBM_ULT3580-TD6_0951859786-nst
86             mtPath: "/bin/mt"
87             ddPath: "/bin/dd"
88             tarPath: "/bin/tar"
89             timeoutInMilliseconds: 3600000
90             readWritePriority: READ
91           -
92             index: 2

```

(suite sur la page suivante)

(suite de la page précédente)

```

92     device: /dev/tape/by-id/scsi-1IBM_ULT3580-TD6_0269493808-nst
93     mtPath: "/bin/mt"
94     ddPath: "/bin/dd"
95     tarPath: "/bin/tar"
96     timeoutInMilliseconds: 3600000
97 -
98     index: 3
99     device: /dev/tape/by-id/scsi-1IBM_ULT3580-TD6_0566471858-nst
100    mtPath: "/bin/mt"
101    ddPath: "/bin/dd"
102    tarPath: "/bin/tar"
103    readWritePriority: READ
104    timeoutInMilliseconds: 3600000
105    offer_log_compaction:
106      ## Expiration, here offer logs 21 days old will be compacted
107      expiration_value: 21
108      ## Choose one of "MILLENNIA", "HALF_DAYS", "MILLIS", "FOREVER", "MICROS
109      ↪", "CENTURIES", "DECADES", "YEARS", "DAYS", "SECONDS", "HOURS", "MONTHS",
110      ↪"WEEKS", "NANOS", "MINUTES", "ERAS"
111      expiration_unit: "DAYS"
112      ## Compaction bulk size here 10 000 offers logs (at most) will be
113      ↪compacted (Expected value between 1 000 and 200 000)
114      compaction_size: 10000
115    offer-fs-1:
116      # param can be filesystem-hash (recomended) or filesystem (not
117      ↪recomended)
118      provider: filesystem-hash
119      # Offer log compaction
120      offer_log_compaction:
121        ## Expiration, here offer logs 21 days old will be compacted
122        expiration_value: 21
123        ## Choose one of "MILLENNIA", "HALF_DAYS", "MILLIS", "FOREVER", "MICROS
124        ↪", "CENTURIES", "DECADES", "YEARS", "DAYS", "SECONDS", "HOURS", "MONTHS",
125        ↪"WEEKS", "NANOS", "MINUTES", "ERAS"
126        expiration_unit: "DAYS"
127        ## Compaction bulk size here 10 000 offers logs (at most) will be
128        ↪compacted (Expected value between 1 000 and 200 000)
129        compaction_size: 10000
130      offer-swift-1:
131        # provider : openstack-swift for v1 or openstack-swift-v3 for v3
132        provider: openstack-swift-v3
133        # swiftKeystoneAuthUrl : URL de connexion à keystone
134        swiftKeystoneAuthUrl: https://openstack-hostname:port/auth/1.0
135        # swiftDomain : domaine OpenStack dans lequel l'utilisateur est
136        ↪enregistré
137        swiftDomain: domaine
138        # swiftUser : identifiant de l'utilisateur
139        swiftUser: utilisateur
140        # swiftPassword: has to be set in vault-vitam.yml (encrypted) with same
141        ↪structure => DO NOT COMMENT OUT
142        # swiftProjectName : nom du projet openstack
143        swiftProjectName: monTenant
144        # swiftUrl: optional variable to force the swift URL
145        # swiftUrl: https://swift-hostname:port/swift/v1
146        #SSL TrustStore
147        swiftTrustStore: /chemin_vers_mon_fichier/monSwiftTrustStore.jks
148        #Max connection (concurrent connections), per route, to keep in pool (if
149        ↪a pooling ConnectionManager is used) (by default 2 for Apache HttpClient)

```

(suite de la page précédente)

```

140 swiftMaxConnectionsPerRoute: 200
141     #Max total connection (concurrent connections) to keep in pool (if a
↳pooling ConnectionManager is used) (by default 20 for Apache HttpClient)
142 swiftMaxConnections: 1000
143     #Max time (in milliseconds) for waiting to establish connection
144 swiftConnectionTimeout: 200000
145     #Max time (in milliseconds) waiting for a data from the server (socket)
146 swiftReadTimeout: 60000
147     #Time (in seconds) to renew a token before expiration occurs (blocking)
148 swiftHardRenewTokenDelayBeforeExpireTime: 60
149 offer_log_compaction:
150     ## Expiration, here offer logs 21 days old will be compacted
151     expiration_value: 21
152     ## Choose one of "MILLENNIA", "HALF_DAYS", "MILLIS", "FOREVER", "MICROS
↳", "CENTURIES", "DECADES", "YEARS", "DAYS", "SECONDS", "HOURS", "MONTHS",
↳"WEEKS", "NANOS", "MINUTES", "ERAS"
153     expiration_unit: "DAYS"
154     ## Compaction bulk size here 10 000 offers logs (at most) will be
↳compacted (Expected value between 1 000 and 200 000)
155     compaction_size: 10000
156 offer-s3-1:
157     # provider : can only be amazon-s3-v1 for Amazon SDK S3 V1
158     provider: 'amazon-s3-v1'
159     # s3Endpoint : : URL of connection to S3
160     s3Endpoint: https://s3.domain/
161     # s3RegionName (optional): Region name (default value us-east-1)
162     s3RegionName: us-east-1
163     # s3SignerType (optional): Signing algorithm.
164     #     - signature V4 : 'AWSS3V4SignerType' (default value)
165     #     - signature V2 : 'S3SignerType'
166     s3SignerType: AWSS3V4SignerType
167     # s3PathStyleAccessEnabled (optional): 'true' to access bucket in "path-
↳style", else "virtual-hosted-style" (false by default in java client, true
↳by default in ansible scripts)
168     s3PathStyleAccessEnabled: true
169     # s3MaxConnections (optional): Max total connection (concurrent
↳connections) (50 by default)
170     s3MaxConnections: 50
171     # s3ConnectionTimeout (optional): Max time (in milliseconds) for waiting
↳to establish connection (10000 by default)
172     s3ConnectionTimeout: 10000
173     # s3SocketTimeout (optional): Max time (in milliseconds) for reading
↳from a connected socket (50000 by default)
174     s3SocketTimeout: 50000
175     # s3RequestTimeout (optional): Max time (in milliseconds) for a request
↳(0 by default, disabled)
176     s3RequestTimeout: 0
177     # s3ClientExecutionTimeout (optional): Max time (in milliseconds) for a
↳request by java client (0 by default, disabled)
178     s3ClientExecutionTimeout: 0
179
180     #Time (in seconds) to renew a token before expiration occurs
181     swiftSoftRenewTokenDelayBeforeExpireTime: 300
182 offer_log_compaction:
183     ## Expiration, here offer logs 21 days old will be compacted
184     expiration_value: 21
185     ## Choose one of "MILLENNIA", "HALF_DAYS", "MILLIS", "FOREVER", "MICROS
↳", "CENTURIES", "DECADES", "YEARS", "DAYS", "SECONDS", "HOURS", "MONTHS",
↳"WEEKS", "NANOS", "MINUTES", "ERAS"

```

(suite sur la page suivante)

(suite de la page précédente)

```

186     expiration_unit: "DAYS"
187     ## Compaction bulk size here 10 000 offers logs (at most) will be
↳ compacted (Expected value between 1 000 and 200 000)
188     compaction_size: 10000
189
190     # example_swift_v1:
191     #     provider: openstack-swift
192     #     swiftKeystoneAuthUrl: https://keystone/auth/1.0
193     #     swiftDomain: domain
194     #     swiftUser: user
195     #     swiftPassword: has to be set in vault-vitam.yml (encrypted) with same
↳ structure => DO NOT COMMENT OUT
196     # example_swift_v3:
197     #     provider: openstack-swift-v3
198     #     swiftKeystoneAuthUrl: https://keystone/v3
199     #     swiftDomain: domaine
200     #     swiftUser: user
201     #     swiftPassword: has to be set in vault-vitam.yml (encrypted) with same
↳ structure => DO NOT COMMENT OUT
202     #     swiftProjectName: monTenant
203     #     projectName: monTenant
204     #     swiftTrustStore: /chemin_vers_mon_fichier/monSwiftTrustStore.jks
205     #     swiftMaxConnectionsPerRoute: 200
206     #     swiftMaxConnections: 1000
207     #     swiftConnectionTimeout: 200000
208     #     swiftReadTimeout: 60000
209     #     Time (in seconds) to renew a token before expiration occurs
210     #     swiftHardRenewTokenDelayBeforeExpireTime: 60
211     #     swiftSoftRenewTokenDelayBeforeExpireTime: 300

```

Se référer aux commentaires dans le fichier pour le renseigner correctement.

Note : Dans le cas d'un déploiement multi-sites, dans la section `vitam_strategy`, la directive `vitam_site_name` définit pour l'offre associée le nom du datacenter Consul. Par défaut, si non définie, c'est la valeur de la variable `vitam_site_name` définie dans l'inventaire qui est prise en compte.

Avertissement : La cohérence entre l'inventaire et la section `vitam_strategy` (et `other_strategies` si multi-stratégies) est critique pour le bon déploiement et fonctionnement de la solution logicielle VITAM. En particulier, la liste d'offres de `vitam_strategy` doit correspondre *exactement* aux noms d'offres déclarés dans l'inventaire (ou les inventaires de chaque datacenter, en cas de fonctionnement multi-site).

Avertissement : Ne pas oublier, en cas de connexion à un keystone en https, de répercuter dans la *PKI* la clé publique de la *CA* du keystone.

4.2.3.2.4 Fichier `cots_vars.yml`

Fichier le fichier `environments/group_vars/all/cots_vars.yml` :


```

1  ---
2
3  consul:
4      retry_interval: 10 # in seconds
5      check_interval: 10 # in seconds
6      check_timeout: 5 # in seconds
7      network: "ip_admin" # Which network to use for consul communications ?
8      ↪ ip_admin or ip_service ?
9
10     consul_remote_sites:
11         # wan contains the wan addresses of the consul server instances of the
12         ↪ external vitam sites
13         # Example, if our local dc is dc1, we will need to set dc2 & dc3 wan
14         ↪ conf:
15             # - dc2:
16             #     wan: ["10.10.10.10", "1.1.1.1"]
17             # - dc3:
18             #     wan: ["10.10.10.11", "1.1.1.1"]
19         # Please uncomment and fill values if you want to connect VITAM to external
20         ↪ SIEM
21         # external_siem:
22         #     host:
23         #     port:
24
25     elasticsearch:
26         log:
27             host: "elasticsearch-log.service.{{ consul_domain }}"
28             port_http: "9201"
29             groupe: "log"
30             baseuri: "elasticsearch-log"
31             cluster_name: "elasticsearch-log"
32             consul_check_http: 10 # in seconds
33             consul_check_tcp: 10 # in seconds
34             action_log_level: error
35             https_enabled: false
36             indices fielddata_cache_size: 0.3 # related to https://www.elastic.
37             ↪ co/guide/en/elasticsearch/reference/6.8/modules-fielddata.html
38             indices breaker fielddata_limit: 0.4 # related to https://www.
39             ↪ elastic.co/guide/en/elasticsearch/reference/6.8/circuit-breaker.html
40             ↪ #fielddata-circuit-breaker
41             dynamic_timeout: 30s
42             # default index template
43             index_templates:
44                 default:
45                     shards: 1
46                     replica: 1
47             packetbeat:
48                 shards: 5
49             log_appenders:
50                 root:
51                     log_level: "info"
52                 rolling:
53                     max_log_file_size: "100MB"
54                     max_total_log_size: "5GB"
55                     max_files: "50"
56                 deprecation_rolling:
57                     max_log_file_size: "100MB"

```

(suite sur la page suivante)

(suite de la page précédente)

```

51         max_total_log_size: "1GB"
52         max_files: "10"
53         log_level: "warn"
54     index_search_slowlog_rolling:
55         max_log_file_size: "100MB"
56         max_total_log_size: "1GB"
57         max_files: "10"
58         log_level: "warn"
59     index_indexing_slowlog_rolling:
60         max_log_file_size: "100MB"
61         max_total_log_size: "1GB"
62         max_files: "10"
63         log_level: "warn"
64     # By default, is commented. Should be uncommented if ansible
65     ↪ computes badly vCPUs number ; values are associated vCPUs numbers ;
66     ↪ please adapt to your configuration
67     # thread_pool:
68     #     index:
69     #         size: 2
70     #     get:
71     #         size: 2
72     #     search:
73     #         size: 2
74     #     write:
75     #         size: 2
76     #     warmer:
77     #         max: 2
78     data:
79         host: "elasticsearch-data.service.{{ consul_domain }}"
80         # default is 0.1 (10%) and should be quite enough in most cases
81         #index_buffer_size_ratio: "0.15"
82         port_http: "9200"
83         groupe: "data"
84         baseuri: "elasticsearch-data"
85         cluster_name: "elasticsearch-data"
86         consul_check_http: 10 # in seconds
87         consul_check_tcp: 10 # in seconds
88         action_log_level: debug
89         https_enabled: false
90         # discovery_zen_minimum_master_nodes: 2 # comented by default ; by
91         ↪ default, value is half the length of ansible associated group whose racks
92         ↪ have the same number of machine. If it is not the case, this value have to
93         ↪ be set with the smallest rack (if using param is_balancing). ONLY EXISTS
94         ↪ FOR DATA CLUSTER !!!! DO NOT FORGET TO APPLY PARAMETER WITH REPLICA NUMBER
95         ↪ !!!!
96         indices_fielddata_cache_size: 0.3 # related to https://www.elastic.
97         ↪ co/guide/en/elasticsearch/reference/6.5/modules-fielddata.html
98         indices_breaker_fielddata_limit: 0.4 # related to https://www.
99         ↪ elastic.co/guide/en/elasticsearch/reference/6.5/circuit-breaker.html
100        ↪ #fielddata-circuit-breaker
101        dynamic_timeout: 30s
102        # default index template
103        index_templates:
104            default:
105                shards: 10
106                replica: 2
107        log_appenders:

```

(suite sur la page suivante)

(suite de la page précédente)

```

98     root:
99         log_level: "info"
100     rolling:
101         max_log_file_size: "100MB"
102         max_total_log_size: "5GB"
103         max_files: "50"
104     deprecation_rolling:
105         max_log_file_size: "100MB"
106         max_total_log_size: "5GB"
107         max_files: "50"
108         log_level: "warn"
109     index_search_slowlog_rolling:
110         max_log_file_size: "100MB"
111         max_total_log_size: "5GB"
112         max_files: "50"
113         log_level: "warn"
114     index_indexing_slowlog_rolling:
115         max_log_file_size: "100MB"
116         max_total_log_size: "5GB"
117         max_files: "50"
118         log_level: "warn"
119     # By default, is commented. Should be uncommented if ansible_
120     ↪computes badly vCPUs number ; values are associated vCPUs numbers ;
121     ↪please adapt to your configuration
122     # thread_pool:
123     #     index:
124     #         size: 2
125     #     get:
126     #         size: 2
127     #     search:
128     #         size: 2
129     #     write:
130     #         size: 2
131     #     warmer:
132     #         max: 2
133
134 mongodb:
135     mongos_port: 27017
136     mongoc_port: 27018
137     mongod_port: 27019
138     mongo_authentication: "true"
139     host: "mongos.service.{{ consul_domain }}"
140     check_consul: 10 # in seconds
141     drop_info_log: false # Drop mongo (I)nformational log, for Verbosity_
142     ↪Level of 0
143
144 logstash:
145     host: "logstash.service.{{ consul_domain }}"
146     user: logstash
147     port: 10514
148     rest_port: 20514
149     check_consul: 10 # in seconds
150     # logstash xms & xmx in Megabytes
151     # jvm_xms: 2048
152     # jvm_xmx: 2048
153     # workers_number: 4
154     log_appenders:

```

(suite sur la page suivante)

(suite de la page précédente)

```

152     rolling:
153         max_log_file_size: "100MB"
154         max_total_log_size: "5GB"
155     json_rolling:
156         max_log_file_size: "100MB"
157         max_total_log_size: "5GB"
158
159     # Curator units: days
160     curator:
161         log:
162             metrics:
163                 close: 5
164                 delete: 30
165             logstash:
166                 close: 5
167                 delete: 30
168             metricbeat:
169                 close: 5
170                 delete: 30
171             packetbeat:
172                 close: 5
173                 delete: 30
174
175     kibana:
176         header_value: "reporting"
177         import_delay: 10
178         import_retries: 10
179         log:
180             baseuri: "kibana_log"
181             api_call_timeout: 120
182             groupe: "log"
183             port: 5601
184             default_index_pattern: "logstash-vitam*"
185             check_consul: 10 # in seconds
186             # default shards & replica
187             shards: 5
188             replica: 1
189             # pour index logstash-*
190             metrics:
191                 shards: 5
192                 replica: 1
193             # pour index metrics-vitam-*
194             logs:
195                 shards: 5
196                 replica: 1
197             # pour index metricbeat-*
198             metricbeat:
199                 shards: 5 # must be a factor of 30
200                 replica: 1
201         data:
202             baseuri: "kibana_data"
203             # OMA : bugdette : api_call_timeout is used for retries ; should_
204             ↳ create a separate variable rather than this one
205             api_call_timeout: 120
206             groupe: "data"
207             port: 5601
208             default_index_pattern: "logbookoperation_*"

```

(suite sur la page suivante)

(suite de la page précédente)

```

208     check_consul: 10 # in seconds
209     # index template for .kibana
210     shards: 1
211     replica: 1
212
213 syslog:
214     # value can be syslog-ng or rsyslog
215     name: "rsyslog"
216
217 cerebro:
218     baseuri: "cerebro"
219     port: 9000
220     check_consul: 10 # in seconds
221
222 siegfried:
223     port: 19000
224     consul_check: 10 # in seconds
225
226 clamav:
227     port: 3310
228     # frequency freshclam for database update per day (from 0 to 24 - 24_
↪ meaning hourly check)
229     db_update_periodicity: 1
230
231 mongo_express:
232     baseuri: "mongo-express"
233
234 ldap_authentication:
235     ldap_protocol: "ldap"
236     ldap_server: "{% if groups['ldap']|length > 0 %}{{ groups['ldap']|first }
↪ }{% endif %}"
237     ldap_port: "389"
238     ldap_base: "dc=programmevitam,dc=fr"
239     ldap_login: "cn=Manager,dc=programmevitam,dc=fr"
240     uid_field: "uid"
241     ldap_userDn_Template: "uid={0},ou=people,dc=programmevitam,dc=fr"
242     ldap_group_request: "(&(objectClass=groupOfNames)(member={0}))"
243     ldap_admin_group: "cn=admin,ou=groups,dc=programmevitam,dc=fr"
244     ldap_user_group: "cn=user,ou=groups,dc=programmevitam,dc=fr"
245     ldap_guest_group: "cn=guest,ou=groups,dc=programmevitam,dc=fr"
246
247 java_prerequisites:
248     debian: "openjdk-11-jre-headless"
249     redhat: "java-11-openjdk-headless"

```

Dans le cas du choix du *COTS* d'envoi des messages syslog dans logstash, il est possible de choisir entre syslog-ng et rsyslog. Il faut alors modifier la valeur de la directive `syslog.name`; la valeur par défaut est `rsyslog`.

Note : si vous décommentez et renseignez les valeurs dans le bloc `external_siem`, les messages seront envoyés (par syslog ou syslog-ng, selon votre choix de déploiement) dans un *SIEM* externe à la solution logicielle *VITAM*, aux valeurs indiquées dans le bloc ; il n'est alors pas nécessaire de renseigner de partitions pour les groupes `ansible [hosts_logstash]` et `[hosts_elasticsearch_log]`.

4.2.3.3 Déclaration des secrets

Avertissement : L'ensemble des mots de passe fournis ci-après le sont par défaut et doivent être changés !

4.2.3.3.1 vitam

Avertissement : Cette section décrit des fichiers contenant des données sensibles. Il est important d'implémenter une politique de mot de passe robuste conforme à ce que l'ANSSI préconise. Par exemple : ne pas utiliser le même mot de passe pour chaque service, renouveler régulièrement son mot de passe, utiliser des majuscules, minuscules, chiffres et caractères spéciaux (Se référer à la documentation ANSSI <https://www.ssi.gouv.fr/guide/mot-de-passe>). En cas d'usage d'un fichier de mot de passe (*vault-password-file*), il faut renseigner ce mot de passe comme contenu du fichier et ne pas oublier de sécuriser ou supprimer ce fichier à l'issue de l'installation.

Les secrets utilisés par la solution logicielle (en-dehors des certificats qui sont abordés dans une section ultérieure) sont définis dans des fichiers chiffrés par `ansible-vault`.

Important : Tous les vault présents dans l'arborescence d'inventaire doivent être tous protégés par le même mot de passe !

La première étape consiste à changer les mots de passe de tous les vaults présents dans l'arborescence de déploiement (le mot de passe par défaut est contenu dans le fichier `vault_pass.txt`) à l'aide de la commande `ansible-vault rekey <fichier vault>`.

Voici la liste des vaults pour lesquels il est nécessaire de modifier le mot de passe :

- `environments/group_vars/all/vault-vitam.yml`
- `environments/group_vars/all/vault-keystores.yml`
- `environments/group_vars/all/vault-extra.yml`
- `environments/certs/vault-certs.yml`

2 vaults sont principalement utilisés dans le déploiement d'une version :

Avertissement : Leur contenu est donc à modifier avant tout déploiement.

- Le fichier `environments /group_vars/all/vault-vitam.yml` contient les secrets généraux :

```

1  ---
2  # Vitam platform secret key
3  plateforme_secret: vitamsecret
4
5  # The consul key must be 16-bytes, Base64 encoded: https://www.consul.io/docs/
6  ↪agent/encryption.html
7  # You can generate it with the "consul keygen" command
8  # Or you can use this script: deployment/pki/scripts/generate_consul_key.sh
9  consul_encrypt: Biz14ohqN4HtvZmrXp3N4A==
10
11 mongodb:
12   mongo-data:
13     passphrase: changeitkM4L6zBgK527tWBb

```

(suite sur la page suivante)

(suite de la page précédente)

```
13 admin:
14     user: vitamdb-admin
15     password: change_it_1MpG22m2MywvKW5E
16 localadmin:
17     user: vitamdb-localadmin
18     password: change_it_HycFEVD74g397iRe
19 system:
20     user: vitamdb-system
21     password: change_it_HycFEVD74g397iRe
22 metadata:
23     user: metadata
24     password: change_it_37b97KVADV8YbCwt
25 logbook:
26     user: logbook
27     password: change_it_jVi6q8eX4H1Ce8UC
28 report:
29     user: report
30     password: change_it_jb7TASZbU6n85t8L
31 functionalAdmin:
32     user: functional-admin
33     password: change_it_9eA2zMCL6tm6KF1e
34 securityInternal:
35     user: security-internal
36     password: change_it_m39XvRQWixyDX566
37 offer-fs-1:
38     passphrase: changeitmB5rnk1M5TY61PqZ
39     admin:
40         user: vitamdb-admin
41         password: change_it_FLkM5emt63N73EcN
42     localadmin:
43         user: vitamdb-localadmin
44         password: change_it_QeH8q4e16ah4QKXS
45     system:
46         user: vitamdb-system
47         password: change_it_HycFEVD74g397iRe
48     offer:
49         user: offer
50         password: change_it_pQi1T1yT9LAF8au8
51 offer-fs-2:
52     passphrase: changeiteSY1By57qZr4MX2s
53     admin:
54         user: vitamdb-admin
55         password: change_it_84aTMFZ7h8e2NgMe
56     localadmin:
57         user: vitamdb-localadmin
58         password: change_it_Am1B37tGY1w5VfvX
59     system:
60         user: vitamdb-system
61         password: change_it_HycFEVD74g397iRe
62     offer:
63         user: offer
64         password: change_it_mLDYds957sNQ53mA
65 offer-tape-1:
66     passphrase: changeitmB5rnk1M5TY61PqZ
67     admin:
68         user: vitamdb-admin
69         password: change_it_FLkM5emt63N73EcN
```

(suite sur la page suivante)

(suite de la page précédente)

```

70  localadmin:
71      user: vitamdb-localadmin
72      password: change_it_QeH8q4e16ah4QKXS
73  system:
74      user: vitamdb-system
75      password: change_it_HycFEVD74g397iRe
76  offer:
77      user: offer
78      password: change_it_pQilT1yT9LAF8au8
79  offer-swift-1:
80      passphrase: changeitgYvt42M2pKL6Zx3T
81      admin:
82          user: vitamdb-admin
83          password: change_it_e21hLp51WNa4sJFS
84  localadmin:
85      user: vitamdb-localadmin
86      password: change_it_QB8857SJrGrQh2yu
87  system:
88      user: vitamdb-system
89      password: change_it_HycFEVD74g397iRe
90  offer:
91      user: offer
92      password: change_it_AWJg2Bp3s69P6nMe
93  offer-s3-1:
94      passphrase: changeituF1jVdR9NqdTG625
95      admin:
96          user: vitamdb-admin
97          password: change_it_5b7cSWcS5M1NF4kv
98  localadmin:
99      user: vitamdb-localadmin
100     password: change_it_S9jE24rxHwUZP6y5
101  system:
102     user: vitamdb-system
103     password: change_it_HycFEVD74g397iRe
104  offer:
105     user: offer
106     password: change_it_TuTB1i2k7iQW3zL2
107  offer-tape-1:
108     passphrase: changeituF1jghT9NqdTG625
109     admin:
110         user: vitamdb-admin
111         password: change_it_5b7cSWcab91NF4kv
112  localadmin:
113     user: vitamdb-localadmin
114     password: change_it_S9jE24rxHwUZP5a6
115  system:
116     user: vitamdb-system
117     password: change_it_HycFEVD74g397iRe
118  offer:
119     user: offer
120     password: change_it_TuTB1i2k7iQW3c2a
121
122  vitam_users:
123      - vitam_aadmin:
124          login: aadmin
125          password: change_it_z5MP7GC4qnR8nL9t
126          role: admin

```

(suite sur la page suivante)

(suite de la page précédente)

```

127 - vitam_uuser:
128   login: uuser
129   password: change_it_w94Q3jPAT2aJYm8b
130   role: user
131 - vitam_gguest:
132   login: gguest
133   password: change_it_E5v7Tr4h6tYaQG2W
134   role: guest
135 - techadmin:
136   login: techadmin
137   password: change_it_K29EluHcPZ8zXji8
138   role: admin
139
140 ldap_authentication:
141   ldap_pwd: "change_it_t69Rn5NdUv39EYkC"
142
143 admin_basic_auth_password: change_it_5Yn74JgXwbQ9KdP8
144
145 vitam_offers:
146   offer-swift-1:
147     swiftPassword: change_it_m44j57aYeRpnPXQ2
148   offer-s3-1:
149     s3AccessKey: accessKey_change_grLS8372Uga5EJSx
150     s3SecretKey: secretKey_change_p97es2m2CHXPJA1m

```

Prudence : Seuls les caractères alphanumériques sont valides pour les directives passphrase.

Avertissement : Le paramétrage du mode d'authentications des utilisateurs à l'*IHM* démo est géré au niveau du fichier `deployment/environments/group_vars/all/vitam_vars.yml`. Plusieurs modes d'authentications sont proposés au niveau de la section `authentication_realms`. Dans le cas d'une authentification se basant sur le mécanisme `iniRealm` (configuration `shiro` par défaut), les mots de passe déclarés dans la section `vitam_users` devront s'appuyer sur une politique de mot de passe robuste, comme indiqué en début de chapitre. Il est par ailleurs possible de choisir un mode d'authentification s'appuyant sur un annuaire LDAP externe (`ldapRealm` dans la section `authentication_realms`).

Note : Dans le cadre d'une installation avec au moins une offre *swift*, il faut déclarer, dans la section `vitam_offers`, le nom de chaque offre et le mot de passe de connexion *swift* associé, défini dans le fichier `offers_opts.yml`. L'exemple ci-dessus présente la déclaration du mot de passe pour l'offre *swift offer-swift-1*.

Note : Dans le cadre d'une installation avec au moins une offre *s3*, il faut déclarer, dans la section `vitam_offers`, le nom de chaque offre et l'accès key secret *s3* associé, défini dans le fichier `offers_opts.yml`. L'exemple ci-dessus présente la déclaration du mot de passe pour l'offre *s3 offer-s3-1*.

- Le fichier `environments/group_vars/all/vault-keystores.yml` contient les mots de passe des magasins de certificats utilisés dans VITAM :

```

1 # NO UNDERSCORE ALLOWED IN VALUES
2 keystores:
3   server:
4     offer: changeit817NR75vWsZtgAgJ
5     access_external: changeitMZFD2YM4279miitu
6     ingest_external: changeita2C74cQhy84BLWCr
7     ihm_recette: changeit4FWYVK1347mxjGfe
8     ihm_demo: changeit6kQl6eyDY7QPS9fy
9   client_external:
10    ihm_demo: changeitGT38hhTiA32x1PLy
11    gatling: changeit2sBC5ac7NfGF9Qj7
12    ihm_recette: changeitdAZ9Eq65UhDZd9p4
13    reverse: changeite5XTzb5yVPcEX464
14    vitam_admin_int: changeitz6xZe5gDu7nhDZd9
15  client_storage:
16    storage: changeit647D7LWiyM6qYMnm
17  timestamping:
18    secure_logbook: changeitMn9Skuyx87VYU62U
19    secure_storage: changeite5gDu9Skuy84BLW9
20 truststores:
21   server: changeitxNe4JLfn528PVHj7
22   client_external: changeitJ2eS93DcPH1v4jAp
23   client_storage: changeitHpSCa31aG8ttB87S
24 grantedstores:
25   client_external: changeitLL22HkmDCA2e2vj7
26   client_storage: changeitR3wvp5C8KQS76Vcu

```

Avvertissement : il convient de sécuriser votre environnement en définissant des mots de passe *forts*.

4.2.3.3.2 Cas des extras

- Le fichier `environments/group_vars/all/vault-extra.yml` contient les mots de passe des magasins de certificats utilisés dans VITAM :

```

1 # Example for git lfs ; uncomment & use if needed
2 #vitam_gitlab_itest_login: "account"
3 #vitam_gitlab_itest_password: "change_it_4DU42JVf2x2xmPBs"

```

Note : le playbook `vitam.yml` comprend des étapes avec la mention `no_log` afin de ne pas afficher en clair des étapes comme les mots de passe des certificats. En cas d'erreur, il est possible de retirer la ligne dans le fichier pour une analyse plus fine d'un éventuel problème sur une de ces étapes.

4.2.3.3.3 Commande `ansible-vault`

Certains fichiers présents sous `environments/group_vars/all` commençant par **`vault-`** doivent être protégés (encryptés) avec l'utilitaire `ansible-vault`.

Note : Ne pas oublier de mettre en conformité le fichier `vault_pass.txt`

4.2.3.3.1 Générer des fichiers *vaultés* depuis des fichier en clair

Exemple du fichier `vault-cots.example`

```
cp vault-cots.example vault-cots.yml
ansible-vault encrypt vault-cots.yml
```

4.2.3.3.2 Ré-encoder un fichier *vaulté*

Exemple du fichier `vault-cots.yml`

```
ansible-vault rekey vault-cots.yml
```

4.2.4 Gestion des certificats

Une vue d'ensemble de la gestion des certificats est présentée *dans l'annexe dédiée* (page 97).

4.2.4.1 Cas 1 : Configuration développement / tests

Pour des usages de développement ou de tests hors production, il est possible d'utiliser la *PKI* fournie avec la solution logicielle *VITAM*.

4.2.4.1.1 Procédure générale

Danger : La *PKI* fournie avec la solution logicielle *VITAM* doit être utilisée UNIQUEMENT pour faire des tests, et ne doit par conséquent surtout pas être utilisée en environnement de production ! De plus il n'est pas possible de l'utiliser pour générer les certificats d'une autre application qui serait cliente de *VITAM*.

La *PKI* de la solution logicielle *VITAM* est une suite de scripts qui vont générer dans l'ordre ci-dessous :

- Les autorités de certification (*CA*)
- Les certificats (clients, serveurs, de *timestamping*) à partir des *CA*
- Les *keystores*, en important les certificats et *CA* nécessaires pour chacun des *keystores*

4.2.4.1.2 Génération des *CA* par les scripts Vitam

Il faut faire la génération des autorités de certification (*CA*) par le script décrit ci-dessous.

Dans le répertoire de déploiement, lancer le script :

```
pki/scripts/generate_ca.sh
```

Ce script génère sous `pki/ca` les autorités de certification *root* et intermédiaires pour générer des certificats clients, serveurs, et de *timestamping*. Les mots de passe des clés privées des autorités de certification sont stockés dans le vault `ansible environments/certs/vault-ca.yml`

Avertissement : Il est impératif de noter les dates de création et de fin de validité des CA. En cas d'utilisation de la PKI fournie, la CA root a une durée de validité de 10 ans ; la CA intermédiaire a une durée de 3 ans.

4.2.4.1.3 Génération des certificats par les scripts Vitam

Le fichier d'inventaire de déploiement `environments/<fichier d'inventaire>` (cf. *Informations plate-forme* (page 22)) doit être correctement renseigné pour indiquer les serveurs associés à chaque service. En prérequis les CA doivent être présentes.

Puis, dans le répertoire de déploiement, lancer le script :

```
pki/scripts/generate_certs.sh <fichier d'inventaire>
```

Ce script génère sous `environments/certs` les certificats (format `crt` & `key`) nécessaires pour un bon fonctionnement dans VITAM. Les mots de passe des clés privées des certificats sont stockés dans le vault ansible `environments/certs/vault-certs.yml`.

Prudence : Les certificats générés à l'issue ont une durée de validité de 3 ans.

4.2.4.2 Cas 2 : Configuration production

4.2.4.2.1 Procédure générale

La procédure suivante s'applique lorsqu'une *PKI* est déjà disponible pour fournir les certificats nécessaires.

Les étapes d'intégration des certificats à la solution *Vitam* sont les suivantes :

- Générer les certificats avec les bons *key usage* par type de certificat
- Déposer les certificats et les autorités de certifications correspondantes dans les bons répertoires.
- Renseigner les mots de passe des clés privées des certificats dans le vault ansible `environments/certs/vault-certs.yml`
- Utiliser le script VITAM permettant de générer les différents *keystores*.

Note : Rappel pré-requis : vous devez disposer d'une ou plusieurs *PKI* pour tout déploiement en production de la solution logicielle *VITAM*.

4.2.4.2.2 Génération des certificats

En conformité avec le document RGSV2 de l'ANSSI, il est recommandé de générer des certificats avec les caractéristiques suivantes :

4.2.4.2.2.1 Certificats serveurs

- **Key Usage**
 - `digitalSignature`, `keyEncipherment`
- **Extended Key Usage**

- TLS Web Server Authentication

Les certificats serveurs générés doivent prendre en compte des alias « web » (`subjectAltName`).

Le *subjectAltName* des certificats serveurs (`deployment/environments/certs/server/hosts/*`) doit contenir le nom DNS du service sur consul associé.

Exemple avec un cas standard : `<composant_vitam>.service.<consul_domain>`. Ce qui donne pour le certificat serveur de `access-external` par exemple :

```
X509v3 Subject Alternative Name:
    DNS:access-external.service.consul, DNS:localhost
```

Il faudra alors mettre le même nom de domaine pour la configuration de Consul (fichier `deployment/environments/group_vars/all/vitam_vars.yml`, variable `consul_domain`)

Cas particulier pour `ihm-demo` et `ihm-recette` : il faut ajouter le nom *DNS* qui sera utilisé pour requêter ces deux applications, si celles-ci sont appelées directement en frontal `https`.

4.2.4.2.2 Certificat clients

- **Key Usage**
 - `digitalSignature`
- **Extended Key Usage**
 - TLS Web Client Authentication

4.2.4.2.3 Certificats d'horodatage

Ces certificats sont à générer pour les composants `logbook` et `storage`.

- **Key Usage**
 - `digitalSignature, nonRepudiation`
- **Extended Key Usage**
 - Time Stamping

4.2.4.2.3 Intégration de certificats existants

Une fois les certificats et *CA* mis à disposition par votre *PKI*, il convient de les positionner sous `environments/certs/...` en respectant la structure indiquée ci-dessous.

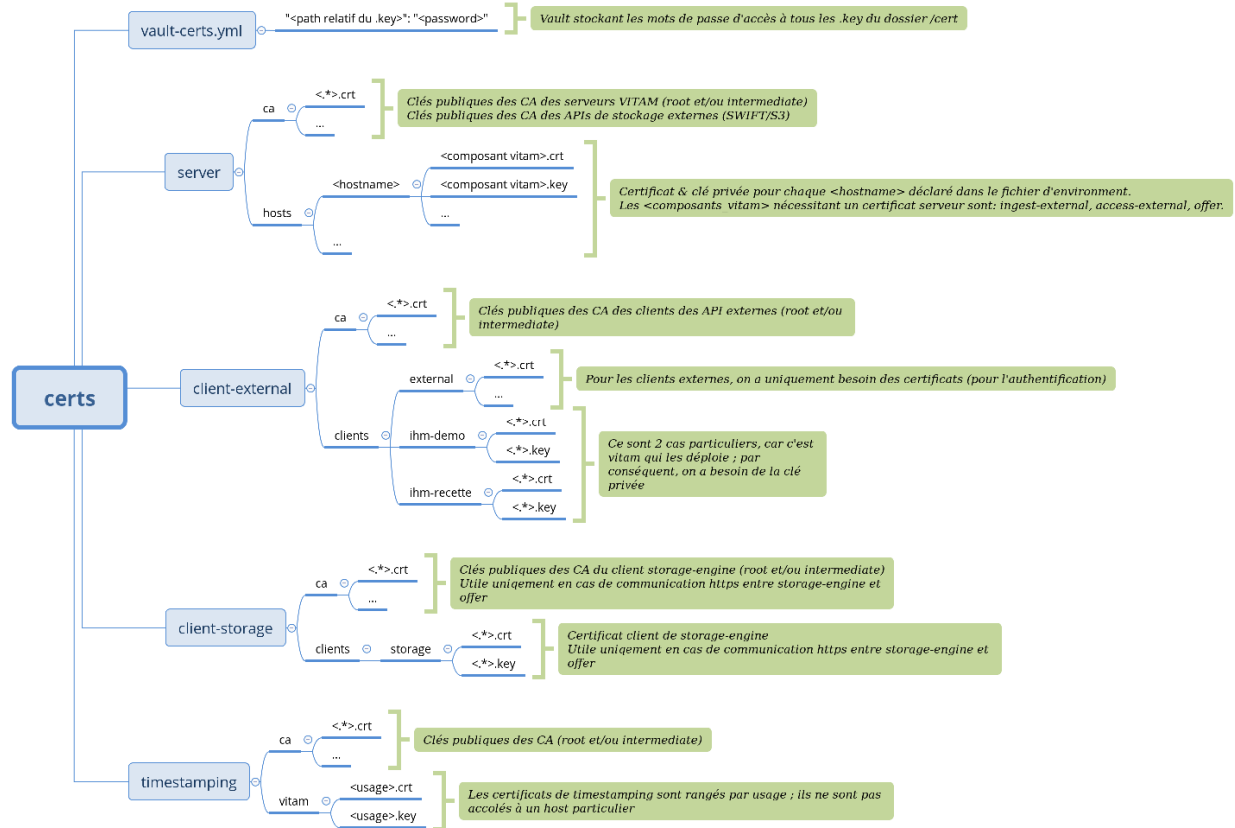


Fig. 3 – Vue détaillée de l'arborescence des certificats

Astuce : Dans le doute, n'hésitez pas à utiliser la *PKI* de test (étapes de génération de *CA* et de certificats) pour générer les fichiers requis au bon endroit et ainsi observer la structure exacte attendue ; il vous suffira ensuite de remplacer ces certificats « placeholders » par les certificats définitifs avant de lancer le déploiement.

Ne pas oublier de renseigner le vault contenant les *passphrases* des clés des certificats : `environments/certs/vault-certs.yml`

Pour modifier/créer un vault ansible, se référer à la documentation Ansible sur [cette url](http://docs.ansible.com/ansible/playbooks_vault.html) ¹³.

Prudence : Durant l'installation de VITAM, il est nécessaire de créer un certificat « vitam-admin-int » (à placer sous `deployment/environments/certs/client-external/clients/vitam-admin-int`).

Prudence : Durant l'installation des extra de VITAM, il est nécessaire de créer un certificat « gatling » (à placer sous `deployment/environments/certs/client-external/clients/gatling`).

http://docs.ansible.com/ansible/playbooks_vault.html

4.2.4.2.4 Intégration de certificats clients de VITAM

4.2.4.2.4.1 Intégration d'une application externe (cliente)

Dans le cas d'ajout de certificats *SIA* externes au déploiement de la solution logicielle *VITAM* :

- Déposer le certificat (.crt) de l'application client dans `environments/certs/client-external/clients/external/`
- Déposer les *CA* du certificat de l'application (.crt) dans `environments/certs/client-external/ca/`
- Editer le fichier `environments/group_vars/all/vitam_security.yml` et ajouter le(s) entrée(s) supplémentaire(s) (sous forme répertoire/fichier.crt, exemple : `external/mon_sia.crt`) dans la directive `admin_context_certs` pour que celles-ci soient associés aux contextes de sécurité durant le déploiement de la solution logicielle *VITAM*.

Note : Les certificats *SIA* externes ajoutés par le mécanisme de déploiement sont, par défaut, rattachés au contexte applicatif d'administration `admin_context_name` lui même associé au profil de sécurité `admin_security_profile` et à la liste de tenants `vitam_tenant_ids` (voir le fichier `environments/group_vars/all/vitam_security.yml`). Pour l'ajout de certificats applicatifs associés à des contextes applicatifs autres, se référer à la procédure du document d'exploitation (*DEX*) décrivant l'intégration d'une application externe dans Vitam.

4.2.4.2.4.2 Intégration d'un certificat personnel (*personae*)

Dans le cas d'ajout de certificats personnels au déploiement de la solution logicielle *VITAM* :

- Déposer le certificat personnel (.crt) dans `environments/certs/client-external/clients/external/`
- Editer le fichier `environments/group_vars/all/vitam_security.yml` et ajouter le(s) entrée(s) supplémentaire(s) (sous forme répertoire/fichier.crt, exemple : `external/mon_personae.crt`) dans la directive `admin_personal_certs` pour que ceux-ci soient ajoutés à la base de données du composant *security-internal* durant le déploiement de la solution logicielle *VITAM*.

4.2.4.2.5 Cas des offres objet

Placer le .crt de la *CA* dans `deployment/environments/certs/server/ca`.

4.2.4.2.6 Absence d'usage d'un *reverse*

Dans ce cas, il convient de :

- supprimer le répertoire `deployment/environments/certs/client-external/clients/reverse`
- supprimer les entrées **reverse** dans le fichier `vault_keystore.yml`

4.2.4.3 Intégration de CA pour une offre *Swift* ou *s3*

En cas d'utilisation d'une offre *Swift* ou *s3* en https, il est nécessaire d'ajouter les *CA* du certificat de l'*API Swift* ou *s3*.

Il faut les déposer dans `environments/certs/server/ca/` avant de jouer le script `./generate_keystores.sh`

4.2.4.4 Génération des magasins de certificats

En prérequis, les certificats et les autorités de certification (*CA*) doivent être présents dans les répertoires attendus.

Prudence : Avant de lancer le script de génération des *stores*, il est nécessaire de modifier le vault contenant les mots de passe des *stores* : `environments/group_vars/all/vault-keystores.yml`, décrit dans la section *Déclaration des secrets* (page 41).

Lancer le script : `./generate_stores.sh`

Ce script génère sous `environments/keystores` les *stores* (aux formats `jks` / `p12`) associés pour un bon fonctionnement dans la solution logicielle *VITAM*.

Il est aussi possible de déposer directement les *keystores* au bon format en remplaçant ceux fournis par défaut et en indiquant les mots de passe d'accès dans le vault : `environments/group_vars/all/vault-keystores.yml`

Note : Le mot de passe du fichier `vault-keystores.yml` est identique à celui des autres *vaults* ansible.

4.2.5 Paramétrages supplémentaires

4.2.5.1 *Tuning JVM*

Prudence : En cas de colocalisation, bien prendre en compte la taille *JVM* de chaque composant (*VITAM* : `-Xmx512m` par défaut) pour éviter de *swapper*.

Un *tuning* fin des paramètres *JVM* de chaque composant *VITAM* est possible. Pour cela, il faut modifier le contenu du fichier `environments/group_vars/all/jvm_opts.yml`

Pour chaque composant, il est possible de modifier ces 3 variables :

- `memory` : paramètres `Xms` et `Xmx`
- `gc` : paramètres `gc`
- `java` : autres paramètres `java`

4.2.5.2 Installation des *griffins* (greffons de préservation)

Note : Fonctionnalité disponible partir de la R9 (2.1.1) .

Prudence : Cette version de *VITAM* ne mettant pas encore en oeuvre de mesure d'isolation particulière des *griffins*, il est recommandé de veiller à ce que l'usage de chaque *griffin* soit en conformité avec la politique de sécurité de l'entité. Il est en particulier déconseillé d'utiliser un griffin qui utiliserait un outil externe qui n'est plus maintenu.

Il est possible de choisir les *griffins* installables sur la plate-forme. Pour cela, il faut éditer le contenu du fichier `environments/group_vars/all/vitam-vars.yml` au niveau de la directive `vitam_griffins`. Cette action est à rapprocher de l'incorporation des binaires d'installation : les binaires d'installation des greffons doivent être accessibles par les machines hébergeant le composant **worker**.

Exemple :

```
vitam_griffins: ["vitam-imagemagick-griffin", "vitam-jhove-griffin"]
```

Voici la liste des greffons disponibles au moment de la présente publication :

```
vitam-imagemagick-griffin
vitam-jhove-griffin
vitam-libreoffice-griffin
vitam-odfvalidator-griffin
vitam-siegfried-griffin
vitam-tesseract-griffin
vitam-verapdf-griffin
```

Avertissement : Ne pas oublier d'avoir déclaré au préalable sur les machines cibles le dépôt de binaires associé aux *griffins*.

4.2.5.3 Rétention liée aux logback

La solution logicielle *VITAM* utilise logback pour la rotation des log, ainsi que leur rétention.

Il est possible d'appliquer un paramétrage spécifique pour chaque composant VITAM.

Editer le fichier `environments/group_vars/all/vitam_vars.yml` (et `extra_vars.yml`, dans le cas des extra) et appliquer le paramétrage dans le bloc `logback_total_size_cap` de chaque composant sur lequel appliquer la modification de paramétrage. Pour chaque **APPENDER**, la valeur associée doit être exprimée en taille et unité (exemple : 14GB ; représente 14 gigabytes).

Note : des *appenders* supplémentaires existent pour le composant storage-engine (`appender offersync`) et `offer` (`offer_tape` et `offer_tape_backup`).

4.2.5.3.1 Cas des access_log

Il est également possible d'appliquer un paramétrage différent par composant VITAM sur le logback *access*.

Editer le fichier `environments/group_vars/all/vitam_vars.yml` (et `extra_vars.yml`, dans le cas des extra) et appliquer le paramétrage dans les directives `access_retention_days` et `access_total_size_GB` de chaque composant sur lequel appliquer la modification de paramétrage.

4.2.5.4 Paramétrage de l'antivirus (ingest-externe)

L'antivirus utilisé par ingest-externe est modifiable (par défaut, ClamAV) ; pour cela :

- Modifier le fichier `environments/group_vars/all/vitam_vars.yml` pour indiquer le nom de l'antivirus qui sera utilisé (norme : `scan-<nom indiqué dans vitam-vars.yml>.sh`)
- Créer un shell (dont l'extension doit être `.sh`) sous `environments/antivirus/` (norme : `scan-<nom indiqué dans vitam-vars.yml>.sh`) ; prendre comme modèle le fichier `scan-clamav.sh`. Ce script shell doit respecter le contrat suivant :
 - Argument : chemin absolu du fichier à analyser
 - **Sémantique des codes de retour**
 - 0 : Analyse OK - pas de virus
 - 1 : Analyse OK - virus trouvé et corrigé
 - 2 : Analyse OK - virus trouvé mais non corrigé
 - 3 : Analyse NOK
 - **Contenu à écrire dans stdout / stderr**
 - stdout : Nom des virus trouvés, un par ligne ; Si échec (code 3) : raison de l'échec
 - stderr : Log « brut » de l'antivirus

Prudence : En cas de remplacement de clamAV par un autre antivirus, l'installation de celui-ci devient dès lors un prérequis de l'installation et le script doit être testé.

Avertissement : Sur plate-forme Debian, ClamAV est installé sans base de données. Pour que l'antivirus soit fonctionnel, il est nécessaire, durant l'installation, de le télécharger ; il est donc nécessaire de renseigner dans l'inventaire la directive `http_proxy_environnement`.

4.2.5.5 Paramétrage des certificats externes (*-externe)

Se reporter au chapitre dédié à la gestion des certificats : *Gestion des certificats* (page 46)

4.2.5.6 Placer « hors Vitam » le composant ihm-demo

Sous `deployment/environments/host_vars`, créer ou éditer un fichier nommé par le nom de machine qui héberge le composant ihm-demo et ajouter le contenu ci-dessous

```
consul_disabled: true
```

Il faut également modifier le fichier `deployment/environments/group_vars/all/vitam_vars.yml` en remplaçant :

- dans le bloc `accessexternal`, la directive `host: "access-external.service.{{ consul_domain }}"` par `host: "<adresse IP de access-external>"` (l'adresse IP peut être une *FIP*)
- dans le bloc `ingestexternal`, la directive `host: "ingest-external.service.{{ consul_domain }}"` par `host: "<adresse IP de ingest-external>"` (l'adresse IP peut être une *FIP*)

À l'issue, le déploiement n'installera pas l'agent Consul. Le composant ihm-demo appellera, alors, par l'adresse *IP* de service les composants « access-external » et « ingest-external ».

Il est également fortement recommandé de positionner la valeur de la directive `vitam.ihm_demo.metrics_enabled` à `false` dans le fichier `deployment/environments/group_vars/all/vitam_vars.yml`, afin que ce composant ne tente pas d'envoyer des données sur « elasticsearch-log ».

4.2.5.7 Paramétrer le `secure_cookie` pour ihm-demo

Le composant ihm-demo (ainsi qu'ihm-recette) dispose d'une option supplémentaire, par rapport aux autres composants VITAM, dans le fichier `deployment/environments/group_vars/all/vitam_vars.yml` : le `secure_cookie` qui permet de renforcer ces deux *IHM* contre certaines attaques assez répandues comme les CSRF (Cross-Site Request Forgery).

Il faut savoir que si cette variable est à `true` (valeur par défaut), le client doit obligatoirement se connecter en https sur l'*IHM*, et ce même si un reverse proxy se trouve entre le serveur web et le client.

Cela peut donc obliger le reverse proxy frontal de la chaîne d'accès à écouter en https.

4.2.5.8 Paramétrage de la centralisation des logs VITAM

2 cas sont possibles :

- Utiliser le sous-système de gestion des logs fourni par la solution logicielle *VITAM* ;
- Utiliser un *SIEM* tiers.

4.2.5.8.1 Gestion par VITAM

Pour une gestion des logs par *VITAM*, il est nécessaire de déclarer les serveurs ad-hoc dans le fichier d'inventaire pour les 3 gro

- `hosts_logstash`
- `hosts_kibana_log`
- `hosts_elasticsearch_log`

4.2.5.8.2 Redirection des logs sur un SIEM tiers

En configuration par défaut, les logs VITAM sont tout d'abord routés vers un serveur rsyslog installé sur chaque machine. Il est possible d'en modifier le routage, qui par défaut redirige vers le serveur logstash, via le protocole syslog en TCP.

Pour cela, il est nécessaire de placer un fichier de configuration dédié dans le dossier `/etc/rsyslog.d/` ; ce fichier sera automatiquement pris en compte par rsyslog. Pour la syntaxe de ce fichier de configuration rsyslog, se référer à la [documentation rsyslog](#)¹⁴.

Astuce : Pour cela, il peut être utile de s'inspirer du fichier de référence *VITAM* `deployment/ansible-vitam/roles/rsyslog/templates/vitam_transport.conf.j2` (attention, il s'agit d'un fichier template ansible, non directement convertible en fichier de configuration sans en ôter les directives `jinja2`).

¹⁴<http://www.rsyslog.com/doc/v7-stable/>

4.2.5.9 Passage des identifiants des référentiels en mode *esclave*

La génération des identifiants des référentiels est géré par *VITAM* lorsqu'il fonctionne en mode maître.

Par exemple :

- Préfixé par PR- pour les profils
- Préfixé par IC- pour les contrats d'entrée
- Préfixé par AC- pour les contrats d'accès

Depuis la version 1.0.4, la configuration par défaut de *VITAM* autorise des identifiants externes (ceux qui sont dans le fichier json importé).

- pour le tenant 0 pour les référentiels : contrat d'entrée et contrat d'accès.
- pour le tenant 1 pour les référentiels : contrat d'entrée, contrat d'accès, profil, profil de sécurité et contexte.

La liste des choix possibles, pour chaque tenant, est :

Tableau 1: Description des identifiants de référentiels

Nom du référentiel	Description
INGEST_CONTRACT	contrats d'entrée
ACCESS_CONTRACT	contrats d'accès
PROFILE	profils <i>SEDA</i>
SECURITY_PROFILE	profils de sécurité (utile seulement sur le tenant d'administration)
CONTEXT	contextes applicatifs (utile seulement sur le tenant d'administration)
ARCHIVEUNITPROFILE	profils d'unités archivistiques

Si vous souhaitez gérer vous-même les identifiants sur un service référentiel, il faut qu'il soit en mode esclave.

Par défaut tous les services référentiels de Vitam fonctionnent en mode maître. Pour désactiver le mode maître de *VITAM*, il faut modifier le fichier ansible `deployment/environments/group_vars/all/vitam_vars.yml` dans les sections `vitam_tenants_usage_external` (pour gérer, par tenant, les collections en mode esclave).

4.2.5.10 Paramétrage du batch de calcul pour l'indexation des règles héritées

La paramétrage du batch de calcul pour l'indexation des règles héritées peut être réalisé dans le fichier `/group_vars/all/vitam_vars.yml`.

La section suivante du fichier `vitam_vars.yml` permet de paramétrer la fréquence de passage du batch :

```
vitam_timers:
  metadata:
    - name: vitam-metadata-computed-inherited-rules
      frequency: "*-*-* 02:30:00"
```

La section suivante du fichier `vitam_vars.yml` permet de paramétrer la liste des tenants sur lesquels s'exécute le batch :

```
vitam:
  worker:
    # api_output_index_tenants : permet d'indexer les règles de gestion, les_
    ↪ chemins des règles et les services producteurs
    api_output_index_tenants: [0,1,2,3,4,5,6,7,8,9]
    # rules_index_tenants : permet d'indexer les règles de gestion
    rules_index_tenants: [0,1,2,3,4,5,6,7,8,9]
```

4.2.5.11 Durées minimales permettant de contrôler les valeurs saisies

Afin de se prémunir contre une alimentation du référentiel des règles de gestion avec des durées trop courtes susceptibles de déclencher des actions indésirables sur la plate-forme (ex. éliminations) – que cette tentative soit intentionnelle ou non –, la solution logicielle **VITAM** vérifie que l'association de la durée et de l'unité de mesure saisies pour chaque champ est supérieure ou égale à une durée minimale définie lors du paramétrage de la plate-forme, dans un fichier de configuration.

Pour mettre en place le comportement attendu par le métier, il faut modifier le contenu de la directive `vitam_tenant_rule_duration` dans le fichier ansible `deployment/environments/group_vars/all/vitam_vars.yml`.

Exemple :

```
vitam_tenant_rule_duration:
  - name: 2 # applied tenant
    rules:
      - AppraisalRule : "1 year" # rule name : rule value
  - name: 3
    rules:
      AppraisalRule : "5 year" # rule name : rule value
      StorageRule : "5 year" # rule name : rule value
      ReuseRule : "2 year" # rule name : rule value
```

Par *tenant*, les directives possibles sont :

Tableau 2: Description des règles

Règle	Valeur par défaut
AppraisalRule	
DisseminationRule	
StorageRule	
ReuseRule	
AccessRule	0 year
ClassificationRule	

Les valeurs associées sont une durée au format <nombre> <unité en anglais, au singulier>

Exemples :

```
6 month
1 year
5 year
```

Voir aussi :

Pour plus de détails, se rapporter à la documentation métier « Règles de gestion ».

4.2.5.12 Fichiers complémentaires

A titre informatif, le positionnement des variables ainsi que des dérivations des déclarations de variables sont effectuées dans les fichiers suivants :

- `environments/group_vars/all/vitam_vars.yml`, comme suit :

```
1 ---
2 ### global ###
```

(suite sur la page suivante)

(suite de la page précédente)

```

3
4 # Disable epel or Debian backports repositories install
5 disable_internet_repositories_install: false
6
7 # TODO MAYBE : permettre la surcharge avec une syntax du genre vitamopts.folder_
  ↳ root | default(vitam_default.folder_root) dans les templates ?
8 droid_filename: "DROID_SignatureFile_V95.xml"
9 droid_container_filename: "container-signature-20180920.xml"
10
11 vitam_defaults:
12   folder:
13     root_path: /vitam
14     folder_permission: "0750"
15     conf_permission: "0640"
16     folder_upload_permission: "0770"
17     script_permission: "0750"
18   users:
19     vitam: "vitam"
20     vitamdb: "vitamdb"
21     group: "vitam"
22   services:
23     # Default log level for vitam components: logback values (TRACE, DEBUG,
  ↳ INFO, WARN, ERROR, OFF)
24     log_level: WARN
25     start_timeout: 300
26     stop_timeout: 3600
27     port_service_timeout: 86400
28     api_call_timeout: 120
29     api_long_call_timeout: 300
30     status_retries_number: 60
31     status_retries_delay: 5
32     # Filter for the vitam package version to install
33     # FIXME : commented as has to be removed because doesn't work under Debain
34     #package_version: "*"
35     ### Trust X-SSL-CLIENT-CERT header for external api auth ? (true | false) ###
36     vitam_ssl_user_header: true
37     ### Force chunk mode : set true if chunk header should be checked
38     vitam_force_chunk_mode: false
39     # syslog_facility
40     syslog_facility: local0
41     # Configuration of log for reconstruction services (INFO or DEBUG for active
  ↳ logs). Logs will be present only on secondary site.
42     reconstruction:
43       log_level: INFO
44
45     # Used in ingest, unitary update, mass-update
46     classificationList: ["Non protégé", "Secret Défense", "Confidentiel Défense"]
47     # Used in ingest, unitary update, mass-update
48     classificationLevelOptional: true
49     # Packages install retries
50     packages_install_retries_number: 1
51     packages_install_retries_delay: 10
52
53
54 vitam_timers:
55   # systemd nomenclature
56   # minutely → *- *-* *:*:00

```

(suite sur la page suivante)

(suite de la page précédente)

```

57 # hourly → *-*-* *:00:00
58 # daily → *-*-* 00:00:00
59 # monthly → *--01 00:00:00
60 # weekly → Mon *-*-* 00:00:00
61 # yearly → *-01-01 00:00:00
62 # quarterly → *-01,04,07,10-01 00:00:00
63 # semiannually → *-01,07-01 00:00:00
64 logbook: # all have to run on only one machine
65 # Sécurisation des journaux des opérations
66 - name: vitam-traceability-operations
67   frequency: "*-*-* 0/2:00:00" # every 2 hours
68 # Sécurisation des journaux du cycle de vie des groupes d'objets
69 - name: vitam-traceability-lfc-objectgroup
70   frequency: "*-*-* 0/4:00:00" # every 4 hours
71 # Sécurisation des journaux du cycle de vie des unités archivistiques
72 - name: vitam-traceability-lfc-unit
73   frequency: "*-*-* 0/3:00:00" # every 3 hours
74 # Audit de traçabilité
75 - name: vitam-traceability-audit
76   frequency: "*-*-* 00:00:00"
77 # Reconstruction
78 - name: vitam-logbook-reconstruction
79   frequency: "*-*-* *:0/5:00"
80 storage:
81 # Sauvegarde des journaux des écritures
82 - name: vitam-storage-accesslog-backup
83   frequency: "*-*-* 0/4:00:00" # every 4 hours
84 # Sécurisation du journal des écritures
85 - name: vitam-storage-log-backup
86   frequency: "*-*-* 0/2:00:00" # every 2 hours
87 # Log traceability
88 - name: vitam-storage-log-traceability
89   frequency: "*-*-* 0/2:10:00" # every 2 hours (10 minutes)
90 functional_administration:
91 - name: vitam-create-accession-register-symbolic
92   frequency: "*-*-* 00:00:00"
93 - name: vitam-functional-administration-accession-register-reconstruction
94   frequency: "*-*-* *:0/5:00"
95 - name: vitam-rule-management-audit
96   frequency: "*-*-* *:00:00"
97 - name: vitam-functional-administration-reconstruction
98   frequency: "*-*-* *:0/5:00"
99 metadata:
100 - name: vitam-metadata-store-graph
101   frequency: "*-*-* *:0/30:00"
102 - name: vitam-metadata-reconstruction
103   frequency: "*-*-* *:0/5:00"
104 - name: vitam-metadata-computed-inherited-rules
105   frequency: "*-*-* 02:30:00"
106 - name: vitam-metadata-purge-dip
107   frequency: "*-*-* 02:20:00"
108 - name: vitam-metadata-purge-transfers-SIP
109   frequency: "*-*-* 02:20:00"
110 offer:
111 # Compaction offer logs
112 - name: vitam-offer-log-compaction
113   frequency: "*-*-* *:00:00" # every hour

```

(suite sur la page suivante)

(suite de la page précédente)

```

114
115 ### consul ###
116 # FIXME: Consul à la racine pour le moment à cause de problèmes de récursivité
117 ↪ dans le parsing yaml
118 # WARNING: consul_domain should be a supported domain name for your organization
119 # You will have to generate server certificates with the same domain
120 ↪ name and the service subdomain name
121 # Example: consul_domain=vitam means you will have to generate some
122 ↪ certificates with .service.vitam domain
123 # access-external.service.vitam, ingest-external.service.vitam,
124 ↪ ...
125 consul_domain: consul
126 consul_component: consul
127 consul_folder_conf: "{{ vitam_defaults.folder.root_path }}/conf/{{ consul_
128 ↪ component }}"
129
130 # Workspace should be useless but storage have a dependency to it...
131 # elastic-kibana-interceptor is present as kibana is present, if kibana-data &
132 ↪ interceptor are not needed in the secondary site, just do not add them in the
133 ↪ hosts file
134 vitam_secondary_site_components: [ "logbook" , "metadata" , "functional-
135 ↪ administration" , "storage" , "storageofferdefault" , "offer" , "elasticsearch-
136 ↪ log" , "elasticsearch-data" , "logstash" , "kibana" , "mongoc" , "mongod" ,
137 ↪ "mongos" , "elastic-kibana-interceptor" , "consul" ]
138
139 # Vitams griffins required to launch preservation scenario
140 vitam_griffins: []
141
142 ### Composants Vitam ###
143
144 vitam:
145 # Ontology cache settings (max entries in cache & retention timeout in
146 ↪ seconds)
147 ontologyCacheMaxEntries: 100
148 ontologyCacheTimeoutInSeconds: 300
149 # Elasticsearch scroll timeout in milliseconds settings
150 elasticSearchScrollTimeoutInMilliseconds: 300000
151 accessexternal:
152 # Component name: do not modify
153 vitam_component: access-external
154 # DNS record for the service:
155 # Modify if ihm-demo is not using consul (typical production deployment)
156 host: "access-external.service.{{ consul_domain }}"
157 port_admin: 28102
158 port_service: 8444
159 baseuri: "access-external"
160 https_enabled: true
161 # Use platform secret for this component ? : do not modify
162 secret_platform: "false"
163 # Force the log level for this component: this are logback values (TRACE,
164 ↪ DEBUG, INFO, WARN, ERROR, OFF)
165 # If this var is not set, the default one will be used (vitam_defaults.
166 ↪ services.log_level)
167 # log_level: "DEBUG"
168 metrics_enabled: true
169 logback_rolling_policy: true
170 logback_max_file_size: "10MB"

```

(suite sur la page suivante)

(suite de la page précédente)

```

158     logback_total_size_cap:
159         file:
160             history_days: 10
161             totalsize: "5GB"
162         security:
163             history_days: 10
164             totalsize: "5GB"
165     jvm_log: false
166     performance_logger: "false"
167     reconstruction:
168     consul_check_business: 10 # value in seconds
169     consul_admin_check: 10 # value in seconds
170     acceptableRequestTime: 10 # value in seconds
171     # metricslevel: DEBUG
172     # metricsinterval: 3
173     # metricsunit: MINUTES
174     access_retention_days: 15 # Number of days for file retention
175     access_total_size_cap: "14GB" # total acceptable size
176     accessinternal:
177         vitam_component: access-internal
178         host: "access-internal.service.{{ consul_domain }}"
179         port_service: 8101
180         port_admin: 28101
181         baseuri: "access-internal"
182         https_enabled: false
183         secret_platform: "true"
184         # log_level: "DEBUG"
185         metrics_enabled: true
186         logback_rolling_policy: true
187         logback_max_file_size: "10MB"
188         logback_total_size_cap:
189             file:
190                 history_days: 10
191                 totalsize: "5GB"
192             security:
193                 history_days: 10
194                 totalsize: "5GB"
195         jvm_log: false
196         performance_logger: "false"
197         reconstruction:
198         consul_check_business: 10 # value in seconds
199         consul_admin_check: 10 # value in seconds
200         acceptableRequestTime: 10 # value in seconds
201         # metricslevel: DEBUG
202         # metricsinterval: 3
203         # metricsunit: MINUTES
204         access_retention_days: 15 # Number of days for file retention
205         access_total_size_cap: "14GB" # total acceptable size
206     functional_administration:
207         vitam_component: functional-administration
208         host: "functional-administration.service.{{ consul_domain }}"
209         port_service: 8004
210         port_admin: 18004
211         baseuri: "adminmanagement"
212         https_enabled: false
213         secret_platform: "true"
214         cluster_name: "{{ elasticsearch.data.cluster_name }}"

```

(suite sur la page suivante)

(suite de la page précédente)

```

215     # log_level: "DEBUG"
216     metrics_enabled: true
217     logback_rolling_policy: true
218     logback_max_file_size: "10MB"
219     logback_total_size_cap:
220       file:
221         history_days: 10
222         totalsize: "5GB"
223       security:
224         history_days: 10
225         totalsize: "5GB"
226     jvm_log: false
227     performance_logger: "false"
228     reconstruction:
229     consul_check_business: 10 # value in seconds
230     consul_admin_check: 10 # value in seconds
231     acceptableRequestTime: 10 # value in seconds
232     # metricslevel: DEBUG
233     # metricsinterval: 3
234     # metricsunit: MINUTES
235     access_retention_days: 15 # Number of days for file retention
236     access_total_size_cap: "14GB" # total acceptable size
237   elasticsearchinterceptor:
238     vitam_component: elastic-kibana-interceptor
239     host: "elastic-kibana-interceptor.service.{{ consul_domain }}"
240     port_service: 8014
241     port_admin: 18014
242     baseuri: ""
243     https_enabled: false
244     secret_platform: "false"
245     cluster_name: "{{ elasticsearch.data.cluster_name }}"
246     # log_level: "DEBUG"
247     metrics_enabled: true
248     logback_rolling_policy: true
249     logback_max_file_size: "10MB"
250     logback_total_size_cap:
251       file:
252         history_days: 10
253         totalsize: "5GB"
254       security:
255         history_days: 10
256         totalsize: "5GB"
257     jvm_log: false
258     performance_logger: "false"
259     reconstruction:
260     consul_check_business: 10 # value in seconds
261     consul_admin_check: 10 # value in seconds
262     acceptableRequestTime: 10 # value in seconds
263     # metricslevel: DEBUG
264     # metricsinterval: 3
265     # metricsunit: MINUTES
266     access_retention_days: 15 # Number of days for file retention
267     access_total_size_cap: "14GB" # total acceptable size
268   batchreport:
269     vitam_component: batch-report
270     host: "batch-report.service.{{ consul_domain }}"
271     port_service: 8015

```

(suite sur la page suivante)

(suite de la page précédente)

```

272     port_admin: 18015
273     baseuri: "batchreport"
274     https_enabled: false
275     secret_platform: "false"
276     # log_level: "DEBUG"
277     metrics_enabled: true
278     logback_rolling_policy: true
279     logback_max_file_size: "10MB"
280     logback_total_size_cap:
281       file:
282         history_days: 10
283         totalsize: "5GB"
284       security:
285         history_days: 10
286         totalsize: "5GB"
287     jvm_log: false
288     performance_logger: "false"
289     reconstruction:
290     consul_check_business: 10 # value in seconds
291     consul_admin_check: 10 # value in seconds
292     acceptableRequestTime: 10 # value in seconds
293     # metricslevel: DEBUG
294     # metricsinterval: 3
295     # metricsunit: MINUTES
296     access_retention_days: 15 # Number of days for file retention
297     access_total_size_cap: "14GB" # total acceptable size
298   ingestexternal:
299     vitam_component: ingest-external
300     # DNS record for the service:
301     # Modify if ihm-demo is not using consul (typical production deployment)
302     host: "ingest-external.service.{{ consul_domain }}"
303     port_admin: 28001
304     port_service: 8443
305     baseuri: "ingest-external"
306     https_enabled: true
307     secret_platform: "false"
308     antivirus: "clamav"
309     # uncomment if huge files need to be analyzed in more than 60s (default_
↪behavior)
310     #scantimeout: 60000 # value in milliseconds
311     # Directory where files should be placed for local ingest
312     upload_dir: "/vitam/data/ingest-external/upload"
313     # Directory where successful ingested files will be moved to
314     success_dir: "/vitam/data/ingest-external/upload/success"
315     # Directory where failed ingested files will be moved to
316     fail_dir: "/vitam/data/ingest-external/upload/failure"
317     # Action done to file after local ingest (see below for further_
↪information)
318     upload_final_action: "MOVE"
319     # log_level: "DEBUG"
320     # upload_final_action can be set to three different values (lower or_
↪upper case does not matter)
321     #   MOVE : After upload, the local file will be moved to either success_
↪dir or fail_dir depending on the status of the ingest towards ingest-internal
322     #   DELETE : After upload, the local file will be deleted if the upload_
↪succeeded
323     #   NONE : After upload, nothing will be done to the local file (default_
↪option set if the value entered for upload_final_action does not exist)

```

(suite sur la page suivante)

(suite de la page précédente)

```

324     metrics_enabled: true
325     logback_rolling_policy: true
326     logback_max_file_size: "10MB"
327     logback_total_size_cap:
328       file:
329         history_days: 10
330         totalsize: "5GB"
331       security:
332         history_days: 10
333         totalsize: "5GB"
334     jvm_log: false
335     performance_logger: "false"
336     reconstruction:
337     consul_check_business: 10 # value in seconds
338     consul_admin_check: 10 # value in seconds
339     acceptableRequestTime: 10 # value in seconds
340     # metricslevel: DEBUG
341     # metricsinterval: 3
342     # metricsunit: MINUTES
343     access_retention_days: 15 # Number of days for file retention
344     access_total_size_cap: "14GB" # total acceptable size
345 ingestinternal:
346     vitam_component: ingest-internal
347     host: "ingest-internal.service.{{ consul_domain }}"
348     port_service: 8100
349     port_admin: 28100
350     baseuri: "ingest"
351     https_enabled: false
352     secret_platform: "true"
353     # log_level: "DEBUG"
354     metrics_enabled: true
355     logback_rolling_policy: true
356     logback_max_file_size: "10MB"
357     logback_total_size_cap:
358       file:
359         history_days: 10
360         totalsize: "5GB"
361       security:
362         history_days: 10
363         totalsize: "5GB"
364     jvm_log: false
365     performance_logger: "false"
366     reconstruction:
367     consul_check_business: 10 # value in seconds
368     consul_admin_check: 10 # value in seconds
369     acceptableRequestTime: 10 # value in seconds
370     # metricslevel: DEBUG
371     # metricsinterval: 3
372     # metricsunit: MINUTES
373     access_retention_days: 15 # Number of days for file retention
374     access_total_size_cap: "14GB" # total acceptable size
375 ihm_demo:
376     vitam_component: ihm-demo
377     host: "ihm-demo.service.{{ consul_domain }}"
378     port_service: 8446
379     port_admin: 28002
380     baseurl: "/ihm-demo"

```

(suite sur la page suivante)

(suite de la page précédente)

```

381 static_content: "{{ vitam_defaults.folder.root_path }}/app/ihm-demo/v2"
382 baseuri: "ihm-demo"
383 https_enabled: true
384 secret_platform: "false"
385 # User session timeout in milliseconds (for shiro)
386 session_timeout: 1800000
387 secure_cookie: true
388 # Specify here the realms you want to use for authentication in ihm-demo
389 # You can set multiple realms, one per line
390 # With multiple realms, the user will be able to choose between the_
→allowed realms
391 # Example: authentication_realms:
392 #           - x509Realm
393 #           - ldapRealm
394 # Authorized values:
395 # x509Realm: certificate
396 # iniRealm: ini file
397 # ldapRealm: ldap
398 authentication_realms:
399 # - x509Realm
400 - iniRealm
401 # - ldapRealm
402 # log_level: "DEBUG"
403 allowedMediaTypes:
404 - type: "application"
405   subtype: "pdf"
406 - type: "text"
407   subtype: "plain"
408 - type: "image"
409   subtype: "jpeg"
410 - type: "image"
411   subtype: "tiff"
412 metrics_enabled: true
413 logback_rolling_policy: true
414 logback_max_file_size: "10MB"
415 logback_total_size_cap:
416   file:
417     history_days: 10
418     totalsize: "5GB"
419   security:
420     history_days: 10
421     totalsize: "5GB"
422 jvm_log: false
423 performance_logger: "false"
424 reconstruction:
425   consul_check_business: 10 # value in seconds
426   consul_admin_check: 10 # value in seconds
427   acceptableRequestTime: 10 # value in seconds
428   # metricslevel: DEBUG
429   # metricsinterval: 3
430   # metricsunit: MINUTES
431   access_retention_days: 15 # Number of days for file retention
432   access_total_size_cap: "14GB" # total acceptable size
433 logbook:
434   vitam_component: logbook
435   host: "logbook.service.{{ consul_domain }}"
436   port_service: 9002

```

(suite sur la page suivante)

(suite de la page précédente)

```

437     port_admin: 29002
438     baseuri: "logbook"
439     https_enabled: false
440     secret_platform: "true"
441     cluster_name: "{{ elasticsearch.data.cluster_name }}"
442     # Temporization delay (in seconds) for recent logbook operation events.
443     # Set it to a reasonable delay to cover max clock difference across_
↪servers + VM/GC pauses
444     operationTraceabilityTemporizationDelay: 300
445     # Temporization delay (in seconds) for recent logbook lifecycle events.
446     # Set it to a reasonable delay to cover max clock difference across_
↪servers + VM/GC pauses
447     lifecycleTraceabilityTemporizationDelay: 300
448     # Max entries selected per (Unit or Object Group) LFC traceability_
↪operation
449     lifecycleTraceabilityMaxEntries: 100000
450     # log_level: "DEBUG"
451     metrics_enabled: true
452     logback_rolling_policy: true
453     logback_max_file_size: "10MB"
454     logback_total_size_cap:
455         file:
456             history_days: 10
457             totalsize: "5GB"
458         security:
459             history_days: 10
460             totalsize: "5GB"
461     jvm_log: false
462     performance_logger: "false"
463     reconstruction:
464     consul_check_business: 10 # value in seconds
465     consul_admin_check: 10 # value in seconds
466     acceptableRequestTime: 10 # value in seconds
467     # metricslevel: DEBUG
468     # metricsinterval: 3
469     # metricsunit: MINUTES
470     access_retention_days: 15 # Number of days for file retention
471     access_total_size_cap: "14GB" # total acceptable size
472     metadata:
473     vitam_component: metadata
474     host: "metadata.service.{{ consul_domain }}"
475     port_service: 8200
476     port_admin: 28200
477     baseuri: "metadata"
478     https_enabled: false
479     secret_platform: "true"
480     cluster_name: "{{ elasticsearch.data.cluster_name }}"
481     # log_level: "DEBUG"
482     metrics_enabled: true
483     logback_rolling_policy: true
484     logback_max_file_size: "10MB"
485     logback_total_size_cap:
486         file:
487             history_days: 10
488             totalsize: "5GB"
489         security:
490             history_days: 10

```

(suite sur la page suivante)

(suite de la page précédente)

```

491         totalsize: "5GB"
492     jvm_log: false
493     performance_logger: "false"
494     reconstruction:
495     consul_check_business: 10 # value in seconds
496     consul_admin_check: 10 # value in seconds
497     # Archive Unit Profile cache settings (max entries in cache & retention_
↳timeout in seconds)
498     archiveUnitProfileCacheMaxEntries: 100
499     archiveUnitProfileCacheTimeoutInSeconds: 300
500     # Schema validator cache settings (max entries in cache & retention_
↳timeout in seconds)
501     schemaValidatorCacheMaxEntries: 100
502     schemaValidatorCacheTimeoutInSeconds: 300
503     acceptableRequestTime: 10 # value in seconds
504     # DIP cleanup delay (in minutes)
505     dipTimeToLiveInMinutes: 10080 # 7 days
506     transfersSIPTTimeToLiveInMinutes: 10080 # 7 days
507     # metricslevel: DEBUG
508     # metricsinterval: 3
509     # metricsunit: MINUTES
510     access_retention_days: 15 # Number of days for file retention
511     access_total_size_cap: "14GB" # total acceptable size
512 processing:
513     vitam_component: processing
514     host: "processing.service.{{ consul_domain }}"
515     port_service: 8203
516     port_admin: 28203
517     baseuri: "processing"
518     https_enabled: false
519     secret_platform: "true"
520     # log_level: "DEBUG"
521     metrics_enabled: true
522     logback_rolling_policy: true
523     logback_max_file_size: "10MB"
524     logback_total_size_cap:
525     file:
526     history_days: 10
527     totalsize: "5GB"
528     security:
529     history_days: 10
530     totalsize: "5GB"
531     jvm_log: false
532     performance_logger: "false"
533     maxDistributionInMemoryBufferSize: 100000
534     maxDistributionOnDiskBufferSize: 100000000
535     reconstruction:
536     consul_check_business: 10 # value in seconds
537     consul_admin_check: 10 # value in seconds
538     acceptableRequestTime: 10 # value in seconds
539     # metricslevel: DEBUG
540     # metricsinterval: 3
541     # metricsunit: MINUTES
542     access_retention_days: 15 # Number of days for file retention
543     access_total_size_cap: "14GB" # total acceptable size
544     security_internal:
545     vitam_component: security-internal

```

(suite sur la page suivante)

(suite de la page précédente)

```

546     host: "security-internal.service.{{ consul_domain }}"
547     port_service: 8005
548     port_admin: 28005
549     baseuri: "security-internal"
550     https_enabled: false
551     secret_platform: "true"
552     # log_level: "DEBUG"
553     metrics_enabled: true
554     logback_rolling_policy: true
555     logback_max_file_size: "10MB"
556     logback_total_size_cap:
557       file:
558         history_days: 10
559         totalsize: "5GB"
560     security:
561       history_days: 10
562       totalsize: "5GB"
563     jvm_log: false
564     performance_logger: "false"
565     reconstruction:
566       consul_check_business: 10 # value in seconds
567       consul_admin_check: 10 # value in seconds
568       acceptableRequestTime: 10 # value in seconds
569       # metricslevel: DEBUG
570       # metricsinterval: 3
571       # metricsunit: MINUTES
572       access_retention_days: 15 # Number of days for file retention
573       access_total_size_cap: "14GB" # total acceptable size
574     storageengine:
575       vitam_component: storage
576       host: "storage.service.{{ consul_domain }}"
577       port_service: 9102
578       port_admin: 29102
579       baseuri: "storage"
580       https_enabled: false
581       secret_platform: "true"
582       storageTraceabilityOverlapDelay: 300
583       restoreBulkSize: 1000
584       # batch thread pool size
585       minBatchThreadPoolSize: 4
586       maxBatchThreadPoolSize: 32
587       # Digest computation timeout in seconds
588       batchDigestComputationTimeout: 300
589       # Offer synchronization batch size & thread pool size
590       offerSynchronizationBulkSize: 1000
591       # Retries attempts
592       offerSyncNumberOfRetries: 3
593       offerSyncFirstAttemptWaitingTime: 15
594       offerSyncWaitingTime: 30
595       offerSyncThreadPoolSize: 32
596       # log_level: "DEBUG"
597       metrics_enabled: true
598       logback_rolling_policy: true
599       logback_max_file_size: "10MB"
600       logback_total_size_cap:
601         file:
602           history_days: 10

```

(suite sur la page suivante)

(suite de la page précédente)

```

603         totalsize: "5GB"
604     security:
605         history_days: 10
606         totalsize: "5GB"
607     offersync:
608         history_days: 10
609         totalsize: "5GB"
610     offerdiff:
611         history_days: 10
612         totalsize: "5GB"
613     jvm_log: false
614     # unit time per kB (in ms) used while calculating the timeout of an http_
↪request between storage and offer (if the calculated result is less than 60s,
↪this time is used)
615     timeoutMsPerKB: 100
616     performance_logger: "false"
617     reconstruction:
618     consul_check_business: 10 # value in seconds
619     consul_admin_check: 10 # value in seconds
620     acceptableRequestTime: 60 # value in seconds
621     # metricslevel: DEBUG
622     # metricsinterval: 3
623     # metricsunit: MINUTES
624     access_retention_days: 15 # Number of days for file retention
625     access_total_size_cap: "14GB" # total acceptable size
626     storageofferdefault:
627         vitam_component: "offer"
628         port_service: 9900
629         port_admin: 29900
630         baseuri: "offer"
631         https_enabled: false
632         secret_platform: "true"
633         # log_level: "DEBUG"
634         metrics_enabled: true
635         logback_rolling_policy: true
636         logback_max_file_size: "10MB"
637         logback_total_size_cap:
638             file:
639                 history_days: 10
640                 totalsize: "5GB"
641             security:
642                 history_days: 10
643                 totalsize: "5GB"
644             offer_tape:
645                 history_days: 10
646                 totalsize: "5GB"
647             offer_tape_backup:
648                 history_days: 10
649                 totalsize: "5GB"
650     jvm_log: false
651     performance_logger: "false"
652     reconstruction:
653     consul_check_business: 10 # value in seconds
654     consul_admin_check: 10 # value in seconds
655     acceptableRequestTime: 60 # value in seconds
656     # metricslevel: DEBUG
657     # metricsinterval: 3

```

(suite sur la page suivante)

(suite de la page précédente)

```

658     # metricsunit: MINUTES
659     access_retention_days: 15 # Number of days for file retention
660     access_total_size_cap: "14GB" # total acceptable size
661 worker:
662     vitam_component: worker
663     host: "worker.service.{{ consul_domain }}"
664     port_service: 9104
665     port_admin: 29104
666     baseuri: "worker"
667     https_enabled: false
668     secret_platform: "true"
669     # log_level: "DEBUG"
670     metrics_enabled: true
671     logback_rolling_policy: true
672     logback_max_file_size: "10MB"
673     logback_total_size_cap:
674         file:
675             history_days: 10
676             totalsize: "5GB"
677         security:
678             history_days: 10
679             totalsize: "5GB"
680     jvm_log: false
681     performance_logger: "false"
682     reconstruction:
683     consul_check_business: 10 # value in seconds
684     consul_admin_check: 10 # value in seconds
685     acceptableRequestTime: 60 # value in seconds
686     api_output_index_tenants: [0,1,2,3,4,5,6,7,8,9]
687     rules_index_tenants: [0,1,2,3,4,5,6,7,8,9]
688     # Archive Unit Profile cache settings (max entries in cache & retention_
↳timeout in seconds)
689     archiveUnitProfileCacheMaxEntries: 100
690     archiveUnitProfileCacheTimeoutInSeconds: 300
691     # Schema validator cache settings (max entries in cache & retention_
↳timeout in seconds)
692     schemaValidatorCacheMaxEntries: 100
693     schemaValidatorCacheTimeoutInSeconds: 300
694     # metricslevel: DEBUG
695     # metricsinterval: 3
696     # metricsunit: MINUTES
697     access_retention_days: 15 # Number of days for file retention
698     access_total_size_cap: "14GB" # total acceptable size
699 workspace:
700     vitam_component: workspace
701     host: "workspace.service.{{ consul_domain }}"
702     port_service: 8201
703     port_admin: 28201
704     baseuri: "workspace"
705     https_enabled: false
706     secret_platform: "true"
707     # log_level: "DEBUG"
708     metrics_enabled: true
709     logback_rolling_policy: true
710     logback_max_file_size: "10MB"
711     logback_total_size_cap:
712         file:

```

(suite sur la page suivante)

(suite de la page précédente)

```

713     history_days: 10
714     totalsize: "5GB"
715     security:
716         history_days: 10
717         totalsize: "5GB"
718     jvm_log: false
719     performance_logger: "false"
720     reconstruction:
721     consul_check_business: 10 # value in seconds
722     consul_admin_check: 10 # value in seconds
723     acceptableRequestTime: 10 # value in seconds
724     # metricslevel: DEBUG
725     # metricsinterval: 3
726     # metricsunit: MINUTES
727     access_retention_days: 15 # Number of days for file retention
728     access_total_size_cap: "14GB" # total acceptable size
729
730 # for functional-administration, manage master/slave tenant configuration
731 vitam_tenants_usage_external:
732 - name: 0
733   identifiers:
734     - INGEST_CONTRACT
735     - ACCESS_CONTRACT
736     - MANAGEMENT_CONTRACT
737     - ARCHIVE_UNIT_PROFILE
738 - name: 1
739   identifiers:
740     - INGEST_CONTRACT
741     - ACCESS_CONTRACT
742     - MANAGEMENT_CONTRACT
743     - PROFILE
744     - SECURITY_PROFILE
745     - CONTEXT
746
747 vitam_tenant_rule_duration:
748 - name: 2 # applied tenant
749   rules:
750     - AppraisalRule : "1 year" # rule name : rule value
751
752 # If you want to deploy vitam in a single VM, add the vm name in this array
753 single_vm_hostnames: ['localhost']

```

Note : Cas du composant ingest-external. Les directives upload_dir, success_dir, fail_dir et upload_final_action permettent de prendre en charge (ingest) des fichiers déposés dans upload_dir et appliquer une règle upload_final_action à l'issue du traitement (NONE, DELETE ou MOVE dans success_dir ou fail_dir selon le cas). Se référer au [DEX](#) pour de plus amples détails. Se référer au manuel de développement pour plus de détails sur l'appel à ce cas.

Avertissement : Selon les informations apportées par le métier, redéfinir les valeurs associées dans les directives classificationList et classificationLevelOptional. Cela permet de définir quels niveaux de protection du secret de la défense nationale, supporte l'instance. Attention : une instance de niveau supérieur doit toujours supporter les niveaux inférieurs.

- environments /group_vars/all/cots_vars.yml, comme suit :

```

1  ---
2
3  consul:
4      retry_interval: 10 # in seconds
5      check_interval: 10 # in seconds
6      check_timeout: 5 # in seconds
7      network: "ip_admin" # Which network to use for consul communications ? ip_
      ↪admin or ip_service ?
8
9  consul_remote_sites:
10     # wan contains the wan addresses of the consul server instances of the_
      ↪external vitam sites
11     # Exemple, if our local dc is dc1, we will need to set dc2 & dc3 wan conf:
12     # - dc2:
13     #   wan: ["10.10.10.10", "1.1.1.1"]
14     # - dc3:
15     #   wan: ["10.10.10.11", "1.1.1.1"]
16 # Please uncomment and fill values if you want to connect VITAM to external SIEM
17 # external_siem:
18 #   host:
19 #   port:
20
21 elasticsearch:
22     log:
23         host: "elasticsearch-log.service.{{ consul_domain }}"
24         port_http: "9201"
25         groupe: "log"
26         baseuri: "elasticsearch-log"
27         cluster_name: "elasticsearch-log"
28         consul_check_http: 10 # in seconds
29         consul_check_tcp: 10 # in seconds
30         action_log_level: error
31         https_enabled: false
32         indices_fielddata_cache_size: 0.3 # related to https://www.elastic.co/
      ↪guide/en/elasticsearch/reference/6.8/modules-fielddata.html
33         indices_breaker_fielddata_limit: 0.4 # related to https://www.elastic.co/
      ↪guide/en/elasticsearch/reference/6.8/circuit-breaker.html#fielddata-circuit-
      ↪breaker
34         dynamic_timeout: 30s
35         # default index template
36         index_templates:
37             default:
38                 shards: 1
39                 replica: 1
40                 packetbeat:
41                     shards: 5
42         log_appenders:
43             root:
44                 log_level: "info"
45             rolling:
46                 max_log_file_size: "100MB"
47                 max_total_log_size: "5GB"
48                 max_files: "50"
49             deprecation_rolling:
50                 max_log_file_size: "100MB"
51                 max_total_log_size: "1GB"

```

(suite sur la page suivante)

(suite de la page précédente)

```

52         max_files: "10"
53         log_level: "warn"
54     index_search_slowlog_rolling:
55         max_log_file_size: "100MB"
56         max_total_log_size: "1GB"
57         max_files: "10"
58         log_level: "warn"
59     index_indexing_slowlog_rolling:
60         max_log_file_size: "100MB"
61         max_total_log_size: "1GB"
62         max_files: "10"
63         log_level: "warn"
64     # By default, is commented. Should be uncommented if ansible computes
↳badly vCPUs number ; values are associated vCPUs numbers ; please adapt to
↳your configuration
65     # thread_pool:
66     #     index:
67     #         size: 2
68     #     get:
69     #         size: 2
70     #     search:
71     #         size: 2
72     #     write:
73     #         size: 2
74     #     warmer:
75     #         max: 2
76     data:
77         host: "elasticsearch-data.service.{{ consul_domain }}"
78         # default is 0.1 (10%) and should be quite enough in most cases
79         #index_buffer_size_ratio: "0.15"
80         port_http: "9200"
81         groupe: "data"
82         baseuri: "elasticsearch-data"
83         cluster_name: "elasticsearch-data"
84         consul_check_http: 10 # in seconds
85         consul_check_tcp: 10 # in seconds
86         action_log_level: debug
87         https_enabled: false
88         # discovery_zen_minimum_master_nodes: 2 # comented by default ; by
↳default, value is half the length of ansible associated group whose racks have
↳the same number of machine. If it is not the case, this value have to be set
↳with the smallest rack (if using param is_balancing). ONLY EXISTS FOR DATA
↳CLUSTER !!!! DO NOT FORGET TO APPLY PARAMETER WITH REPLICA NUMBER !!!!
89         indices fielddata_cache_size: 0.3 # related to https://www.elastic.co/
↳guide/en/elasticsearch/reference/6.5/modules-fielddata.html
90         indices_breaker_fielddata_limit: 0.4 # related to https://www.elastic.co/
↳guide/en/elasticsearch/reference/6.5/circuit-breaker.html#fielddata-circuit-
↳breaker
91         dynamic_timeout: 30s
92         # default index template
93         index_templates:
94             default:
95                 shards: 10
96                 replica: 2
97         log_appenders:
98             root:
99                 log_level: "info"

```

(suite sur la page suivante)

(suite de la page précédente)

```

100     rolling:
101         max_log_file_size: "100MB"
102         max_total_log_size: "5GB"
103         max_files: "50"
104     deprecation_rolling:
105         max_log_file_size: "100MB"
106         max_total_log_size: "5GB"
107         max_files: "50"
108         log_level: "warn"
109     index_search_slowlog_rolling:
110         max_log_file_size: "100MB"
111         max_total_log_size: "5GB"
112         max_files: "50"
113         log_level: "warn"
114     index_indexing_slowlog_rolling:
115         max_log_file_size: "100MB"
116         max_total_log_size: "5GB"
117         max_files: "50"
118         log_level: "warn"
119     # By default, is commented. Should be uncommented if ansible computes
120     ↪ badly vCPUs number ; values are associated vCPUs numbers ; please adapt to
121     ↪ your configuration
122     # thread_pool:
123     #     index:
124     #         size: 2
125     #     get:
126     #         size: 2
127     #     search:
128     #         size: 2
129     #     write:
130     #         size: 2
131     #     warmer:
132     #         max: 2
133
134 mongoddb:
135     mongos_port: 27017
136     mongoc_port: 27018
137     mongod_port: 27019
138     mongo_authentication: "true"
139     host: "mongos.service.{{ consul_domain }}"
140     check_consul: 10 # in seconds
141     drop_info_log: false # Drop mongo (I)nformational log, for Verbosity Level of
142     ↪ 0
143
144 logstash:
145     host: "logstash.service.{{ consul_domain }}"
146     user: logstash
147     port: 10514
148     rest_port: 20514
149     check_consul: 10 # in seconds
150     # logstash xms & xmx in Megabytes
151     # jvm_xms: 2048
152     # jvm_xmx: 2048
153     # workers_number: 4
154     log_appenders:
155         rolling:
156             max_log_file_size: "100MB"

```

(suite sur la page suivante)

(suite de la page précédente)

```

154         max_total_log_size: "5GB"
155     json_rolling:
156         max_log_file_size: "100MB"
157         max_total_log_size: "5GB"
158
159 # Curator units: days
160 curator:
161     log:
162         metrics:
163             close: 5
164             delete: 30
165         logstash:
166             close: 5
167             delete: 30
168         metricbeat:
169             close: 5
170             delete: 30
171         packetbeat:
172             close: 5
173             delete: 30
174
175 kibana:
176     header_value: "reporting"
177     import_delay: 10
178     import_retries: 10
179     log:
180         baseuri: "kibana_log"
181         api_call_timeout: 120
182         groupe: "log"
183         port: 5601
184         default_index_pattern: "logstash-vitam*"
185         check_consul: 10 # in seconds
186         # default shards & replica
187         shards: 5
188         replica: 1
189         # pour index logstash-*
190         metrics:
191             shards: 5
192             replica: 1
193         # pour index metrics-vitam-*
194         logs:
195             shards: 5
196             replica: 1
197         # pour index metricbeat-*
198         metricbeat:
199             shards: 5 # must be a factor of 30
200             replica: 1
201     data:
202         baseuri: "kibana_data"
203         # OMA : bugdette : api_call_timeout is used for retries ; should ceate a
204         ↪ separate variable rather than this one
205         api_call_timeout: 120
206         groupe: "data"
207         port: 5601
208         default_index_pattern: "logbookoperation_*"
209         check_consul: 10 # in seconds
210         # index template for .kibana

```

(suite sur la page suivante)

(suite de la page précédente)

```

210     shards: 1
211     replica: 1
212
213 syslog:
214     # value can be syslog-ng or rsyslog
215     name: "rsyslog"
216
217 cerebro:
218     baseuri: "cerebro"
219     port: 9000
220     check_consul: 10 # in seconds
221
222 siegfried:
223     port: 19000
224     consul_check: 10 # in seconds
225
226 clamav:
227     port: 3310
228     # frequency freshclam for database update per day (from 0 to 24 - 24 meaning
↪hourly check)
229     db_update_periodicity: 1
230
231 mongo_express:
232     baseuri: "mongo-express"
233
234 ldap_authentication:
235     ldap_protocol: "ldap"
236     ldap_server: "{% if groups['ldap']|length > 0 %}{{ groups['ldap']|first }}{%
↪endif %}"
237     ldap_port: "389"
238     ldap_base: "dc=programmevitam,dc=fr"
239     ldap_login: "cn=Manager,dc=programmevitam,dc=fr"
240     uid_field: "uid"
241     ldap_userDn_Template: "uid={0},ou=people,dc=programmevitam,dc=fr"
242     ldap_group_request: "(&(objectClass=groupOfNames)(member={0}))"
243     ldap_admin_group: "cn=admin,ou=groups,dc=programmevitam,dc=fr"
244     ldap_user_group: "cn=user,ou=groups,dc=programmevitam,dc=fr"
245     ldap_guest_group: "cn=guest,ou=groups,dc=programmevitam,dc=fr"
246
247 java_prerequisites:
248     debian: "openjdk-11-jre-headless"
249     redhat: "java-11-openjdk-headless"

```

Note : Installation multi-sites. Déclarer dans `consul_remote_sites` les datacenters Consul des autres site ; se référer à l'exemple fourni pour renseigner les informations.

Note : Concernant Curator, en environnement de production, il est recommandé de procéder à la fermeture des index au bout d'une semaine pour les index de type « logstash » (3 jours pour les index « metrics »), qui sont le reflet des traces applicatives des composants de la solution logicielle *VITAM*. Il est alors recommandé de lancer le *delete* de ces index au bout de la durée minimale de rétention : 1 an (il n'y a pas de durée de rétention minimale légale sur les index « metrics », qui ont plus une vocation technique et, éventuellement, d'investigations).

- `environments/group_vars/all/jvm_vars.yml`, comme suit :


```

1 ---
2
3 vitam:
4   accessinternal:
5     jvm_opts:
6       # memory: "-Xms512m -Xmx512m"
7       # gc: ""
8       # java: ""
9   accessexternal:
10    jvm_opts:
11      # memory: "-Xms512m -Xmx512m"
12      # gc: ""
13      # java: ""
14   elasticsearch:
15     jvm_opts:
16       # memory: "-Xms512m -Xmx512m"
17       # gc: ""
18       # java: ""
19   batchreport:
20     jvm_opts:
21       # memory: "-Xms512m -Xmx512m"
22       # gc: ""
23       # java: ""
24   ingestinternal:
25     jvm_opts:
26       # memory: "-Xms512m -Xmx512m"
27       # gc: ""
28       # java: ""
29   ingestexternal:
30     jvm_opts:
31       # memory: "-Xms512m -Xmx512m"
32       # gc: ""
33       # java: ""
34   metadata:
35     jvm_opts:
36       # memory: "-Xms512m -Xmx512m"
37       # gc: ""
38       # java: ""
39   ihm_demo:
40     jvm_opts:
41       # memory: "-Xms512m -Xmx512m"
42       # gc: ""
43       # java: ""
44   ihm_recette:
45     jvm_opts:
46       # memory: "-Xms512m -Xmx512m"
47       # gc: ""
48       # java: ""
49   logbook:
50     jvm_opts:
51       # memory: "-Xms512m -Xmx512m"
52       # gc: ""
53       # java: ""
54   workspace:
55     jvm_opts:
56       # memory: "-Xms512m -Xmx512m"
57       # gc: ""

```

(suite sur la page suivante)

(suite de la page précédente)

```

58     # java: ""
59 processing:
60     jvm_opts:
61         # memory: "-Xms512m -Xmx512m"
62         # gc: ""
63         # java: ""
64 worker:
65     jvm_opts:
66         # memory: "-Xms512m -Xmx512m"
67         # gc: ""
68         # java: ""
69 storageengine:
70     jvm_opts:
71         # memory: "-Xms512m -Xmx512m"
72         # gc: ""
73         # java: ""
74 storageofferdefault:
75     jvm_opts:
76         # memory: "-Xms512m -Xmx512m"
77         # gc: ""
78         # java: ""
79 functional_administration:
80     jvm_opts:
81         # memory: "-Xms512m -Xmx512m"
82         # gc: ""
83         # java: ""
84 security_internal:
85     jvm_opts:
86         # memory: "-Xms512m -Xmx512m"
87         # gc: ""
88         # java: ""
89 library:
90     jvm_opts:
91         memory: "-Xms32m -Xmx128m"
92         # gc: ""
93         # java: ""

```

Note : Cette configuration est appliquée à la solution logicielle *VITAM* ; il est possible de créer un tuning par « groupe » défini dans ansible.

4.2.5.13 Paramétrage de l'Offre Froide (librairies de cartouches)

Suite à l'introduction des offres bandes, plusieurs notions supplémentaires sont prises en compte dans ce fichier. De nouvelles entrées ont été ajoutées pour décrire d'une part le matériel robotique assigné à l'offre froide, et les répertoires d'échanges temporaires d'autre part. Les éléments de configuration doivent être renseignés par l'exploitant.

- Lecture asynchrone

Un paramètre a été ajouté aux définitions de stratégie. *AsyncRead* permet de déterminer si l'offre associée fonctionne en lecture asynchrone, et désactive toute possibilité de lecture directe sur l'offre. Une offre froide « offer-tape » doit être configurée en lecture asynchrone. La valeur par défaut pour *asyncRead* est False.

Exemple :

```
vitam_strategy:
- name: offer-tape-1
  referent: false
  asyncRead: **true**
- name: offer-fs-2
  referent: true
  asyncRead: false
```

- Périphériques liés à l'usage des bandes magnétiques

Terminologie :

- **tapeLibrary** une librairie de bande dans son ensemble. Une *tapeLibrary* est constituée de 1 à n « robot » et de 1 à n « drives ». Une offre froide nécessite la déclaration d'au moins une librairie pour fonctionner. L'exploitant doit déclarer un identifiant pour chaque librairie. Ex : TAPE_LIB_1
- **drive** un drive est un lecteur de cartouches. Il doit être identifié par un *path* scsi unique. Une offre froide nécessite la déclaration d'au moins un lecteur pour fonctionner.

Note : il existe plusieurs fichiers périphériques sur Linux pour un même lecteur

Les plus classiques sont par exemple `/dev/st0` et `/dev/nst0` pour le premier drive détecté par le système. L'usage de `/dev/st0` indique au système que la bande utilisée dans le lecteur associé devra être rembobinée après l'exécution de la commande appelante. A contrario, `/dev/nst0` indique au système que la bande utilisée dans le lecteur associé devra rester positionnée après le dernier marqueur de fichier utilisé par l'exécution de la commande appelante.

Important : Pour que l'offre froide fonctionne correctement, il convient de configurer une version `/dev/nstxx`

Note : Il peut arriver sur certains systèmes que l'ordre des lecteurs de bandes varient après un reboot de la machine. Pour s'assurer la persistance de l'ordre des lecteurs dans la configuration VITAM, il est conseillé d'utiliser les fichiers périphériques présents dans `/dev/tape/by-id/` qui s'appuient sur des références au hardware pour définir les drives.

- **robot** un robot est le composant chargé de procéder au déplacement des cartouches dans une *tapeLibrary*, et de procéder à l'inventaire de ses ressources. Une offre froide nécessite la déclaration d'au moins un robot pour fonctionner. L'exploitant doit déclarer un fichier de périphérique scsi générique (ex : `/dev/sg4`) associé à la robotique sur son système. A l'instar de la configuration des drives, il est recommandé d'utiliser le device présent dans `/dev/tape/by-id` pour déclarer les robots.

Définition d'une offre froide :

Une offre froide (OF) doit être définie dans la rubrique « `vitam_offers` » avec un provider de type *tape-library*

Exemple :

```
vitam_offers:
  offer-tape-1:
    provider: tape-library
    tapeLibraryConfiguration:
```

La description « tapeLibraryConfiguration » débute par la définition des répertoires de sockage ainsi que le paramétrage des *tar*.

inputFileStorageFolder Répertoire où seront stockés les objets à intégrer à l'OF **inputTarStorageFolder** Répertoire où seront générés et stockés les tars avant transfert sur bandes **outputTarStorageFolder** Répertoire où seront rapatriés les *tars* depuis les bandes. **MaxTarEntrySize** Taille maximale au-delà de la laquelle les fichiers entrant seront découpés en segment, en octets **maxTarFileSize** Taille maximale des tars à constituer, en octets. **forceOverrideNonEmptyCartridge** Permet de passer outre le contrôle vérifiant que les bandes nouvellement introduites sont vides. Par défaut à *false* **useSudo** Réservé à un usage futur – laisser à *false*.

Note : N.B. : MaxTarEntrySize doit être strictement inférieur à maxTarFileSize

Exemple :

```
inputFileStorageFolder: "/vitam/data/offer/offer/inputFiles"
inputTarStorageFolder: "/vitam/data/offer/offer/inputTars"
outputTarStorageFolder: "/vitam/data/offer/offer/outputTars"
maxTarEntrySize: 10000000
maxTarFileSize: 10000000000
ForceOverrideNonEmptyCartridge: False
useSudo: false
```

Par la suite, un paragraphe « topology » décrivant la topologie de l'offre doit être renseigné. L'objectif de cet élément est de pouvoir définir une segmentation de l'usage des bandes pour répondre à un besoin fonctionnel. Il convient ainsi de définir des *buckets*, qu'on peut voir comme un ensemble logique de bandes, et de les associer à un ou plusieurs tenants.

tenants tableau de 1 à n identifiants de tenants au format [1,...,n] **tarBufferingTimeoutInMinutes** Valeur en minutes durant laquelle un *tar* peut rester ouvert

Exemple :

```
topology:
  buckets:
    test:
      tenants: [0]
      tarBufferingTimeoutInMinutes: 60
    admin:
      tenants: [1]
      tarBufferingTimeoutInMinutes: 60
    prod:
      tenants: [2,3,4,5,6,7,8,9]
      tarBufferingTimeoutInMinutes: 60
```

Enfin, la définition des équipements robotiques proprement dite doit être réalisée dans le paragraphe « tapeLibraries ».

robots : Définition du bras robotique de la librairie.

device : Chemin du fichier de périphérique scsi générique associé au bras.

mtxPath : Chemin vers la commande Linux de manipulation du bras.

timeoutInMilliseconds : timeout en millisecondes à appliquer aux ordres du bras.

drives : Définition du/ou des lecteurs de cartouches de la librairie.

index : Numéro de lecteur, valeur débutant à 0

device : Chemin du fichier de périphérique scsi SANS REMBOBINAGE associé au lecteur.

mtPath : Chemin vers la commande Linux de manipulation des lecteurs.

ddPath : Chemin vers la commande Linux de copie de bloc de données.

tarPath : Chemin vers la commande Linux de création d'archives tar.

timeoutInMilliseconds : timeout en millisecondes à appliquer aux ordres du lecteur.

Exemple :

```
tapeLibraries:
  TAPE_LIB_1:
    robots:
      -
        device: /dev/tape/by-id/scsiQUANTUM_10F73224E6664C84A1D00000
        mtPath: "/usr/sbin/mtx"
        timeoutInMilliseconds: 3600000
    drives:
      -
        index: 0
        device: /dev/tape/by-id/scsi-1IBM_ULT3580-TD6_1235308739-nst
        mtPath: "/bin/mt"
        ddPath: "/bin/dd"
        tarPath: "/bin/tar"
        timeoutInMilliseconds: 3600000
      -
        index: 1
        device: /dev/tape/by-id/scsi-1IBM_ULT3580-TD6_0951859786-nst
        mtPath: "/bin/mt"
        ddPath: "/bin/dd"
        tarPath: "/bin/tar"
        timeoutInMilliseconds: 3600000
      -
        index: 2
        device: /dev/tape/by-id/scsi-1IBM_ULT3580-TD6_0269493808-nst
        mtPath: "/bin/mt"
        ddPath: "/bin/dd"
        tarPath: "/bin/tar"
        timeoutInMilliseconds: 3600000
      -
        index: 3
        device: /dev/tape/by-id/scsi-1IBM_ULT3580-TD6_0566471858-nst
        mtPath: "/bin/mt"
        ddPath: "/bin/dd"
        tarPath: "/bin/tar"
        timeoutInMilliseconds: 3600000
```

4.2.5.14 Sécurisation SELinux

Depuis la release R13, la solution logicielle *VITAM* prend désormais en charge l'activation de SELinux sur le périmètre du composant worker et des processus associés aux *griffins* (greffons de préservation).

SELinux (Security-Enhanced Linux) permet de définir des politiques de contrôle d'accès à différents éléments du système d'exploitation en répondant essentiellement à la question « May <subject> do <action> to <object> », par exemple « May a web server access files in users' home directories ».

Chaque processus est ainsi confiné à un (voire plusieurs) domaine(s), et les fichiers sont étiquetés en conséquence. Un processus ne peut ainsi accéder qu'aux fichiers étiquetés pour le domaine auquel il est confiné.

Note : La solution logicielle *VITAM* ne gère actuellement que le mode *targeted* (« only *targeted* processes are protected »)

Les enjeux de la sécurisation SELinux dans le cadre de la solution logicielle *VITAM* sont de garantir que les processus associés aux *griffins* (greffons de préservation) n'auront accès qu'au ressources système strictement requises pour leur fonctionnement et leurs échanges avec les composants *worker*.

Note : La solution logicielle *VITAM* ne gère actuellement SELinux que pour le système d'exploitation Centos

Avertissement : SELinux n'a pas vocation à remplacer quelque système de sécurité existant, mais vise plutôt à les compléter. Aussi, la mise en place de politiques de sécurité reste de mise et à la charge de l'exploitant. Par ailleurs, l'implémentation SELinux proposée avec la solution logicielle *VITAM* est minimale et limitée au greffon de préservation Siegfried. Cette implémentation pourra si nécessaire être complétée ou améliorée par le projet d'implémentation.

SELinux propose trois modes différents :

- *Enforcing* : dans ce mode, les accès sont restreints en fonction des règles SELinux en vigueur sur la machine ;
- *Permissive* : ce mode est généralement à considérer comme un mode de débogage. En mode permissif, les règles SELinux seront interrogées, les erreurs d'accès logguées, mais l'accès ne sera pas bloqué.
- *Disabled* : SELinux est désactivé. Rien ne sera restreint, rien ne sera loggué.

La mise en oeuvre de SELinux est prise en charge par le processus de déploiement et s'effectue de la sorte :

- Isoler dans l'inventaire de déploiement les composants worker sur des hosts dédiés (ne contenant aucun autre composant *VITAM*)
- Positionner pour ces hosts un fichier *hostvars* sous *environments/host_vars/* contenant la déclaration suivante :

```
selinux_state: "enforcing"
```

- Procéder à l'installation de la solution logicielle *VITAM* grâce aux playbooks ansible fournis, et selon la procédure d'installation classique décrite dans le DIN

4.2.6 Procédure de première installation

4.2.6.1 Déploiement

4.2.6.1.1 Cas particulier : utilisation de ClamAv en environnement Debian

Dans le cas de l'installation en environnement Debian, la base de données n'est pas intégrée avec l'installation de ClamAv, C'est la commande `freshclam` qui en assure la charge. Si vous n'êtes pas connecté à internet, la base de données doit être installée manuellement. Les liens suivants indiquent la procédure à suivre : [Installation ClamAv](#)¹⁵ et [Section Virus Database](#)¹⁶

<https://www.clamav.net/documents/installing-clamav>
<https://www.clamav.net/downloads>

4.2.6.1.2 Fichier de mot de passe

Par défaut, le mot de passe des *vault* sera demandé à chaque exécution d'ansible. Si le fichier `deployment/vault_pass.txt` est renseigné avec le mot de passe du fichier `environments/group_vars/all/vault-vitam.yml`, le mot de passe ne sera pas demandé (dans ce cas, changez l'option `--ask-vault-pass` des invocations ansible par l'option `--vault-password-file=VAULT_PASSWORD_FILES`).

4.2.6.1.3 Mise en place des repositories VITAM (optionnel)

VITAM fournit un playbook permettant de définir sur les partitions cible la configuration d'appel aux repositories spécifiques à *VITAM* :

Editer le fichier `environments/group_vars/all/repositories.yml` à partir des modèles suivants (décommenter également les lignes) :

Pour une cible de déploiement CentOS :

```
1 #vitam_repositories:
2 #- key: repo 1
3 #   value: "file:///code"
4 #   proxy: http://proxy
5 #- key: repo 2
6 #   value: "http://www.programmevitam.fr"
7 #   proxy: _none_
8 #- key: repo 3
9 #   value: "ftp://centos.org"
10 #   proxy:
```

Pour une cible de déploiement Debian :

```
1 #vitam_repositories:
2 #- key: repo 1
3 #   value: "file:///code"
4 #   subtree: "/"
5 #   trusted: "[trusted=yes]"
6 #- key: repo 2
7 #   value: "http://www.programmevitam.fr"
8 #   subtree: "/"
9 #   trusted: "[trusted=yes]"
10 #- key: repo 3
11 #   value: "ftp://centos.org"
12 #   subtree: "binary"
13 #   trusted: "[trusted=yes]"
```

Ce fichier permet de définir une liste de repositories. Décommenter et adapter à votre cas.

Pour mettre en place ces repositories sur les machines cibles, lancer la commande :

```
ansible-playbook ansible-vitam-extra/bootstrap.yml -i environments/<fichier d
↪ 'inventaire> --ask-vault-pass
```

Note : En environnement CentOS, il est recommandé de créer des noms de *repository* commençant par *vitam-* .

4.2.6.1.4 Génération des *hostvars*

Une fois l'étape de *PKI* effectuée avec succès, il convient de procéder à la génération des *hostvars*, qui permettent de définir quelles interfaces réseau utiliser. Actuellement la solution logicielle *VITAM* est capable de gérer 2 interfaces réseau :

- Une d'administration
- Une de service

4.2.6.1.4.1 Cas 1 : Machines avec une seule interface réseau

Si les machines sur lesquelles *VITAM* sera déployé ne disposent que d'une interface réseau, ou si vous ne souhaitez en utiliser qu'une seule, il convient d'utiliser le playbook `ansible-vitam/generate_hostvars_for_1_network_interface.yml`

Cette définition des *host_vars* se base sur la directive `ansible_default_ipv4.address`, qui se base sur l'adresse *IP* associée à la route réseau définie par défaut.

Avertissement : Les communications d'administration et de service transiteront donc toutes les deux via l'unique interface réseau disponible.

4.2.6.1.4.2 Cas 2 : Machines avec plusieurs interfaces réseau

Si les machines sur lesquelles *VITAM* sera déployé disposent de plusieurs interfaces et si celles-ci respectent cette règle :

- Interface nommée `eth0` = `ip_service`
- Interface nommée `eth1` = `ip_admin`

Alors il est possible d'utiliser le playbook `ansible-vitam-extra/generate_hostvars_for_2_network_interfaces.yml`

Note : Pour les autres cas de figure, il sera nécessaire de générer ces *hostvars* à la main ou de créer un script pour automatiser cela.

4.2.6.1.4.3 Vérification de la génération des *hostvars*

A l'issue, vérifier le contenu des fichiers générés sous `environments/host_vars/` et les adapter au besoin.

Prudence : Cas d'une installation multi-sites. Sur site secondaire, s'assurer que, pour les machines hébergeant les offres, la directive `ip_wan` a bien été déclarée (l'ajouter manuellement, le cas échéant), pour que le site *primaire* sache les contacter via une IP particulière. Par défaut, c'est l'IP de service qui sera utilisée.

4.2.6.1.5 Déploiement

Une fois les étapes précédentes correctement effectuées (en particulier, la section *Génération des magasins de certificats* (page 51)), le déploiement s'effectue depuis la machine *ansible* et va distribuer la solution *VITAM* selon l'inventaire correctement renseigné.

Une fois l'étape de la génération des hosts effectuée avec succès, le déploiement est à réaliser avec la commande suivante :

```
ansible-playbook ansible-vitam/vitam.yml -i environments/<fichier d'inventaire> --ask-  
↪ vault-pass
```

Note : Une confirmation est demandée pour lancer ce script. Il est possible de rajouter le paramètre `-e confirmation=yes` pour bypasser cette demande de confirmation (cas d'un déploiement automatisé).

Prudence : Dans le cas où l'installateur souhaite utiliser un *repository* de binaires qu'il gère par lui-même, il est fortement recommandé de rajouter `--skip-tags "enable_vitam_repo"` à la commande `ansible-playbook` ; dans ce cas, le comportement de `yum` n'est pas impacté par la solution de déploiement.

4.2.7 Elements *extras* de l'installation

Prudence : Les éléments décrits dans cette section sont des éléments « extras » ; il ne sont pas officiellement supportés, et ne sont par conséquent pas inclus dans l'installation de base. Cependant, ils peuvent s'avérer utile, notamment pour les installations sur des environnements hors production.

Prudence : Dans le cas où l'installateur souhaite utiliser un *repository* de binaires qu'il gère par lui-même, il est fortement recommandé de rajouter `--skip-tags "enable_vitam_repo"` à la commande `ansible-playbook` ; dans ce cas, le comportement de `yum` n'est pas impacté par la solution de déploiement.

4.2.7.1 Configuration des *extras*

Le fichier `environments /group_vars/all/extra_vars.yml` contient la configuration des *extras* :

```
1  ---  
2  
3  vitam:  
4    ihm_recette:  
5      vitam_component: ihm-recette  
6      host: "ihm-recette.service.{{ consul_domain }}"  
7      port_service: 8445  
8      port_admin: 28204  
9      baseurl: /ihm-recette  
10     static_content: "{{ vitam_defaults.folder.root_path }}/app/ihm-recette"  
11     baseuri: "ihm-recette"  
12     secure_mode:  
13       - authc  
14     https_enabled: true  
15     secret_platform: "false"  
16     cluster_name: "{{ elasticsearch.data.cluster_name }}"  
17     session_timeout: 1800000  
18     secure_cookie: true  
19     use_proxy_to_clone_tests: "yes"  
20     metrics_enabled: true
```

(suite sur la page suivante)

(suite de la page précédente)

```

21     logback_rolling_policy: true
22     logback_max_file_size: "10MB"
23     logback_total_size_cap:
24         file:
25             history_days: 10
26             totalsize: "5GB"
27         security:
28             history_days: 10
29             totalsize: "5GB"
30     jvm_log: false
31     performance_logger: "false"
32     reconstruction:
33     consul_check_business: 10 # value in seconds
34     consul_admin_check: 10 # value in seconds
35     acceptableRequestTime: 10 # value in seconds
36     # metricslevel: DEBUG
37     # metricsinterval: 3
38     # metricsunit: MINUTES
39     access_retention_days: 15 # Number of days for file retention
40     access_total_size_cap: "14GB" # total acceptable size
41 library:
42     vitam_component: library
43     host: "library.service.{{ consul_domain }}"
44     port_service: 8090
45     port_admin: 28090
46     baseuri: "doc"
47     https_enabled: false
48     secret_platform: "false"
49     metrics_enabled: false
50     logback_rolling_policy: true
51     logback_max_file_size: "10MB"
52     logback_total_size_cap:
53         file:
54             history_days: 10
55             totalsize: "5GB"
56         security:
57             history_days: 10
58             totalsize: "5GB"
59     jvm_log: false
60     performance_logger: "false"
61     reconstruction:
62     consul_check_business: 30 # value in seconds
63     consul_admin_check: 30 # value in seconds
64     acceptableRequestTime: 10 # value in seconds
65     # metricslevel: DEBUG
66     # metricsinterval: 3
67     # metricsunit: MINUTES
68     access_retention_days: 15 # Number of days for file retention
69     access_total_size_cap: "14GB" # total acceptable size
70
71 tenant_to_clean_before_tnr: ["0", "1"]
72
73 # Period units in seconds
74 metricbeat:
75     system:
76         period: 10
77     mongodb:

```

(suite sur la page suivante)

(suite de la page précédente)

```

78     period: 10
79     elasticsearch:
80         period: 10
81
82 docker_opts:
83     registry_httponly: yes
84     vitam_docker_tag: latest
85
86 gatling_install: false
87 docker_install: true # whether or not install docker & docker images

```

Avertissement : A modifier selon le besoin avant de lancer le playbook ! Les composants ihm-recette et ihm-demo ont la variable `secure_cookie` paramétrée à `true` par défaut, ce qui impose de pouvoir se connecter dessus uniquement en https (même derrière un reverse proxy). Le paramétrage de cette variable se fait dans le fichier `environments/group_vars/all/vitam_vars.yml`

Note : La section `metricbeat` permet de configurer la périodicité d'envoi des informations collectées. Selon l'espace disponible sur le `cluster` Elasticsearch de log et la taille de l'environnement **VITAM** (en particulier, le nombre de machines), il peut être nécessaire d'allonger cette périodicité (en secondes).

Le fichier `environments/group_vars/all/all/vault-extra.yml` contient les secrets supplémentaires des *extras* ; ce fichier est encrypté par `ansible-vault` et doit être paramétré avant le lancement de l'orchestration du déploiement, si le composant ihm-recette est déployé avec récupération des **TNR**.

```

1 # Example for git lfs ; uncomment & use if needed
2 #vitam_gitlab_itest_login: "account"
3 #vitam_gitlab_itest_password: "change_it_4DU42JVf2x2xmPBs"

```

Note : Pour ce fichier, l'encrypter avec le même mot de passe que `vault-vitam.yml`.

4.2.7.2 Déploiement des extras

Plusieurs *playbooks* d'*extras* sont fournis pour usage « tel quel ».

4.2.7.2.1 ihm-recette

Ce *playbook* permet d'installer également le composant **VITAM** ihm-recette.

```

ansible-playbook ansible-vitam-extra/ihm-recette.yml -i environments/<fichier d
↪ 'inventaire> --ask-vault-pass

```

Prudence : Avant de jouer le *playbook*, ne pas oublier, selon le contexte d'usage, de positionner correctement la variable `secure_cookie` décrite plus haut.

4.2.7.2.2 Extras complet

Ce *playbook* permet d'installer :

- des éléments de monitoring système
- un serveur Apache pour naviguer sur le `/vitam` des différentes machines hébergeant *VITAM*
- mongo-express (en docker ; une connexion internet est alors nécessaire)
- le composant *VITAM* library, hébergeant la documentation du projet
- le composant *VITAM* ihm-recette (utilise si configuré des dépôts de jeux de tests)
- un reverse proxy, afin de fournir une page d'accueil pour les environnements de test
- l'outillage de tests de performance

Avertissement : Pour se connecter aux *IHM*, il faut désormais configurer `reverse_proxy_port=443` dans l'inventaire.

```
ansible-playbook ansible-vitam-extra/extra.yml -i environments/<fichier d'inventaire> -  
↪-ask-vault-pass
```

Procédures de mise à jour de la configuration

Cette section décrit globalement les processus de reconfiguration d'une solution logicielle *VITAM* déjà en place et ne peut se substituer aux recommandations effectuées dans la « release-notes » associée à la fourniture des composants mis à niveau.

Se référer également aux *DEX* pour plus de procédures.

5.1 Cas d'une modification du nombre de tenants

Modifier dans le fichier d'inventaire la directive `vitam_tenant_ids`

Exemple :

```
vitam_tenant_ids=[0,1,2]
```

A l'issue, il faut lancer le playbook de déploiement de *VITAM* (et, si déployé, les extras) avec l'option supplémentaire `--tags update_vitam_configuration`.

Exemple :

```
ansible-playbook -i environments/hosts.deployment ansible-vitam/vitam.yml --ask-vault-  
↪pass --tags update_vitam_configuration  
ansible-playbook -i environments/hosts.deployment ansible-vitam-extra/extra.yml --ask-  
↪vault-pass --tags update_vitam_configuration
```

5.2 Cas d'une modification des paramètres JVM

Se référer à *Tuning JVM* (page 51)

Pour les partitions sur lesquelles une modification des paramètres *JVM* est nécessaire, il faut modifier les « hostvars » associées.

A l'issue, il faut lancer le playbook de déploiement de *VITAM* (et, si déployé, les *extras*) avec l'option supplémentaire `--tags update_jvmoptions_vitam`.

Exemple :

```
ansible-playbook -i environments/hosts.deployment ansible-vitam/vitam.yml --ask-vault-  
↪pass --tags update_jvmoptions_vitam  
ansible-playbook -i environments/hosts.deployment ansible-vitam-extra/extra.yml --ask-  
↪vault-pass --tags update_jvmoptions_vitam
```

Prudence : Limitation technique à ce jour ; il n'est pas possible de définir des variables *JVM* différentes pour des composants colocalisés sur une même partition.

5.3 Cas de la mise à jour des *griffins*

Modifier la directive `vitam_griffins` contenue dans le fichier `environments/group_vars/all/vitam-vars.yml`.

Note : Dans le cas d'une montée de version des composant *griffins*, ne pas oublier de mettre à jour l'URL du dépôt de binaire associé.

Relancer le script de déploiement en ajoutant en fin de ligne `--tags griffins` pour ne procéder qu'à l'installation/mise à jour des *griffins*.

6.1 Validation du déploiement

La procédure de validation est commune aux différentes méthodes d'installation.

6.1.1 Sécurisation du fichier `vault_pass.txt`

Le fichier `vault_pass.txt` est très sensible; il contient le mot de passe du fichier `environments/group_vars/all/vault.yml` qui contient les divers mots de passe de la plate-forme. A l'issue de l'installation, il est primordial de le sécuriser (suppression du fichier ou application d'un `chmod 400`).

6.1.2 Validation manuelle

Chaque service *VITAM* (en dehors de bases de données) expose des URL de statut à l'adresse suivante : `<protocole web http ou https>://<host>:<port>/admin/v1/status` Cette URL doit retourner une réponse HTTP 204 sur une requête HTTP GET, si OK.

Un playbook d'appel de l'intégralité des autotests est également inclus (`deployment/ansible-vitam-exploitation/status_vitam.yml`). Il est à lancer de la même manière que pour l'installation de *VITAM* (en renommant le playbook à exécuter).

Il est également possible de vérifier la version installée de chaque composant par l'URL :

`<protocole web http ou https>://<host>:<port>/admin/v1/version`

6.1.3 Validation via Consul

Consul possède une *IHM* pour afficher l'état des services *VITAM* et supervise le « `/admin/v1/status` » de chaque composant *VITAM*, ainsi que des check TCP sur les bases de données.

Pour se connecter à Consul : `http//<Nom du 1er host dans le groupe ansible hosts_consul_server>:8500/ui`

Pour chaque service, la couleur à gauche du composant doit être verte (correspondant à un statut OK).

Si une autre couleur apparaît, cliquer sur le service « KO » et vérifier le test qui ne fonctionne pas.

6.1.4 Post-installation : administration fonctionnelle

À l'issue de l'installation, puis la validation, un **administrateur fonctionnel** doit s'assurer que :

- le référentiel PRONOM ([lien vers pronom¹⁷](#)) est correctement importé depuis « Import du référentiel des formats » et correspond à celui employé dans Siegfried
- le fichier « rules » a été correctement importé via le menu « Import du référentiel des règles de gestion »
- à terme, le registre des fonds a été correctement importé

Les chargements sont effectués depuis l'*IHM* demo.

6.2 Sauvegarde des éléments d'installation

Après installation, il est fortement recommandé de sauvegarder les éléments de configuration de l'installation (i.e. le contenu du répertoire `déploiement/environnements`) ; ces éléments seront à réutiliser pour les mises à jour futures.

Astuce : Une bonne pratique consiste à gérer ces fichiers dans un gestionnaire de version (ex : git)

Prudence : Si vous avez modifié des fichiers internes aux rôles, ils devront également être sauvegardés.

6.3 Troubleshooting

Cette section a pour but de recenser les problèmes déjà rencontrés et y apporter une solution associée.

6.3.1 Erreur au chargement des *index template* kibana

Cette erreur ne se produit qu'en cas de *filesystem* plein sur les partitions hébergeant un cluster elasticsearch. Par sécurité, kibana passe alors ses *index* en `READ ONLY`.

Pour fixer cela, il est d'abord nécessaire de déterminer la cause du *filesystem* plein, puis libérer ou agrandir l'espace disque.

Ensuite, comme indiqué sur [ce fil de discussion¹⁸](#), vous devez désactiver le mode `READ ONLY` dans les *settings* de l'*index .kibana* du cluster elasticsearch.

Exemple :

```
PUT .kibana/_settings
{
  "index": {
    "blocks": {
```

(suite sur la page suivante)

<http://www.nationalarchives.gov.uk/aboutapps/pronom/droid-signature-files.htm>
<https://discuss.elastic.co/t/forbidden-12-index-read-only-allow-delete-api/110282/2>

(suite de la page précédente)

```
        "read_only_allow_delete": "false"
    }
}
}
```

Indication : Il est également possible de lancer cet appel via l'*IHM* du kibana associé, dans l'onglet `Dev Tools`.

À l'issue, vous pouvez relancer l'installation de la solution logicielle *VITAM*.

6.3.2 Erreur au chargement des tableaux de bord Kibana

Dans le cas de machines petitement taillées, il peut arriver que, durant le déploiement, la tâche `Wait for the kibana port to be opened` prenne plus de temps que le *timeout* défini (`vitam_defaults.services.start_timeout`). Pour fixer cela, il suffit de relancer le déploiement.

6.4 Retour d'expérience / cas rencontrés

6.4.1 Crash rsyslog, code killed, signal : BUS

Il a été remarqué chez un partenaire du projet Vitam, que rsyslog se faisait *killer* peu après son démarrage par le signal SIGBUS. Il s'agit très probablement d'un bug rsyslog <= 8.24 <https://github.com/rsyslog/rsyslog/issues/1404>

Pour fixer ce problème, il est possible d'upgrader rsyslog sur une version plus à jour en suivant cette documentation :

- Centos¹⁹
- Debian²⁰

6.4.2 Mongo-express ne se connecte pas à la base de données associée

Si mongoDB a été redémarré, il faut également redémarrer mongo-express.

6.4.3 Elasticsearch possède des shard non alloués (état « UNASSIGNED »)

Lors de la perte d'un noeud d'un cluster elasticsearch, puis du retour de ce noeud, certains shards d'elasticsearch peuvent rester dans l'état UNASSIGNED ; dans ce cas, cerebro affiche les shards correspondant en gris (au-dessus des noeuds) dans la vue « cluster », et l'état du cluster passe en « yellow ». Il est possible d'avoir plus d'informations sur la cause du problème via une requête POST sur l'API `elasticsearch/_cluster/reroute?explain`. Si la cause de l'échec de l'assignation automatique a été résolue, il est possible de relancer les assignations automatiques en échec via une requête POST sur l'API `_cluster/reroute?retry_failed`. Dans le cas où l'assignation automatique ne fonctionne pas, il est nécessaire de faire l'assignation à la main pour chaque shard incriminé (requête POST sur `_cluster/reroute`) :

<https://www.rsyslog.com/rhelcentos-rpms/>
<https://www.rsyslog.com/debian-repository/>

```
{
  "commands": [
    {
      "allocate": {
        "index": "topbeat-2016.11.22",
        "shard": 3,
        "node": "vitam-iaas-dblog-01.int"
      }
    }
  ]
}
```

Cependant, un shard primaire ne peut être réalloué de cette manière (il y a risque de perte de données). Si le défaut d'allocation provient effectivement de la perte puis de la récupération d'un noeud, et que TOUS les noeuds du cluster sont de nouveaux opérationnels et dans le cluster, alors il est possible de forcer la réallocation sans perte.

```
{
  "commands": [
    {
      "allocate": {
        "index": "topbeat-2016.11.22",
        "shard": 3,
        "node": "vitam-iaas-dblog-01.int",
        "allow_primary": "true"
      }
    }
  ]
}
```

Sur tous ces sujets, Cf. la [documentation officielle](#)²¹.

6.4.4 Elasticsearch possède des shards non initialisés (état « **INITIALIZING** »)

Tout d'abord, il peut être difficile d'identifier les shards en questions dans cerebro ; une requête HTTP GET sur l'API `_cat/shards` permet d'avoir une liste plus compréhensible. Un shard non initialisé correspond à un shard en cours de démarrage (Cf. [une ancienne page de documentation](#)²²). Si les shards non initialisés sont présents sur un seul noeud, il peut être utile de redémarrer le noeud en cause. Sinon, une investigation plus poussée doit être menée.

6.4.5 Elasticsearch est dans l'état « **read-only** »

Lorsque Elasticsearch répond par une erreur 403 et que le message suivant est observé dans les logs `ClusterBlockException[blocked by: [FORBIDDEN/xx/index read-only / allow delete (api)]`; cela est probablement consécutif à un remplissage à 100% de l'espace de stockage associé aux index Elasticsearch. Elasticsearch passe alors en lecture seule s'il ne peut plus indexer de documents et garantit ainsi la disponibilité des requêtes en lecture seule uniquement.

Afin de rétablir Elasticsearch dans un état de fonctionnement nominal, il vous faudra alors exécuter la requête suivante :

```
curl -XPUT -H "Content-Type: application/json" http://<es-host>:<es-port>/_all/_
→settings -d '{"index.blocks.read_only_allow_delete": null}'
```

<https://www.elastic.co/guide/en/elasticsearch/reference/current/cluster-reroute.html>
<https://www.elastic.co/guide/en/elasticsearch/reference/1.4/states.html>

6.4.6 MongoDB semble lent

Pour analyser la performance d'un cluster MongoDB, ce dernier fournit quelques outils permettant de faire une première analyse du comportement : [mongostat](#)²³ et [mongotop](#)²⁴.

Dans le cas de VITAM, le cluster MongoDB comporte plusieurs shards. Dans ce cas, l'usage de ces deux commandes peut se faire :

- soit sur le cluster au global (en pointant sur les noeuds mongos) : cela permet d'analyser le comportement global du cluster au niveau de ses points d'entrées ;

```
mongostat --host <ip_service> --port 27017 --username vitamdb-admin --
↳password <password ; défaut : azerty> --authenticationDatabase admin
mongotop --host <ip_service> --port 27017 --username vitamdb-admin --
↳password <password ; défaut : azerty> --authenticationDatabase admin
```

- soit directement sur les noeuds de stockage (mongod) : cela donne des résultats plus fins, et permet notamment de séparer l'analyse sur les noeuds primaires & secondaires d'un même replicaset.

```
mongotop --host <ip_service> --port 27019 --username vitamdb-localadmin --
↳password <password ; défaut : qwerty> --authenticationDatabase admin
mongostat --host <ip_service> --port 27019 --username vitamdb-localadmin --
↳password <password ; défaut : qwerty> --authenticationDatabase admin
```

D'autres outils sont disponibles directement dans le client mongo, notamment pour troubleshoot [les problèmes dûs à la réplication](#)²⁵ :

```
mongo --host <ip_service> --port 27019 --username vitamdb-localadmin --password
↳<password ; défaut : qwerty> --authenticationDatabase admin
> rs.printSlaveReplicationInfo()
> rs.printReplicationInfo()
> db.runCommand( { serverStatus: 1 } )
```

D'autres commandes plus complètes existent et permettent d'avoir plus d'informations, mais leur analyse est plus complexe :

```
# returns a variety of storage statistics for a given collection
> use metadata
> db.stats()
> db.runCommand( { collStats: "Unit" } )
```

Enfin, un outil est disponible en standard afin de mesurer des performances des lecture/écritures avec des patterns proches de ceux utilisés par la base de données ([mongoperf](#)²⁶) :

```
echo "{nThreads:16,fileSizeMB:10000,r:true,w:true}" | mongoperf
```

6.4.7 Les shards de MongoDB semblent mal équilibrés

Normalement, un processus interne à MongoDB (le balancer) s'occupe de déplacer les données entre les shards (par chunk) pour équilibrer la taille de ces derniers. Les commandes suivantes (à exécuter dans un shell mongo sur une instance mongos - attention, ces commandes ne fonctionnent pas directement sur les instances mongod) permettent de s'assurer du bon fonctionnement de ce processus :

<https://docs.mongodb.com/manual/reference/program/mongostat/>
<https://docs.mongodb.com/manual/reference/program/mongotop/>
<https://docs.mongodb.com/manual/tutorial/troubleshoot-replica-sets>
<https://docs.mongodb.com/manual/reference/program/mongoperf/>

- `sh.status()` : donne le status du sharding pour le cluster complet ; c'est un bon premier point d'entrée pour connaître l'état du balancer.
- `use <dbname>`, puis `db.<collection>.getShardDistribution()`, en indiquant le bon nom de base de données (ex : `metadata`) et de collection (ex : `Unit`) : donne les informations de répartition des chunks dans les différents shards pour cette collection.

6.4.8 L'importation initiale (profil de sécurité, certificats) retourne une erreur

Les playbooks d'initialisation importent des éléments d'administration du système (profils de sécurité, certificats) à travers des APIs de la solution VITAM. Cette importation peut être en échec, par exemple à l'étape TASK `[init_contexts_and_security_profiles : Import admin security profile to fonctionnal-admin]`, avec une erreur de type 400. Ce type d'erreur peut avoir plusieurs causes, et survient notamment lors de redéploiements après une première tentative non réussie de déploiement ; même si la cause de l'échec initial est résolue, le système peut se trouver dans un état instable. Dans ce cas, un déploiement complet sur environnement vide est nécessaire pour revenir à un état propre.

Une autre cause possible ici est une incohérence entre l'inventaire, qui décrit notamment les offres de stockage liées aux composants offer, et le paramétrage `vitam_strategy` porté par le fichier `offers_opts.yml`. Si une offre indiquée dans la stratégie n'existe nulle part dans l'inventaire, le déploiement sera en erreur. Dans ce cas, il faut remettre en cohérence ces paramètres et refaire un déploiement complet sur environnement vide.

6.4.9 Problème d'ingest et/ou d'access

Si vous repérez un message de ce type dans les log *VITAM* :

```
fr.gouv.vitam.common.security.filter.AuthorizationWrapper.  
↪checkTimestamp(AuthorizationWrapper.java:117) : [vitam-env-int8-app-04.vitam-  
↪env:storage:239079175] Timestamp check failed
```

Il faut vérifier / corriger l'heure des machines hébergeant la solution logicielle *VITAM* ; un *delta* de temps supérieur à 10s a été détecté entre les machines.

CHAPITRE 7

Montée de version

Pour toute montée de version applicative de la solution logicielle *VITAM*, se référer au *DMV*.

8.1 Vue d'ensemble de la gestion des certificats

8.1.1 Liste des suites cryptographiques & protocoles supportés par VITAM

Il est possible de consulter les *ciphers* supportés par la solution logicielle *VITAM* dans deux fichiers disponibles sur ce chemin : *ansible-vitam/roles/vitam/templates/*

- **Le fichier `jetty-config.xml.j2`**
 - La balise contenant l'attribut `name= »IncludeCipherSuites »` référence les ciphers supportés
 - La balise contenant l'attribut `name= »ExcludeCipherSuites »` référence les ciphers non supportés
- **Le fichier `java.security.j2`**
 - La ligne `jdk.tls.disabledAlgorithms` renseigne les *ciphers* désactivés au niveau java

Avertissement : Les 2 balises concernant les *ciphers* sur le fichier `jetty-config.xml.j2` sont complémentaires car elles comportent des *wildcards* (*) ; en cas de conflit, l'exclusion est prioritaire.

Voir aussi :

Ces fichiers correspondent à la configuration recommandée ; celle-ci est décrite plus en détail dans le *DAT* (chapitre sécurité).

8.1.2 Vue d'ensemble de la gestion des certificats

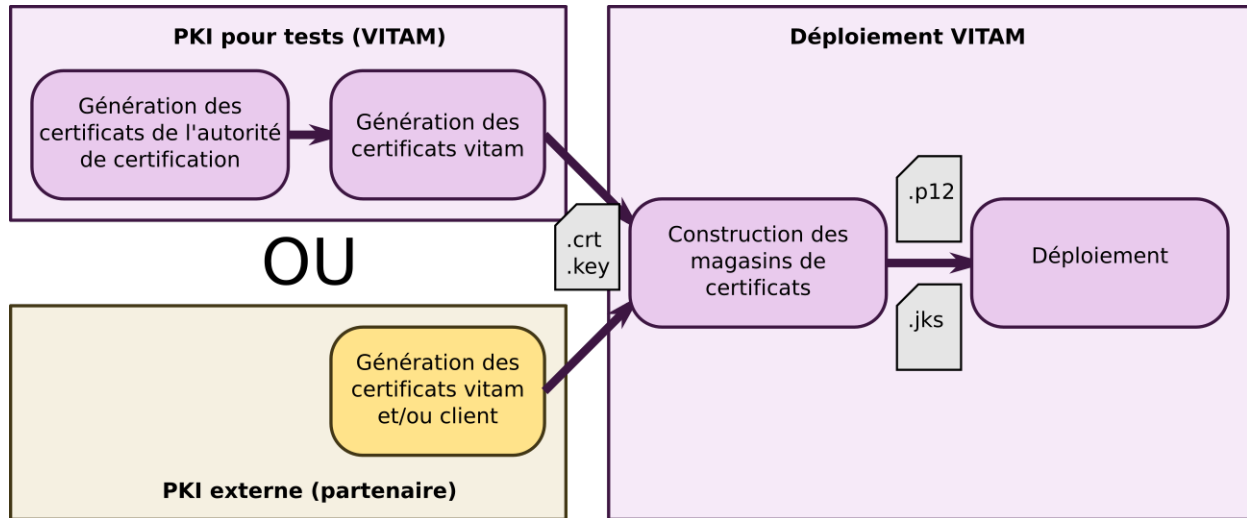
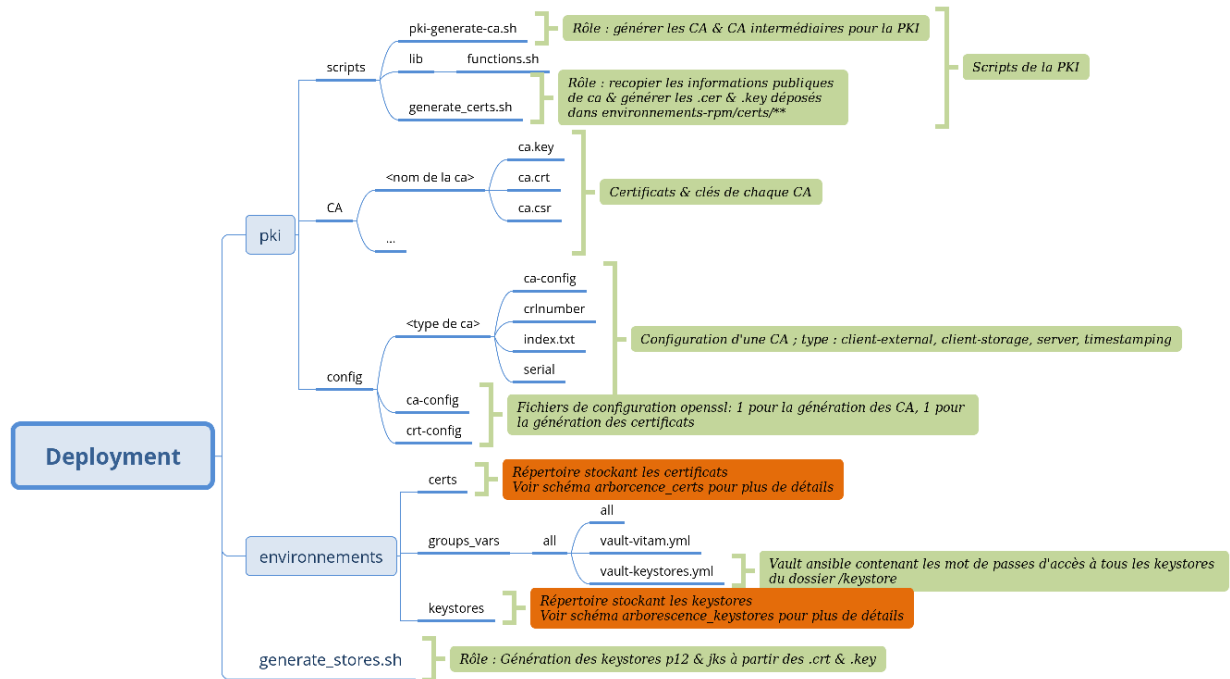


Fig. 1 – Vue d'ensemble de la gestion des certificats au déploiement

8.1.3 Description de l'arborescence de la PKI

Tous les fichiers de gestion de la *PKI* se trouvent dans le répertoire `deployment` de l'arborescence *VITAM* :

- Le sous répertoire `pki` contient les scripts de génération des *CA* & des certificats, les *CA* générées par les scripts, et les fichiers de configuration d'`openssl`
- Le sous répertoire `environments` contient tous les certificats nécessaires au bon déploiement de *VITAM* :
 - certificats publics des *CA*
 - certificats clients, serveurs, de timestamping, et coffre fort contenant les mots de passe des clés privées des certificats (sous-répertoire `certs`)
 - magasins de certificats (`keystores` / `truststores` / `grantedstores`), et coffre fort contenant les mots de passe des magasins de certificats (sous-répertoire `keystores`)
- Le script `generate_stores.sh` génère les magasins de certificats (`keystores`), cf la section *Fonctionnement des scripts de la PKI* (page 101)


Fig. 2 – Vue l'arborescence de la *PKI* Vitam

8.1.4 Description de l'arborescence du répertoire deployment/environments/certs

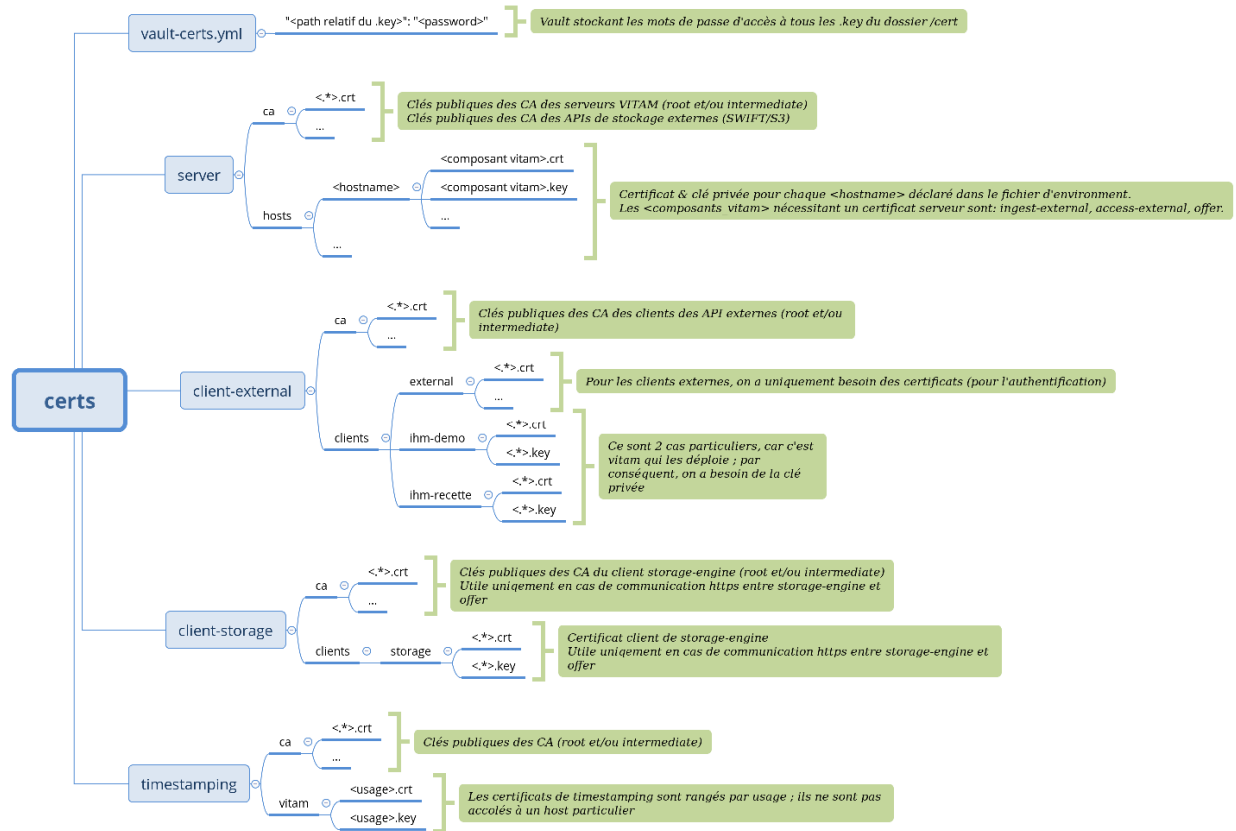


Fig. 3 – Vue détaillée de l'arborescence des certificats

8.1.5 Description de l'arborescence du répertoire deployment/environments/keystores

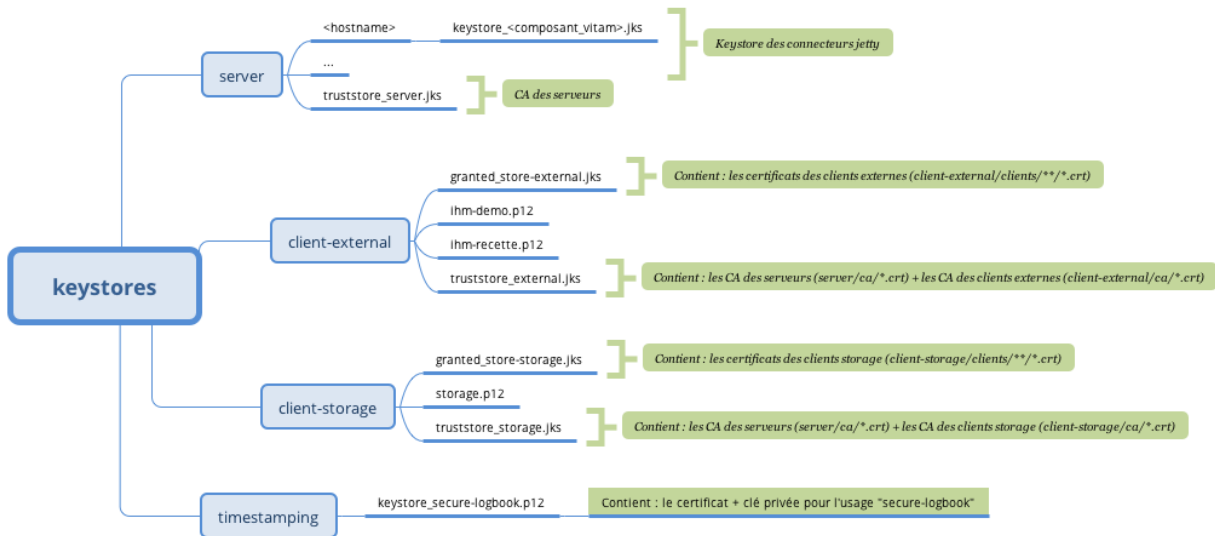


Fig. 4 – Vue détaillée de l'arborescence des keystores

8.1.6 Fonctionnement des scripts de la PKI

La gestion de la *PKI* se fait avec 3 scripts situés dans le répertoire deployment de l'arborescence *VITAM* :

- `pki/scripts/generate_ca.sh` : génère des autorités de certifications (si besoin)
- `pki/scripts/generate_certs.sh` : génère des certificats à partir des autorités de certifications présentes (si besoin)
 - Récupère le mot de passe des clés privées à générer dans le vault `environments/certs/vault-certs.yml`
 - Génère les certificats & les clés privées
- `generate_stores.sh` : génère les magasins de certificats nécessaires au bon fonctionnement de *VITAM*
 - Récupère le mot de passe du magasin indiqué dans `environments/group_vars/all/vault-keystore.yml`
 - Insère les bons certificats dans les magasins qui en ont besoin

Si les certificats sont créés par la *PKI* externe, il faut les positionner dans l'arborescence attendue avec le nom attendu pour certains (cf *l'image ci-dessus* (page 100)).

8.2 Spécificités des certificats

Trois différents types de certificats sont nécessaires et utilisés dans *VITAM* :

- Certificats serveur
- Certificats client
- Certificats d'horodatage

Pour générer des certificats, il est possible de s'inspirer du fichier `pki/config/crt-config`. Il s'agit du fichier de configuration openssl utilisé par la *PKI* de test de *VITAM*. Ce fichier dispose des 3 modes de configurations nécessaires pour générer les certificats de *VITAM* :

- `extension_server` : pour générer les certificats serveur
- `extension_client` : pour générer les certificats client
- `extension_timestamping` : pour générer les certificats d'horodatage

8.2.1 Cas des certificats serveur

8.2.1.1 Généralités

Les services *VITAM* qui peuvent utiliser des certificats serveur sont : `ingest-external`, `access-external`, `offer` (les seuls pouvant écouter en https). Par défaut, `offer` n'écoute pas en https par soucis de performances.

Pour les certificats serveur, il est nécessaire de bien réfléchir au *CN* et *subjectAltName* qui vont être spécifiés. Si par exemple le composant `offer` est paramétré pour fonctionner en https uniquement, il faudra que le *CN* ou un des *subjectAltName* de son certificat corresponde à son nom de service sur consul.

8.2.1.2 Noms DNS des serveurs https VITAM

Les noms *DNS* résolus par *Consul* seront ceux ci :

- `<nom_service>.service.<domaine_consul>` sur le datacenter local
- `<nom_service>.service.<dc_consul>.<domaine_consul>` sur n'importe quel datacenter

Rajouter le nom « Consul » avec le nom du datacenter dedans peut par exemple servir si une installation multi-site de *VITAM* est faite (appels storage -> `offer` inter *DC*)

Les variables pouvant impacter les noms d'hosts *DNS* sur *Consul* sont :

- `consul_domain` dans le fichier `environments/group_vars/all/vitam_vars.yml` -> `<domain_consul>`
- `vitam_site_name` dans le fichier d'inventaire `environments/hosts` (variable globale) -> `<dc_consul>`
- Service `offer` seulement : `offer_conf` dans le fichier d'inventaire `environments/hosts` (différente pour chaque instance du composant `offer`) -> `<nom_service>`

Exemples :

Avec `consul_domain: consul`, `vitam_site_name: dc2`, l'offre `offer-fs-1` sera résolue par

- `offer-fs-1.service.consul` depuis le `dc2`
- `offer-fs-1.service.dc2.consul` depuis n'importe quel *DC*

Avec `consul_domain: preprod.vitam`, `vitam_site_name: dc1`, les composants `ingest-external` et `access-external` seront résolu par

- `ingest-external.service.preprod.vitam` et `access-external.service.preprod.vitam` depuis le *DC* local
- `ingest-external.service.dc1.preprod.vitam` et `access-external.service.dc1.preprod.vitam` depuis n'importe quel *DC*

Avvertissement : Si les composants `ingest-external` et `access-external` sont appelés via leur *IP* ou des records *DNS* autres que ceux de *Consul*, il faut également ne pas oublier de les rajouter dans les *subjectAltName*.

8.2.2 Cas des certificats client

Les services qui peuvent utiliser des certificats client sont :

- N'importe quelle application utilisant les !term :API *VITAM* exposées sur ingest-external et access-external
- Le service storage si le service offer est configuré en https
- **Un certificat client nommé vitam-admin-int est obligatoire**
 - Pour déployer *VITAM* (nécessaire pour initialisation du fichier pronom)
 - Pour lancer certains actes d'exploitation

8.2.3 Cas des certificats d'horodatage

Les services logbook et storage utilisent des certificats d'horodatage.

8.2.4 Cas des certificats des services de stockage objets

En cas d'utilisation d'offres de stockage objet avec *VITAM*, si une connexion https est utilisée, il est nécessaire de déposer les *CA* (root et/ou intermédiaire) des serveurs de ces offres de stockage dans le répertoire deployment/environments/certs/server/ca. Cela permettra d'ajouter ces *CA* dans le **truststore** du serveur offer lorsque les **keystores** seront générés.

8.3 Cycle de vie des certificats

Le tableau ci-dessous indique le mode de fonctionnement actuel pour les différents certificats et *CA*. Précisions :

- Les « procédures par défaut » liées au cycle de vie des certificats dans la présente version de la solution *VITAM* peuvent être résumées ainsi :
 - Création : génération par *PKI* partenaire + copie dans répertoires de déploiement + script generate_stores.sh + déploiement ansible
 - Suppression : suppression dans répertoires de déploiement + script generate_stores.sh + déploiement ansible
 - Renouvellement : régénération par *PKI* partenaire + suppression / remplacement dans répertoires de déploiement + script generate_stores.sh + redéploiement ansible
- Il n'y a pas de contrainte au niveau des *CA* utilisées (une *CA* unique pour tous les usages *VITAM* ou plusieurs *CA* séparées – cf. *DAT*). On appelle ici :
 - « *PKI* partenaire » : *PKI* / *CA* utilisées pour le déploiement et l'exploitation de la solution *VITAM* par le partenaire.
 - « *PKI* distante » : *PKI* / *CA* utilisées pour l'usage des frontaux en communication avec le back office *VITAM*.

Classe	Type	Usages	Origine	Création	Suppression	Renouvellement
Interne	CA	ingest & access	PKI partenaire	proc. par défaut	proc. par défaut	proc. par défaut
Interne	CA	offer	PKI partenaire	proc. par défaut	proc. par défaut	proc. par défaut
Interne	Certif	Horodatage	PKI partenaire	proc. par défaut	proc. par défaut	proc. par défaut
Interne	Certif	Storage (Swift)	Offre de stockage	proc. par défaut	proc. par défaut	proc. par défaut
Interne	Certif	Storage (s3)	Offre de stockage	proc. par défaut	proc. par défaut	proc. par défaut
Interne	Certif	ingest	PKI partenaire	proc. par défaut	proc. par défaut	proc. par défaut
Interne	Certif	access	PKI partenaire	proc. par défaut	proc. par défaut	proc. par défaut
Interne	Certif	offer	PKI partenaire	proc. par défaut	proc. par défaut	proc. par défaut
Interne	Certif	Timestamp	PKI partenaire	proc. par défaut	proc. par défaut	proc. par défaut
IHM demo	CA	ihm-demo	PKI partenaire	proc. par défaut	proc. par défaut	proc. par défaut
IHM demo	Certif	ihm-demo	PKI partenaire	proc. par défaut	proc. par défaut	proc. par défaut
SIA	CA	Appel API	PKI distante	proc. par défaut (PKI distante)	proc. par défaut	proc. par défaut (PKI distante)+recharger Certifs
SIA	Certif	Appel API	PKI distante	Génération + copie répertoire + deploy(par la suite appel API d'insertion)	Suppression Mongo	Suppression Mongo + API d'insertion
Personae	Certif	Appel API	PKI distante	API ajout	API suppression	API suppression + API ajout

Remarques :

- Lors d'un renouvellement de CA SIA, il faut s'assurer que les certificats qui y correspondaient soient retirés de MongoDB et que les nouveaux certificats soient ajoutés par le biais de l'API dédiée.
- Lors de toute suppression ou remplacement de certificats SIA, s'assurer que la suppression ou remplacement des contextes associés soit également réalisé.
- L'expiration des certificats n'est pas automatiquement prise en charge par la solution VITAM (pas de notification en fin de vie, pas de renouvellement automatique). Pour la plupart des usages, un certificat expiré est proprement rejeté et la connexion ne se fera pas ; les seules exceptions sont les certificats Personae, pour lesquels la validation de l'arborescence CA et des dates est à charge du front office en interface avec VITAM.

8.4 Ansible & SSH

En fonction de la méthode d'authentification sur les serveurs et d'élévation de privilège, il faut rajouter des options aux lignes de commande ansible. Ces options seront à rajouter pour toutes les commandes ansible du document .

Pour chacune des 3 sections suivantes, vous devez être dans l'un des cas décrits

8.4.1 Authentification du compte utilisateur utilisé pour la connexion SSH

Pour le login du compte utilisateur, voir la section *Informations plate-forme* (page 22).

8.4.1.1 Par clé SSH avec passphrase

Dans le cas d'une authentification par clé avec passphrase, il est nécessaire d'utiliser ssh-agent pour mémoriser la clé privée. Pour ce faire, il faut :

- exécuter la commande `ssh-agent <shell utilisé>` (exemple `ssh-agent /bin/bash`) pour lancer un shell avec un agent de mémorisation de la clé privée associé à ce shell
- exécuter la commande `ssh-add` et renseigner la passphrase de la clé privée

Vous pouvez maintenant lancer les commandes ansible comme décrites dans ce document.

A noter : ssh-agent est un démon qui va stocker les clés privées (déchiffrées) en mémoire et que le client *SSH* va interroger pour récupérer les informations privées pour initier la connexion. La liaison se fait par un socket UNIX présent dans /tmp (avec les droits 600 pour l'utilisateur qui a lancé le ssh-agent). Cet agent disparaît avec le shell qui l'a lancé.

8.4.1.2 Par login/mot de passe

Dans le cas d'une authentification par login/mot de passe, il est nécessaire de spécifier l'option `--ask-pass` (ou `-k` en raccourci) aux commandes ansible ou ansible-playbook de ce document .

Au lancement de la commande ansible (ou ansible-playbook), il sera demandé le mot de passe

8.4.1.3 Par clé SSH sans passphrase

Dans ce cas, il n'y a pas de paramétrage particulier à effectuer.

8.4.2 Authentification des hôtes

Pour éviter les attaques de type *MitM*, le client *SSH* cherche à authentifier le serveur sur lequel il se connecte. Ceci se base généralement sur le stockage des clés publiques des serveurs auxquels il faut faire confiance (`~/.ssh/known_hosts`).

Il existe différentes méthodes pour remplir ce fichier (vérification humaine à la première connexion, gestion centralisée, *DNSSEC*). La gestion de fichier est hors périmètre *VITAM* mais c'est un pré-requis pour le lancement d'ansible.

8.4.3 Elévation de privilèges

Une fois que l'on est connecté sur le serveur cible, il faut définir la méthode pour accéder aux droits `root`

8.4.3.1 Par sudo avec mot de passe

Dans ce cas, il faut rajouter les options `--ask-sudo-pass`

Au lancement de la commande `ansible` (ou `ansible-playbook`), il sera demandé le mot de passe demandé par `sudo`

8.4.3.2 Par su

Dans ce cas, il faut rajouter les options `--become-method=su --ask-su-pass`

Au lancement de la commande `ansible` (ou `ansible-playbook`), il sera demandé le mot de passe `root`

8.4.3.3 Par sudo sans mot de passe

Il n'y a pas d'option à rajouter (l'élévation par `sudo` est la configuration par défaut)

8.4.3.4 Déjà Root

Dans ce cas, il n'y a pas de paramétrages supplémentaires à effectuer.

Table des figures

1	Cinématique de déploiement	14
2	Vue détaillée des certificats entre le storage et l'offre en multi-site	20
3	Vue détaillée de l'arborescence des certificats	49
1	Vue d'ensemble de la gestion des certificats au déploiement	98
2	Vue l'arborescence de la <i>PKI</i> Vitam	99
3	Vue détaillée de l'arborescence des certificats	100
4	Vue détaillée de l'arborescence des keystores	101

Liste des tableaux

1	Documents de référence VITAM	2
1	Description des identifiants de référentiels	55
2	Description des règles	56

A

API, 3
AU, 3

B

BDD, 3
BDO, 3

C

CA, 3
CAS, 3
CCFN, 3
CN, 3
COTS, 3
CRL, 3
CRUD, 3

D

DAT, 3
DC, 3
DEX, 3
DIN, 3
DIP, 3
DMV, 3
DNS, 3
DNSSEC, 3
DSL, 3
DUA, 3

E

EAD, 3
EBIOS, 3
ELK, 3

F

FIP, 3

G

GOT, 3

I

IHM, 3
IP, 3
IsaDG, 3

J

JRE, 3
JVM, 4

L

LAN, 4
LFC, 4
LTS, 4

M

M2M, 4
MitM, 4
MoReq, 4

N

NoSQL, 4
NTP, 4

O

OAIS, 4
OOM, 4
OS, 4
OWASP, 4

P

PCA, 4
PDMA, 4
PKI, 4
PRA, 4

R

REST, 4
RGAA, 4
RGI, 4

RPM, [4](#)

S

SAE, [4](#)

SEDA, [4](#)

SGBD, [5](#)

SGBDR, [5](#)

SIA, [5](#)

SIEM, [5](#)

SIP, [5](#)

SSH, [5](#)

Swift, [5](#)

T

TLS, [5](#)

TNR, [5](#)

TTL, [5](#)

U

UDP, [5](#)

UID, [5](#)

V

VITAM, [5](#)

VM, [5](#)

W

WAF, [5](#)

WAN, [5](#)