



VITAM - Documentation d'installation

Version 7.0.1

VITAM

déc. 12, 2023

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	8
3.2.1	Base commune	8
3.2.2	PKI	9
3.2.3	Systèmes d’exploitation	9
3.2.3.1	Déploiement sur environnement CentOS	10
3.2.3.2	Déploiement sur environnement Debian	10
3.2.3.3	Présence d’un agent antivirus	10
3.2.4	Matériel	11
3.2.5	Librairie de cartouches pour offre froide	11
3.3	Questions préparatoires	11
3.4	Récupération de la version	12
3.4.1	Utilisation des dépôts <i>open-source</i>	12
3.4.1.1	<i>Repository</i> pour environnement CentOS	12
3.4.1.1.1	Cas de <i>griffins</i>	12
3.4.1.2	<i>Repository</i> pour environnement Debian	13
3.4.1.2.1	Cas de <i>griffins</i>	13
3.4.2	Utilisation du package global d’installation	13
4	Procédures d’installation / mise à jour	14
4.1	Vérifications préalables	14
4.2	Procédures	14
4.2.1	Cinématique de déploiement	14
4.2.2	Cas particulier d’une installation multi-sites	15
4.2.2.1	Procédure d’installation	15
4.2.2.1.1	<code>vitam_site_name</code>	15

4.2.2.1.2	primary_site	15
4.2.2.1.3	consul_remote_sites	16
4.2.2.1.4	vitam_offers	16
4.2.2.1.5	vitam_strategy	17
4.2.2.1.6	other_strategies	18
4.2.2.1.7	plateforme_secret	19
4.2.2.1.8	consul_encrypt	20
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	22
4.2.3	Configuration du déploiement	22
4.2.3.1	Fichiers de déploiement	22
4.2.3.2	Informations <i>plate-forme</i>	22
4.2.3.2.1	Inventaire	22
4.2.3.2.2	Fichier main.yml	32
4.2.3.2.3	Fichier vitam_security.yml	35
4.2.3.2.4	Fichier offers_opts.yml	36
4.2.3.2.5	Fichier cots_vars.yml	41
4.2.3.2.6	Fichier tenants_vars.yml	48
4.2.3.3	Déclaration des secrets	52
4.2.3.3.1	vitam	52
4.2.3.3.2	Cas des extras	56
4.2.3.3.3	Commande ansible-vault	57
4.2.3.3.3.1	Générer des fichiers <i>vaultés</i> depuis des fichier en clair	57
4.2.3.3.3.2	Re-chiffrer un fichier <i>vaulté</i> avec un nouveau mot de passe	57
4.2.3.4	Le mapping Elasticsearch pour Unit et ObjectGroup	57
4.2.4	Gestion des certificats	64
4.2.4.1	Cas 1 : Configuration développement / tests	64
4.2.4.1.1	Procédure générale	64
4.2.4.1.2	Génération des CA par les scripts Vitam	64
4.2.4.1.3	Génération des certificats par les scripts Vitam	64
4.2.4.2	Cas 2 : Configuration production	65
4.2.4.2.1	Procédure générale	65
4.2.4.2.2	Génération des certificats	65
4.2.4.2.2.1	Certificats serveurs	65
4.2.4.2.2.2	Certificat clients	66
4.2.4.2.2.3	Certificats d'horodatage	66
4.2.4.2.3	Intégration de certificats existants	66
4.2.4.2.4	Intégration de certificats clients de VITAM	68
4.2.4.2.4.1	Intégration d'une application externe (cliente)	68
4.2.4.2.4.2	Intégration d'un certificat personnel (<i>personae</i>)	68
4.2.4.2.5	Cas des offres objet	68
4.2.4.2.6	Absence d'usage d'un <i>reverse</i>	68
4.2.4.3	Intégration de CA pour une offre <i>Swift</i> ou <i>s3</i>	68
4.2.4.4	Génération des magasins de certificats	69
4.2.5	Paramétrages supplémentaires	69
4.2.5.1	<i>Tuning</i> JVM	69
4.2.5.2	Installation des <i>griffins</i> (greffons de préservation)	69
4.2.5.3	Rétention liée aux logback	70
4.2.5.3.1	Cas des accesslog	70
4.2.5.4	Paramétrage de l'antivirus (ingest-external)	70
4.2.5.4.1	Extra : Avast Business Antivirus for Linux	71
4.2.5.5	Paramétrage des certificats externes (*-externe)	72

4.2.5.6	Placer « hors Vitam » le composant ihm-demo	72
4.2.5.7	Paramétrer le <code>secure_cookie</code> pour ihm-demo	73
4.2.5.8	Paramétrage de la centralisation des logs VITAM	73
4.2.5.8.1	Gestion par VITAM	73
4.2.5.8.2	Redirection des logs sur un SIEM tiers	73
4.2.5.9	Passage des identifiants des référentiels en mode <i>esclave</i>	74
4.2.5.10	Paramétrage du batch de calcul pour l'indexation des règles héritées	74
4.2.5.11	Durées minimales permettant de contrôler les valeurs saisies	75
4.2.5.12	Augmenter la précision sur le nombre de résultats retournés dépassant 10000	75
4.2.5.13	Fichiers complémentaires	76
4.2.5.14	Paramétrage de l'Offre Froide (librairies de cartouches)	99
4.2.5.15	Sécurisation SELinux	103
4.2.5.16	Installation de la stack Prometheus	104
4.2.5.16.1	Playbooks ansible	105
4.2.5.17	Installation de Grafana	105
4.2.5.17.1	Configuration	105
4.2.5.17.2	Configuration spécifique derrière un proxy	106
4.2.5.18	Installation de restic	106
4.2.5.18.1	Configuration	106
4.2.5.18.2	Limitations actuelles	106
4.2.6	Procédure de première installation	107
4.2.6.1	Déploiement	107
4.2.6.1.1	Cas particulier : utilisation de ClamAv en environnement Debian	107
4.2.6.1.2	Fichier de mot de passe des vaults ansible	107
4.2.6.1.3	Mise en place des repositories VITAM (optionnel)	107
4.2.6.1.4	Génération des <i>hostvars</i>	108
4.2.6.1.4.1	Cas 1 : Machines avec une seule interface réseau	108
4.2.6.1.4.2	Cas 2 : Machines avec plusieurs interfaces réseau	108
4.2.6.1.4.3	Vérification de la génération des <i>hostvars</i>	108
4.2.6.1.5	Tests d'infrastructure	109
4.2.6.1.6	Déploiement	109
4.2.7	Éléments <i>extras</i> de l'installation	110
4.2.7.1	Configuration des <i>extras</i>	110
4.2.7.2	Déploiement des <i>extras</i>	111
4.2.7.2.1	ihm-recette	112
4.2.7.2.2	<i>Extras</i> complet	112
5	Procédures de mise à jour de la configuration	113
5.1	Cas d'une modification du nombre de tenants	113
5.2	Cas d'une modification des paramètres JVM	114
5.3	Cas de la mise à jour des <i>griffins</i>	114
6	Post installation	115
6.1	Validation du déploiement	115
6.1.1	Sécurisation du fichier <code>vault_pass.txt</code>	115
6.1.2	Validation manuelle	115
6.1.3	Validation via Consul	115
6.1.4	Post-installation : administration fonctionnelle	116
6.2	Sauvegarde des éléments d'installation	116
6.3	Troubleshooting	116
6.3.1	Erreur au chargement des <i>index template</i> kibana	116
6.3.2	Erreur au chargement des tableaux de bord Kibana	117
6.4	Retour d'expérience / cas rencontrés	117
6.4.1	Crash rsyslog, code killed, signal : BUS	117

6.4.2	Mongo-express ne se connecte pas à la base de données associée	117
6.4.3	Elasticsearch possède des shard non alloués (état « UNASSIGNED »)	117
6.4.4	Elasticsearch possède des shards non initialisés (état « INITIALIZING »)	118
6.4.5	Elasticsearch est dans l'état « <i>read-only</i> »	118
6.4.6	MongoDB semble lent	119
6.4.7	Les shards de MongoDB semblent mal équilibrés	119
6.4.8	L'importation initiale (profil de sécurité, certificats) retourne une erreur	120
6.4.9	Problème d'ingest et/ou d'access	120
7	Montée de version	121
8	Annexes	122
8.1	Vue d'ensemble de la gestion des certificats	122
8.1.1	Liste des suites cryptographiques & protocoles supportés par VITAM	122
8.1.2	Vue d'ensemble de la gestion des certificats	123
8.1.3	Description de l'arborescence de la PKI	123
8.1.4	Description de l'arborescence du répertoire <code>deployment/environments/certs</code>	125
8.1.5	Description de l'arborescence du répertoire <code>deployment/environments/keystores</code>	126
8.1.6	Fonctionnement des scripts de la PKI	126
8.2	Spécificités des certificats	126
8.2.1	Cas des certificats serveur	127
8.2.1.1	Généralités	127
8.2.1.2	Noms DNS des serveurs https VITAM	127
8.2.2	Cas des certificats client	128
8.2.3	Cas des certificats d'horodatage	128
8.2.4	Cas des certificats des services de stockage objets	128
8.3	Cycle de vie des certificats	128
8.4	Ansible & SSH	130
8.4.1	Authentification du compte utilisateur utilisé pour la connexion SSH	130
8.4.1.1	Par clé SSH avec passphrase	130
8.4.1.2	Par login/mot de passe	130
8.4.1.3	Par clé SSH sans passphrase	130
8.4.2	Authentification des hôtes	130
8.4.3	Élévation de privilèges	130
8.4.3.1	Par sudo avec mot de passe	131
8.4.3.2	Par su	131
8.4.3.3	Par sudo sans mot de passe	131
8.4.3.4	Déjà Root	131
	Index	134

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*.

2.1 Information concernant les licences

La solution logicielle *VITAM* est publiée sous la licence CeCILL 2.1¹ ; la documentation associée (comprenant le présent document) est publiée sous Licence Ouverte V2.0².

Les clients externes java de solution *VITAM* sont publiés sous la licence CeCILL-C³ ; la documentation associée (comprenant le présent document) est publiée sous Licence Ouverte V2.0⁴.

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

1. https://cecill.info/licences/Licence_CeCILL_V2.1-fr.html
2. <https://www.etalab.gouv.fr/wp-content/uploads/2017/04/ETALAB-Licence-Ouverte-v2.0.pdf>
3. https://cecill.info/licences/Licence_CeCILL-C_V1-fr.html
4. <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 Suite logicielle *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.

5. 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 paquets logiciels nativement utilisé par les distributions Linux RedHat/CentOS (entre autres)

SAE Système d'Archivage Électronique

SEDA Standard d'Échange de Données pour l'Archivage

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

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

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

9. 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*

TNA *The National Archives, Pronom*¹⁰

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

10. <https://www.nationalarchives.gov.uk/PRONOM/>

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 :

TABLEAU 1 – Matrice de compétences

Thème	Outil	Description de l'outil	Niveau requis	Niveau de criticité	Exemples de compétences requises
Système	Linux (Centos 7 ou Debian 10)	Système d'exploitation	3/4 : maîtrise	3/4 : Majeur	Etre à l'aise avec l'arborescence linux / Configurer une interface réseau / Analyse avancée des logs systèmes et réseaux
Configuration	Git	Suivi des modifications quotidiennes des sources de déploiement VITAM	1/4 : débutant	1/4 : Mineur	Savoir exécuter les commandes de bases (commit, pull, push, etc...)
Configuration	Git	Adaptation des sources de déploiement VITAM dans le cadre d'une montée de version	2/4 : intermédiaire	1/4 : Mineur	Savoir exécuter les commandes intermédiaires (branche, merge, etc...)
Configuration	Ansible	Gestion de configuration et déploiement automatisé	3/4 : maîtrise	3/4 : Majeur	Adapter les paramètres pour permettre une installation spécifique / Comprendre l'arborescence des rôles et des playbooks
Exploitation	Consul	Outil d'enregistrement des services VITAM	1/4 : débutant	4/4 : critique	Contrôler l'état des services via l'interface consul Eteindre et redémarrer un Consul Agent sur une machine virtuelle
Supervision	Kibana	Interface de visualisation du contenu des bases Elasticsearch	1/4 : débutant	2/4 : significatif	Créer un nouveau dashboard avec des indicateurs spécifiques / Lire et relever les données pertinentes dans un dashboard donné
Supervision	Cerebro	Interface de contrôle des clusters Elasticsearch	1/4 : débutant	2/4 : significatif	Contrôler l'état des clusters elasticsearch via l'interface cerebro
Base de données	MongoDB	Base de données NoSQL	2/4 : intermédiaire	4/4 : critique	Effectuer une recherche au sein d'une base mongoDB / Sauvegarder et restaurer une base mongoDB (data ou offer) / Augmenter la capacité de stockage d'une base mongoDB
Base de données	Elasticsearch	Moteur de recherche et d'indexation de données distribué	2/4 : intermédiaire	4/4 : critique	Sauvegarder et restaurer une base elasticsearch (data ou log) / Augmenter la capacité de stockage d'une base elasticsearch / Effectuer une procédure de maintenance d'un nœud au sein d'un cluster elasticsearch
Appliquatif	Applicatifs Java	Composants logiciels Vitam	2/4 : intermédiaire	4/4 : critique	Appeler le point "v1/status" manuellement sur tous les composants VITAM / Arrêter et relancer selectivement les composants VITAM à l'aide d'Ansible (ordre important) / Lancer une procédure d'indisponibilité de VITAM (fermeture des services external, arrêt des timers)
3.1. Expertises requises	Solution	Administration	2/4 : intermédiaire	4/4 : critique	pouvoir lister les containers/buckets et objets, vérifier la

- Niveau requis : Qualifie le niveau de compétence attendue par l'exploitant de la solution logicielle Vitam.
- Niveau de criticité : Qualifie le degré d'importance pour le bon fonctionnement de la plateforme.

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](#)¹¹ pour la procédure d'installation)
 - **openssh-client** (client SSH utilisé par ansible)
 - **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](#)¹² 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 : Aucune version pré-installée de la JRE OpenJDK ne doit être présente sur les machines cibles où sera installé *VITAM*.

Prudence : La solution *VITAM* ne tolère qu'une très courte désynchronisation de temps entre les machines (par défaut, 10 secondes). La configuration NTP doit être finement monitorée. Idéalement une synchronisation doit être planifiée chaque 5/10 minutes.

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

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

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 *prerequisoffrefroide*.

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 122) 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 :

Veuillez vous référer à la section *Vue d'ensemble de la gestion des certificats* (page 122) 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 antiviral

Dans le cas de partitions sur lesquelles un agent antiviral 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 antiviral en prérequis des *ingest*; se reporter à *Paramétrage de l'antivirus (ingest-external)* (page 70)).

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 et mongodb 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 configurés sur la machine devant supporter une instance de ce composant. La commande `ls -l /dev/scsi` peut permettre de vérifier si des périphériques sont détectés.
- Le dossier `/vitam/data/offer/` doit correspondre à une seule partition de système de fichiers (i.e. tout le contenu du dossier `/vitam/data/offer` doit appartenir au même point de montage). Le système de fichiers doit supporter les opérations atomiques (type atomic rename / move) et la création de liens symboliques (ex. XFS, EXT4...)

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 tenant 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 tenant 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*¹³, 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.

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
```

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

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 :

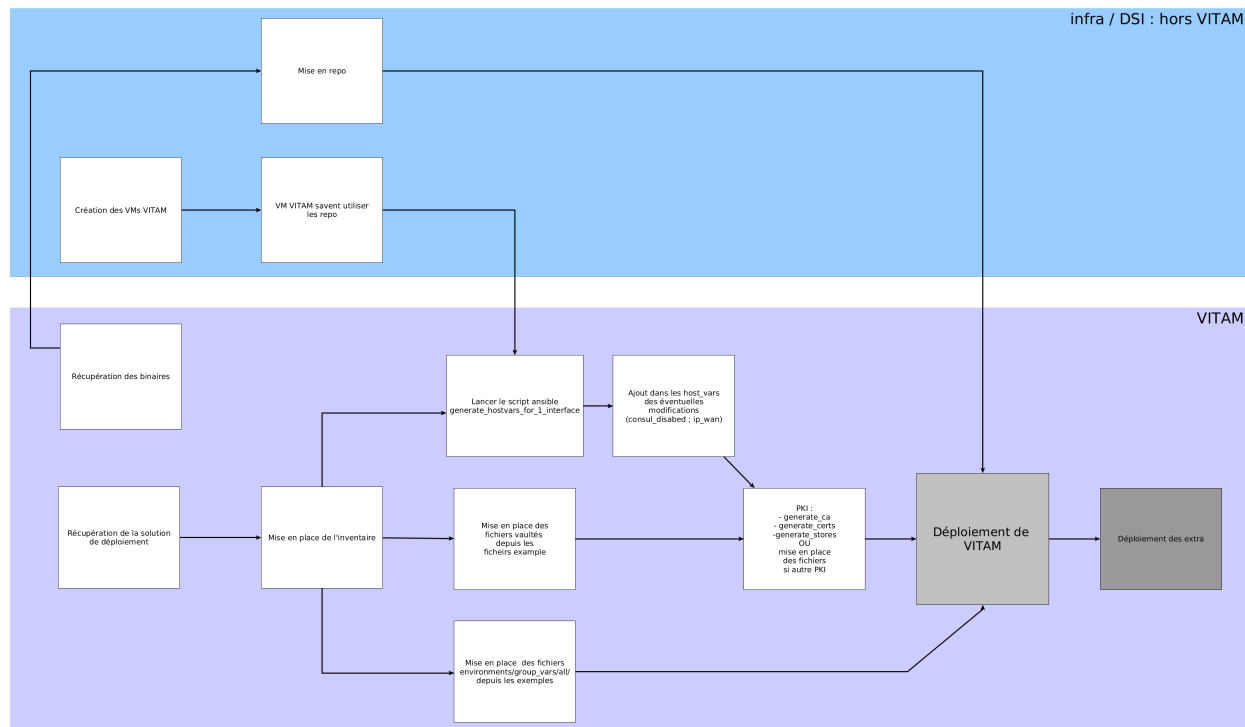


FIG. 1 – Cinématique de déploiement

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.<environnement>`

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.<environnement>`

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/main/main.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. Sur les sites secondaires, il suffit de référencer les offres disponible localement.

Exemple :

```
vitam_offers:
  offer-fs-1:
    provider: filesystem-hash
  offer-fs-2:
    provider: filesystem-hash
```

(suite sur la page suivante)

(suite de la page précédente)

```
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, via la variable *vitam_site_name*, 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
    rank: 0
  - name: offer-fs-2
    referent: false
    distant: true
    vitam_site_name: site2
    rank: 1
  - name: offer-fs-3
    referent: false
    distant: true
    vitam_site_name: site3
    rank: 2
# Optional params for each offers in vitam_strategy. If not set, the default values
→are applied.
#   referent: false           # true / false (default), only one per site must be
→referent
#   status: ACTIVE           # ACTIVE (default) / INACTIVE
#   vitam_site_name: distant-dc2 # default is the value of vitam_site_name defined
→in your local inventory file, should be specified with the vitam_site_name defined
→for the distant offer
#   distant: false           # true / false (default). If set to true, it will
→not check if the provider for this offer is correctly set
#   id: idoffre              # OPTIONAL, but IF ACTIVATED, MUST BE UNIQUE & SAME
→if on another site
#   asyncRead: false         # true / false (default). Should be set to true for
→tape offer only
#   rank: 0                  # Integer that indicates in ascending order the
→priority of the offer in the strategy
```

Exemple pour le site 2 (site secondaire) :

```
vitam_strategy:
  - name: offer-fs-2
    referent: true
```

Exemple pour le site 3 (site secondaire) :

```
vitam_strategy:
  - name: offer-fs-3
    referent: true
```

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
      rank: 0
    - name: offer-fs-2
      referent: false
      distant: true
      vitam_site_name: site2
      rank: 1
    - name: offer-fs-3
      referent: false
      distant: true
      vitam_site_name: site3
      rank: 2
    - name: offer-s3-1
      referent: false
      rank: 3
    - name: offer-s3-2
      referent: false
      distant: true
      vitam_site_name: site2
      rank: 4
    - name: offer-s3-3
      referent: false
```

(suite sur la page suivante)

(suite de la page précédente)

```

    distant: true
    vitam_site_name: site3
    rank: 5
  binary:
    - name: offer-s3-1
      referent: false
      rank: 0
    - name: offer-s3-2
      referent: false
      distant: true
      vitam_site_name: site2
      rank: 1
    - name: offer-s3-3
      referent: false
      distant: true
      vitam_site_name: site3
      rank: 2

```

Exemple pour le site 2 (site secondaire) :

```

other_strategies:
  metadata:
    - name: offer-fs-2
      referent: true
      rank: 0
    - name: offer-s3-2
      referent: false
      rank: 1
  binary:
    - name: offer-s3-2
      referent: false
      rank: 0

```

Exemple pour le site 3 (site secondaire) :

```

other_strategies:
  metadata:
    - name: offer-fs-3
      referent: true
      rank: 0
    - name: offer-s3-3
      referent: false
      rank: 1
  binary:
    - name: offer-s3-3
      referent: false
      rank: 0

```

4.2.2.1.7 plateforme_secret

Fichier : `deployment/environments/group_vars/all/main/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/main/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/advanced/vitam_vars.yml` et indiquer `https_enabled: true` dans `storageofferdefault`.

Il convient également 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

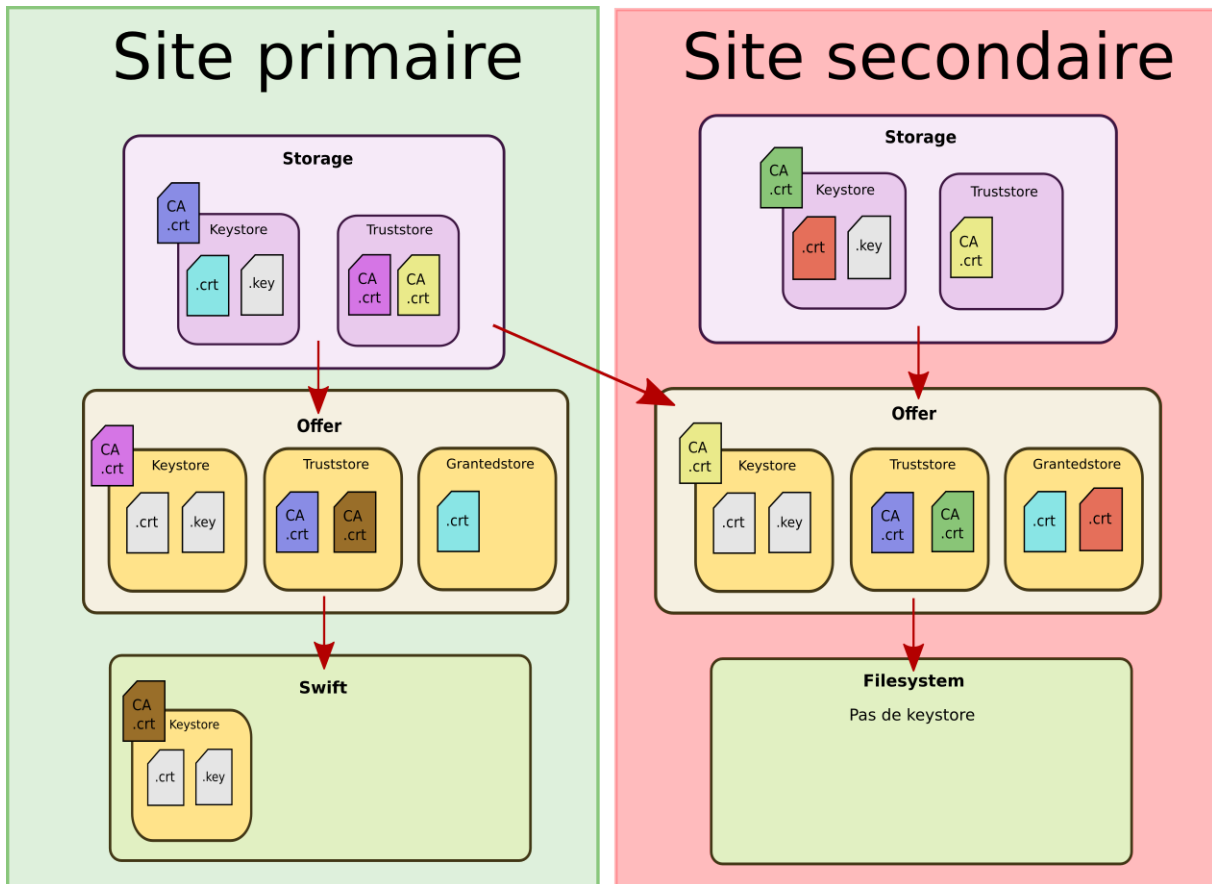


FIG. 2 – Vue détaillée des certificats entre le storage et l'offre en multi-site

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/*.crt` 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 playbooks 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 ##### Tests environments specifics #####
12
13 # EXTRA : Front reverse-proxy (test environments ONLY) ; add machine name after
```

(suite sur la page suivante)

(suite de la page précédente)

```

14 [reverse]
15 # optional : after machine, if this machine is different from VITAM machines, you can
16 ↪ specify another become user
17 # Example
18 # vitam-centos-01.vitam ansible_ssh_user=centos
19
20 [ldap] # Extra : OpenLDAP server
21 # LDAP server !!! NOT FOR PRODUCTION !!! Test only
22
23
24 [library]
25 # TODO: Put here servers where this service will be deployed : library
26
27
28 [hosts_dev_tools]
29 # TODO: Put here servers where this service will be deployed : mongo-express,
30 ↪ elasticsearch-head
31 # /\ WARNING !!! NOT FOR PRODUCTION
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
48
49 ##### Zone externe
50 [zone_external:children]
51 hosts_ihm_demo
52 hosts_ihm_recette
53
54 [hosts_ihm_demo]
55 # TODO: Put here servers where this service will be deployed : ihm-demo. If you use
56 ↪ vitam-ui or your own frontend, it is recommended to leave this group blank
57 # If you don't need consul for ihm-demo, you can set this var after each hostname :
58 # consul_disabled=true
59 # DEPRECATED / We'll soon be removed. Please consider using vitam-ui or your own
60 ↪ front-end
61 # /\ WARNING !!! NOT recommended for PRODUCTION
62
63 [hosts_ihm_recette]
64 # TODO: Put here servers where this service will be deployed : ihm-recette (extra
65 ↪ feature)
66 # DEPRECATED / We'll soon be removed.
67 # /\ WARNING !!! NOT FOR PRODUCTION

```

(suite sur la page suivante)

```
66
67
68 ##### Zone access
69
70 # Group definition ; DO NOT MODIFY
71 [zone_access:children]
72 hosts_ingest_external
73 hosts_access_external
74 hosts_collect_external
75
76 [hosts_ingest_external]
77 # TODO: Put here servers where this service will be deployed : ingest-external
78
79
80 [hosts_access_external]
81 # TODO: Put here servers where this service will be deployed : access-external
82
83
84 [hosts_collect_external]
85 # TODO: Put here servers where this service will be deployed : collect-external
86
87
88 ##### Zone applicative
89
90 # Group definition ; DO NOT MODIFY
91 [zone_applicative:children]
92 hosts_ingest_internal
93 hosts_processing
94 hosts_batch_report
95 hosts_worker
96 hosts_access_internal
97 hosts_metadata
98 hosts_functional_administration
99 hosts_scheduler
100 hosts_logbook
101 hosts_workspace
102 hosts_storage_engine
103 hosts_security_internal
104 hosts_collect_internal
105 hosts_metadata_collect
106 hosts_workspace_collect
107
108
109 [hosts_security_internal]
110 # TODO: Put here servers where this service will be deployed : security-internal
111
112
113 [hosts_logbook]
114 # TODO: Put here servers where this service will be deployed : logbook
115
116
117 [hosts_workspace]
118 # TODO: Put the server where this service will be deployed : workspace
119 # WARNING: put only ONE server for this service, not more !
120
121
122 [hosts_ingest_internal]
```

(suite sur la page suivante)

(suite de la page précédente)

```

123 # TODO: Put here servers where this service will be deployed : ingest-internal
124
125
126 [hosts_access_internal]
127 # TODO: Put here servers where this service will be deployed : access-internal
128
129
130 [hosts_metadata]
131 # TODO: Put here servers where this service will be deployed : metadata
132
133
134 [hosts_functional_administration]
135 # TODO: Put here servers where this service will be deployed : functional-
    ↪administration
136
137
138 [hosts_scheduler]
139 # TODO: Put here servers where this service will be deployed : scheduler
140 # Optional parameter after each host : vitam_scheduler_thread_count=<integer> ; This_
    ↪is the number of threads that are available for concurrent execution of jobs. ;_
    ↪default is 3 thread
141
142
143 [hosts_processing]
144 # TODO: Put the server where this service will be deployed : processing
145 # WARNING: put only one server for this service, not more !
146
147
148 [hosts_storage_engine]
149 # TODO: Put here servers where this service will be deployed : storage-engine
150
151
152 [hosts_batch_report]
153 # TODO: Put here servers where this service will be deployed : batch-report
154
155
156 [hosts_worker]
157 # TODO: Put here servers where this service will be deployed : worker
158 # 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)
159
160
161 [hosts_collect_internal]
162 # TODO: Put here servers where this service will be deployed : collect_internal
163
164
165 [hosts_metadata_collect]
166 # TODO: Put here servers where this service will be deployed : metadata_collect
167
168
169 [hosts_workspace_collect]
170 # TODO: Put the server where this service will be deployed : workspace_collect
171 # WARNING: put only ONE server for this service, not more !
172
173
174

```

(suite sur la page suivante)

```

175 ##### Zone storage
176
177 [zone_storage:children] # DO NOT MODIFY
178 hosts_storage_offer_default
179 hosts_mongodb_offer
180
181 [hosts_storage_offer_default]
182 # TODO: Put here servers where this service will be deployed : storage-offer-default
183 # LIMIT : only 1 offer per machine
184 # LIMIT and 1 machine per offer when filesystem or filesystem-hash provider
185 # Possibility to declare multiple machines with same provider only when provider is_
  ↳ s3 or swift.
186 # Mandatory param for each offer is offer_conf and points to offer_opts.yml & vault-
  ↳ vitam.yml (with same tree)
187 # Optionnal parameter: restic_enabled=true (only 1 per offer_conf) available for_
  ↳ providers filesystem*, openstack-swift-v3 & amazon-s3-v1
188 # for swift
189 # hostname-offre-1.vitam offer_conf=offer-swift-1 restic_enabled=true
190 # hostname-offre-2.vitam offer_conf=offer-swift-1
191 # for filesystem
192 # hostname-offre-2.vitam offer_conf=offer-fs-1 restic_enabled=true
193 # for s3
194 # hostname-offre-3.vitam offer_conf=offer-s3-1 restic_enabled=true
195 # hostname-offre-4.vitam offer_conf=offer-s3-1
196
197
198 [hosts_mongodb_offer:children]
199 hosts_mongos_offer
200 hosts_mongoc_offer
201 hosts_mongod_offer
202
203 [hosts_mongos_offer]
204 # WARNING : DO NOT COLLOCATE WITH [hosts_mongos_data]
205 # TODO: put here servers where this service will be deployed : mongos cluster for_
  ↳ storage offers
206 # Mandatory params
207 # - mongo_cluster_name=<offer_name> ; name of the cluster (should exist on vitam_
  ↳ strategy configuration in offer_opts.yml)
208 # The recommended practice is to install the mongos instance on the same servers as_
  ↳ the mongoc instances
209 # Example
210 # vitam-mongo-swift-offer-01 mongo_cluster_name=offer-swift-1
211 # vitam-mongo-swift-offer-02 mongo_cluster_name=offer-swift-1
212 # vitam-mongo-fs-offer-01 mongo_cluster_name=offer-fs-1
213 # vitam-mongo-fs-offer-02 mongo_cluster_name=offer-fs-1
214 # vitam-mongo-s3-offer-01 mongo_cluster_name=offer-s3-1
215 # vitam-mongo-s3-offer-02 mongo_cluster_name=offer-s3-1
216
217
218 [hosts_mongoc_offer]
219 # WARNING : DO NOT COLLOCATE WITH [hosts_mongoc_data]
220 # TODO: put here servers where this service will be deployed : mongoc cluster for_
  ↳ storage offers
221 # Mandatory params
222 # - mongo_cluster_name=<offer_name> ; name of the cluster (should exist on vitam_
  ↳ strategy configuration in offer_opts.yml)
223 # Optional params

```

(suite sur la page suivante)

(suite de la page précédente)

```

224 # - mongo_rs_bootstrap=true ; mandatory for 1 node, some init commands will be_
↳executed on it
225 # The recommended practice is to install the mongoc instance on the same servers as_
↳the mongos instances
226 # Recommended practice in production: use 3 instances
227 # IMPORTANT : Updating cluster configuration is NOT supported. Do NOT add/remove a_
↳host to an existing replica set.
228 # Example :
229 # vitam-mongo-swift-offer-01    mongo_cluster_name=offer-swift-1    mongo_rs_
↳bootstrap=true
230 # vitam-mongo-swift-offer-02    mongo_cluster_name=offer-swift-1
231 # vitam-swift-offer             mongo_cluster_name=offer-swift-1
232 # vitam-mongo-fs-offer-01       mongo_cluster_name=offer-fs-1        mongo_rs_
↳bootstrap=true
233 # vitam-mongo-fs-offer-02       mongo_cluster_name=offer-fs-1
234 # vitam-fs-offer                mongo_cluster_name=offer-fs-1
235 # vitam-mongo-s3-offer-01       mongo_cluster_name=offer-s3-1        mongo_rs_
↳bootstrap=true
236 # vitam-mongo-s3-offer-02       mongo_cluster_name=offer-s3-1
237 # vitam-s3-offer                mongo_cluster_name=offer-s3-1
238
239
240 [hosts_mongod_offer]
241 # WARNING : DO NOT COLLOCATE WITH [hosts_mongod_data]
242 # TODO: put here servers where this service will be deployed : mongod cluster for_
↳storage offers
243 # Mandatory params
244 # - mongo_cluster_name=<offer_name> ; name of the cluster (should exist on vitam_
↳strategy configuration in offer_opts.yml)
245 # - mongo_shard_id=x ; increment by 1 from 0 to n to create multiple shards
246 # Optional params
247 # - mongo_rs_bootstrap=true (default: false); mandatory for 1 node of the shard,_
↳some init commands will be executed on it
248 # - mongo_arbiter=true (default: false); the node will be only an arbiter, it will_
↳not store data ; do not add this parameter on a mongo_rs_bootstrap node, maximum 1_
↳node per shard
249 # - mongod_memory=x (default: unset); this will force the wiredtiger cache size to x_
↳(unit is GB)
250 # - is_small=true (default: false); this will force the priority for this server to_
↳be lower when electing master ; hardware can be downgraded for this machine
251 # Recommended practice in production: use 3 instances per shard
252 # IMPORTANT : Updating cluster configuration is NOT supported. Do NOT add/remove a_
↳host to an existing replica set, update shard id, arbiter mode or PSSmin_
↳configuration.
253 # Example :
254 # vitam-mongo-swift-offer-01    mongo_cluster_name=offer-swift-1    mongo_shard_id=0    _
↳mongo_rs_bootstrap=true
255 # vitam-mongo-swift-offer-02    mongo_cluster_name=offer-swift-1    mongo_shard_id=0
256 # vitam-swift-offer             mongo_cluster_name=offer-swift-1    mongo_shard_id=0    _
↳mongo_arbiter=true
257 # vitam-mongo-fs-offer-01       mongo_cluster_name=offer-fs-1        mongo_shard_id=0    _
↳mongo_rs_bootstrap=true
258 # vitam-mongo-fs-offer-02       mongo_cluster_name=offer-fs-1        mongo_shard_id=0
259 # vitam-fs-offer                mongo_cluster_name=offer-fs-1        mongo_shard_id=0    _
↳mongo_arbiter=true
260 # vitam-mongo-s3-offer-01       mongo_cluster_name=offer-s3-1        mongo_shard_id=0    _
↳mongo_rs_bootstrap=true

```

(suite sur la page suivante)

(suite de la page précédente)

```

261 # vitam-mongo-s3-offer-02      mongo_cluster_name=offer-s3-1      mongo_shard_id=0
↳is_small=true # PSSmin, this machine needs less hardware
262 # vitam-s3-offer              mongo_cluster_name=offer-s3-1      mongo_shard_id=0
↳mongo_arbiter=true
263
264
265 ##### Zone data
266
267 # Group definition ; DO NOT MODIFY
268 [zone_data:children]
269 hosts_elasticsearch_data
270 hosts_mongodb_data
271
272 [hosts_elasticsearch_data]
273 # TODO: Put here servers where this service will be deployed : elasticsearch-data
↳cluster
274 # 2 params available for huge environments (parameter to be declared after each
↳server) :
275 #   is_data=true/false
276 #   is_master=true/false
277 #   for site/room balancing : is_balancing=<whatever> so replica can be applied on
↳all sites/rooms ; default is vitam_site_name
278 #   other options are not handled yet
279 # defaults are set to true, if undefined. If defined, at least one server MUST be is_
↳data=true
280 # Examples :
281 # server1 is_master=true is_data=false
282 # server2 is_master=false is_data=true
283 # More explanation here : https://www.elastic.co/guide/en/elasticsearch/reference/5.6/
↳modules-node.html
284
285
286 # Group definition ; DO NOT MODIFY
287 [hosts_mongodb_data:children]
288 hosts_mongos_data
289 hosts_mongoc_data
290 hosts_mongod_data
291
292 [hosts_mongos_data]
293 # WARNING : DO NOT COLLOCATE WITH [hosts_mongos_offer]
294 # TODO: Put here servers where this service will be deployed : mongos_data cluster
295 # Mandatory params
296 # - mongo_cluster_name=mongo-data ; "mongo-data" is mandatory
297 # The recommended practice is to install the mongos instance on the same servers as
↳the mongoc instances
298 # Example :
299 # vitam-mdbs-01      mongo_cluster_name=mongo-data
300 # vitam-mdbs-02      mongo_cluster_name=mongo-data
301 # vitam-mdbs-03      mongo_cluster_name=mongo-data
302
303
304 [hosts_mongoc_data]
305 # WARNING : DO NOT COLLOCATE WITH [hosts_mongoc_offer]
306 # TODO: Put here servers where this service will be deployed : mongoc_data cluster
307 # Mandatory params
308 # - mongo_cluster_name=mongo-data ; "mongo-data" is mandatory
309 # Optional params

```

(suite sur la page suivante)

(suite de la page précédente)

```

310 # - mongo_rs_bootstrap=true ; mandatory for 1 node, some init commands will be_
    ↳executed on it
311 # The recommended practice is to install the mongoc instance on the same servers as_
    ↳the mongos instances
312 # Recommended practice in production: use 3 instances
313 # IMPORTANT : Updating cluster configuration is NOT supported. Do NOT add/remove a_
    ↳host to an existing replica set.
314 # Example :
315 # vitam-mdbs-01  mongo_cluster_name=mongo-data  mongo_rs_bootstrap=true
316 # vitam-mdbs-02  mongo_cluster_name=mongo-data
317 # vitam-mdbs-03  mongo_cluster_name=mongo-data
318
319
320 [hosts_mongod_data]
321 # WARNING : DO NOT COLLOCATE WITH [hosts_mongod_offer]
322 # TODO: Put here servers where this service will be deployed : mongod_data cluster
323 # Each replica_set should have an odd number of members (2n + 1)
324 # Reminder: For Vitam, one mongoddb shard is using one replica_set
325 # Mandatory params
326 # - mongo_cluster_name=mongo-data ; "mongo-data" is mandatory
327 # - mongo_shard_id=x ; increment by 1 from 0 to n to create multiple shards
328 # Optional params
329 # - mongo_rs_bootstrap=true (default: false); mandatory for 1 node of the shard,_
    ↳some init commands will be executed on it
330 # - mongo_arbiter=true (default: false); the node will be only an arbiter, it will_
    ↳not store data ; do not add this parameter on a mongo_rs_bootstrap node, maximum 1_
    ↳node per shard
331 # - mongod_memory=x (default: unset); this will force the wiredtiger cache size to x_
    ↳(unit is GB) ; can be usefull when colocalization with elasticsearch
332 # - is_small=true (default: false); this will force the priority for this server to_
    ↳be lower when electing master ; hardware can be downgraded for this machine
333 # Recommended practice in production: use 3 instances per shard
334 # IMPORTANT : Updating cluster configuration is NOT supported. Do NOT add/remove a_
    ↳host to an existing replica set, update shard id, arbiter mode or PSSmin_
    ↳configuration.
335 # Example:
336 # vitam-mdbd-01  mongo_cluster_name=mongo-data  mongo_shard_id=0  mongo_rs_
    ↳bootstrap=true
337 # vitam-mdbd-02  mongo_cluster_name=mongo-data  mongo_shard_id=0
338 # vitam-mdbd-03  mongo_cluster_name=mongo-data  mongo_shard_id=0  is_small=true #_
    ↳PSSmin, this machine needs less hardware
339 # vitam-mdbd-04  mongo_cluster_name=mongo-data  mongo_shard_id=1  mongo_rs_
    ↳bootstrap=true
340 # vitam-mdbd-05  mongo_cluster_name=mongo-data  mongo_shard_id=1
341 # vitam-mdbd-06  mongo_cluster_name=mongo-data  mongo_shard_id=1  mongo_arbiter=true
342
343
344 ##### Zone admin
345
346 # Group definition ; DO NOT MODIFY
347 [zone_admin:children]
348 hosts_cerebro
349 hosts_consul_server
350 hosts_kibana_data
351 log_servers
352 hosts_elasticsearch_log
353 prometheus

```

(suite sur la page suivante)

```
354 hosts_grafana
355
356 [hosts_cerebro]
357 # TODO: Put here servers where this service will be deployed : vitam-elasticsearch-
    ↳cerebro
358 # /\ WARNING !!! NOT recommended for PRODUCTION
359
360
361 [hosts_consul_server]
362 # TODO: Put here servers where this service will be deployed : consul
363 # Recommended practice in production: use 3 instances
364
365
366 [hosts_kibana_data]
367 # TODO: Put here servers where this service will be deployed : kibana (for data_
    ↳cluster)
368 # WARNING : DEPRECATED / We'll soon be removed.
369 # /\ WARNING !!! NOT FOR PRODUCTION
370
371
372 [log_servers:children]
373 hosts_kibana_log
374 hosts_logstash
375
376 [hosts_kibana_log]
377 # TODO: Put here servers where this service will be deployed : kibana (for log_
    ↳cluster)
378
379
380 [hosts_logstash]
381 # TODO: Put here servers where this service will be deployed : logstash
382 # IF you connect VITAM to external SIEM, DO NOT FILL THE SECTION
383
384
385 [hosts_elasticsearch_log]
386 # TODO: Put here servers where this service will be deployed : elasticsearch-log_
    ↳cluster
387 # IF you connect VITAM to external SIEM, DO NOT FILL THE SECTION
388
389
390 ##### Extra VITAM applications #####
391 [prometheus:children]
392 hosts_prometheus
393 hosts_alertmanager
394
395 [hosts_prometheus]
396 # TODO: Put here server where this service will be deployed : prometheus server
397
398
399 [hosts_alertmanager]
400 # TODO: Put here servers where this service will be deployed : alertmanager
401
402
403 [hosts_grafana]
404 # TODO: Put here servers where this service will be deployed : grafana-server
405
406
```

(suite sur la page suivante)

(suite de la page précédente)

```

407 ##### Global vars #####
408
409 [hosts:vars]
410
411 # =====
412 # VITAM
413 # =====
414
415 # Declare user for ansible on target machines
416 ansible_ssh_user=
417 # Can target user become as root ? ; true is required by VITAM (usage of a sudoer is
↳mandatory)
418 ansible_become=true
419 # How can ansible switch to root ?
420 # See https://docs.ansible.com/ansible/latest/user_guide/become.html
421
422 # Related to Consul ; apply in a table your DNS server(s)
423 # Example : dns_servers=["8.8.8.8","8.8.4.4"]
424 # If no dns recursors are available, leave this value empty.
425 dns_servers=
426
427 # Define local Consul datacenter name
428 # CAUTION !!! Only alphanumeric characters when using s3 as offer backend !!!
429 vitam_site_name=prod-dc1
430
431 # On offer, value is the prefix for all container's names. If upgrading from R8, you
↳MUST UNCOMMENT this parameter AS IS !!!
432 #vitam_prefix_offer=""
433
434 # check whether on primary site (true) or secondary (false)
435 primary_site=true
436
437 # =====
438 # EXTRA
439 # =====
440
441 ### vitam-itest repository ###
442 vitam_tests_branch=master
443 vitam_tests_gitrepo_protocol=
444 vitam_tests_gitrepo_baseurl=
445 vitam_tests_gitrepo_url=
446
447 # Used when VITAM is behind a reverse proxy (provides configuration for reverse proxy
↳&& displayed in header page)
448 vitam_reverse_external_dns=
449 # For reverse proxy use
450 reverse_proxy_port=443
451 vitam_reverse_external_protocol=https
452 # http_proxy env var to use ; has to be declared even if empty
453 http_proxy_environment=

```

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 (regex : `[A-Za-z0-9-]`).

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 `main.yml`

La configuration des principaux paramètres est réalisée dans le fichier `reper-toire_inventory|'group_vars/all/main/main.yml'`, comme suit :

```
1 ---
2
3 # TENANTS
4 # List of active tenants
5 vitam_tenant_ids: [0,1,2,3,4,5,6,7,8,9]
6 # For functional-administration, manage master/slave tenant configuration
7 # http://www.programmevitam.fr/ressources/DocCourante/html/installation/installation/
8 # ↪21-addons.html#passage-des-identifiants-des-referentiels-en-mode-esclave
9 vitam_tenants_usage_external:
10   - name: 0
11     identifiers:
12       - INGEST_CONTRACT
13       - ACCESS_CONTRACT
14       - MANAGEMENT_CONTRACT
15       - ARCHIVE_UNIT_PROFILE
16   - name: 1
17     identifiers:
18       - INGEST_CONTRACT
19       - ACCESS_CONTRACT
20       - MANAGEMENT_CONTRACT
21       - PROFILE
22       - SECURITY_PROFILE
23       - CONTEXT
24
25 # GRIFFINS
26 # Vitam griffins required to launch preservation scenario
27 # Example:
```

(suite sur la page suivante)

(suite de la page précédente)

```

27 # vitam_griffins: ["vitam-imagemagick-griffin", "vitam-libreoffice-griffin", "vitam-
↳jhove-griffin", "vitam-odfvalidator-griffin", "vitam-siegfried-griffin", "vitam-
↳tesseract-griffin", "vitam-verapdf-griffin", "vitam-ffmpeg-griffin"]
28 vitam_griffins: []
29
30 # CONSUL
31 consul:
32   network: "ip_admin" # Which network to use for consul communications ? ip_admin or
↳ip_service ?
33 consul_remote_sites:
34 # wan contains the wan addresses of the consul server instances of the external
↳vitam sites
35 # Exemple, if our local dc is dc1, we will need to set dc2 & dc3 wan conf:
36 #   - dc2:
37 #     wan: ["10.10.10.10", "1.1.1.1"]
38 #   - dc3:
39 #     wan: ["10.10.10.11", "1.1.1.1"]
40
41 # LOGGING
42 # vitam_defaults:
43 #   access_retention_days: 30 # Number of days for file retention
44 #   access_total_size_cap: "10GB" # total acceptable size
45 #   logback_max_file_size: "10MB"
46 #   logback_total_size_cap:
47 #     file:
48 #       history_days: 30
49 #       totalsize: "5GB"
50 #   security:
51 #     history_days: 30
52 #     totalsize: "5GB"
53
54 # ELASTICSEARCH
55 # 'number_of_shards': number of shards per index, every ES shard is stored as a
↳lucene index
56 # 'number_of_replicas': number of additional copies of primary shards
57 # Total number of shards: number_of_shards * (1 primary + M number_of_replicas)
58 # CAUTION: The total number of shards should be lower than or equal to the number of
↳elasticsearch-data instances in the cluster
59 # More details in groups_vars/all/advanced/tenants_vars.yml file
60 vitam_elasticsearch_tenant_indexation:
61   default_config:
62     # Default settings for masterdata collections (1 index per collection)
63     masterdata:
64       number_of_shards: 1
65       number_of_replicas: 2
66     # Default settings for unit indexes (1 index per tenant)
67     unit:
68       number_of_shards: 1
69       number_of_replicas: 2
70     # Default settings for object group indexes (1 index per tenant)
71     objectgroup:
72       number_of_shards: 1
73       number_of_replicas: 2
74     # Default settings for logbook operation indexes (1 index per tenant)
75     logbookoperation:
76       number_of_shards: 1
77       number_of_replicas: 2

```

(suite sur la page suivante)

```

78 # Default settings for collect_unit indexes
79 collect_unit:
80     number_of_shards: 1
81     number_of_replicas: 2
82 # Default settings for collect_objectgroup indexes
83 collect_objectgroup:
84     number_of_shards: 1
85     number_of_replicas: 2
86
87 collect_grouped_tenants:
88 - name: 'all'
89 # Group all tenants for collect's indexes (collect_unit & collect_objectgroup)
90 tenants: "{{ vitam_tenant_ids | join(',') }}"
91
92 elasticsearch:
93     log:
94         index_templates:
95             default:
96                 shards: 1
97                 replica: 1
98     data:
99         index_templates:
100             default:
101                 shards: 1
102                 replica: 2
103 curator:
104     log:
105         metrics:
106             close: 7
107             delete: 30
108         logstash:
109             close: 7
110             delete: 30
111
112 # PACKAGES
113 disable_internet_repositories_install: true # Disable EPEL or Debian backports_
↪repositories install

```

Une attention particulière doit être portée à la configuration du nombre de shards et de replicas dans le paramètre `vitam_elasticsearch_tenant_indexation.default_config`.

Voir aussi :

Se référer au chapitre « Gestion des indexes Elasticsearch dans un contexte massivement multi-tenants » du *DEX* pour plus d'informations sur cette fonctionnalité.

Avertissement : Attention, en cas de modification de la distribution des tenants, une procédure de réindexation de la base `elasticsearch-data` est nécessaire. Cette procédure est à la charge de l'exploitation et nécessite un arrêt de service sur la plateforme. La durée d'exécution de cette réindexation dépend de la quantité de données à traiter.

Voir aussi :

Se référer au chapitre « Réindexation » du *DEX* pour plus d'informations.

4.2.3.2.3 Fichier vitam_security.yml

La configuration des droits d'accès à VITAM est réalisée dans le fichier `reper-toire_inventory|group_vars/all/advanced/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
9  # Indicate context certificates relative paths under {{ inventory_dir }}/certs/client-
10 ↪external/clients
11 # vitam-admin-int is mandatory for internal use (PRONOM upload)
12 admin_context_certs:
13   - "{{ 'collect-external/collect-external.crt' if groups['hosts_collect_external'] |
14 ↪default([]) | length > 0 else '' }}"
15   - "{{ 'ihm-demo/ihm-demo.crt' if groups['hosts_ihm_demo'] | default([]) | length >
16 ↪0 else '' }}"
17   - "{{ 'ihm-recette/ihm-recette.crt' if groups['hosts_ihm_recette'] | default([]) |
18 ↪length > 0 else '' }}"
19   - "vitam-admin-int/vitam-admin-int.crt"
20
21 # Indicate here all the personal certificates relative paths under {{ inventory_dir }}
22 ↪/certs/client-vitam-users/clients
23 admin_personal_certs: [ ]
24
25 # Admin security profile name
26 admin_security_profile: "admin-security-profile"
27
28 admin_basic_auth_user: "adminUser"
29
30 # SELinux state, can be: enforcing, permissive, disabled
31 selinux_state: "disabled"
32 # SELinux Policy, can be: targeted, minimum, mls
33 selinux_policy: "targeted"
34 # If needed, reboot the VM to enable SELinux
35 selinux_reboot: True
36 # Relabel the entire filesystem ?
37 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 68).

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 68).

4.2.3.2.4 Fichier offers_opts.yml

La déclaration de configuration des offres de stockage associées se fait dans le fichier `!reper-toire_inventory|'group_vars/all/main/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 will be ordered by the property rank. It has to be
   ↳completed if more offers are necessary
3  # The property rank indicates the rank of the offer in the strategy. The
   ↳ranking is done in ASC order and should be different for all declared
   ↳offers
4  vitam_strategy:
5    - name: offer-fs-1
6      referent: true
7      rank: 0
8
9  # Optional params for each offers in vitam_strategy. If not set, the default
   ↳values are applied.
10 #   referent: false           # true / false (default), only one per
   ↳site must be referent
11 #   status: ACTIVE           # ACTIVE (default) / INACTIVE
12 #   vitam_site_name: distant-dc2 # default is the value of vitam_site_name
   ↳defined in your local inventory file, should be specified with the vitam
   ↳site_name defined for the distant offer
13 #   distant: false           # true / false (default). If set to true,
   ↳it will not check if the provider for this offer is correctly set
14 #   id: idoffre              # OPTIONAL, but IF ACTIVATED, MUST BE
   ↳UNIQUE & SAME if on another site
15 #   asyncRead: false         # true / false (default). Should be set to
   ↳true for tape offer only
16 #   rank: 0                  # Integer that indicates in ascending
   ↳order the priority of the offer in the strategy
17
18 # Example for tape offer:
19 # Tape offer mustn't be referent (referent: false) and should be configured
   ↳as asynchrone read (asyncRead: true)
20 # - name: offer-tape-1
21 #   referent: false
22 #   asyncRead: true
23 #   rank: 0
24
25 # Example distant offer:
26 # - name: distant
27 #   referent: false
28 #   vitam_site_name: distant-dc2
29 #   distant: true # Only add this parameter when distant offer (not on same
   ↳platform)
30 #   rank: 1
31
32 # WARNING : multi-strategy is a BETA functionality
33 # More strategies can be added but are optional
34 # Strategy name must only use [a-z][a-z0-9-]* pattern
35 # Any strategy must contain at least one offer
36 # This list of offers is ordered. It can and has to be completed if more
   ↳offers are necessary
37 # Every strategy can define at most one referent offer.

```

(suite sur la page suivante)

(suite de la page précédente)

```

38 # other_strategies:
39 #   metadata:
40 #     - name: offer-fs-1
41 #       referent: true
42 #       rank: 0
43 #     - name: offer-fs-2
44 #       referent: false
45 #       rank: 1
46 #   binary:
47 #     - name: offer-fs-2
48 #       referent: false
49 #       rank: 0
50 #     - name: offer-s3-1
51 #       referent: false
52 #       rank: 1
53
54 # DON'T forget to add associated passwords in vault-vitam.yml with same tree,
↳when using provider openstack-swift*
55 # ATTENTION !!! Each offer has to have a distinct name, except for clusters,
↳binding a same physical storage
56 # WARNING : for offer names, please only use [a-z][a-z0-9-]* pattern
vitam_offers:
57   offer-fs-1:
58     # param can be filesystem-hash (recomended) or filesystem (not,
↳recomended)
59     provider: filesystem-hash
60     ### Optional parameters
61     # Offer log compaction
62     offer_log_compaction:
63       ## Expiration, here offer logs 21 days old will be compacted
64       expiration_value: 21
65       ## Choose one of "MILLENNIA", "HALF_DAYS", "MILLIS", "FOREVER", "MICROS
↳", "CENTURIES", "DECADES", "YEARS", "DAYS", "SECONDS", "HOURS", "MONTHS",
↳"WEEKS", "NANOS", "MINUTES", "ERAS"
66       expiration_unit: "DAYS"
67       ## Compaction bulk size here 10 000 offers logs (at most) will be,
↳compacted (Expected value between 1 000 and 200 000)
68       compaction_size: 10000
69       # Batch processing thread pool size
70       maxBatchThreadPoolSize: 32
71       # Batch metadata computation timeout in seconds
72       batchMetadataComputationTimeout: 600
73 #####
↳###
74
75   offer-swift-1:
76     # provider : openstack-swift for v1 or openstack-swift-v3 for v3
77     provider: openstack-swift-v3
78     # swiftKeystoneAuthUrl : URL de connexion à keystone
79     swiftKeystoneAuthUrl: https://openstack-hostname:port/auth/1.0
80     # swiftDomain : domaine OpenStack dans lequel l'utilisateur est,
↳enregistré
81     swiftDomain: domaine
82     # swiftUser: has to be set in vault-vitam.yml (encrypted) with same,
↳structure => DO NOT COMMENT OUT
83     # swiftPassword: has to be set in vault-vitam.yml (encrypted) with same,
↳structure => DO NOT COMMENT OUT
84     # swiftProjectName : nom du projet openstack

```

(suite sur la page suivante)

(suite de la page précédente)

```

85  swiftProjectName: monTenant
86  ### Optional parameters
87  # swiftUrl: optional variable to force the swift URL
88  # swiftUrl: https://swift-hostname:port/swift/v1
89  #SSL TrustStore
90  swiftTrustStore: /chemin_vers_mon_fichier/monSwiftTrustStore.jks
91  #Max connection (concurrent connections), per route, to keep in pool (if a
↳ pooling ConnectionManager is used) (optional, 200 by default)
92  swiftMaxConnectionsPerRoute: 200
93  #Max total connection (concurrent connections) to keep in pool (if a
↳ pooling ConnectionManager is used) (optional, 1000 by default)
94  swiftMaxConnections: 1000
95  #Max time (in milliseconds) for waiting to establish connection
↳ (optional, 200000 by default)
96  swiftConnectionTimeout: 200000
97  #Max time (in milliseconds) waiting for a data from the server (socket)
↳ (optional, 60000 by default)
98  swiftReadTimeout: 60000
99  #Default number of retries on errors
100 swiftNbRetries: 3
101 #Time (in seconds) to renew a token before expiration occurs (blocking)
↳ (optional, 60 by default)
102 swiftHardRenewTokenDelayBeforeExpireTime: 60
103 #Time (in seconds) to renew a token before expiration occurs (optional,
↳ 300 by default)
104 swiftSoftRenewTokenDelayBeforeExpireTime: 300
105 # Offer log compaction
106 offer_log_compaction:
107   ## Expiration, here offer logs 21 days old will be compacted
108   expiration_value: 21
109   ## Choose one of "MILLENNIA", "HALF_DAYS", "MILLIS", "FOREVER", "MICROS
↳ ", "CENTURIES", "DECADES", "YEARS", "DAYS", "SECONDS", "HOURS", "MONTHS",
↳ "WEEKS", "NANOS", "MINUTES", "ERAS"
110   expiration_unit: "DAYS"
111   ## Compaction bulk size here 10 000 offers logs (at most) will be
↳ compacted (Expected value between 1 000 and 200 000)
112   compaction_size: 10000
113   # Batch processing thread pool size
114   maxBatchThreadPoolSize: 32
115   # Batch metadata computation timeout in seconds
116   batchMetadataComputationTimeout: 600
117   # Enable / Disable use of vitam custom headers for offer requests
118   enableCustomHeaders: false
119   # List of vitam custom headers used by offer requests
120   #customHeaders:
121   # - key: 'Cookie'
122   #   value: 'Origin=vitam'
123   #####
↳ ###
124 offer-s3-1:
125   # provider : can only be amazon-s3-v1 for Amazon SDK S3 V1
126   provider: 'amazon-s3-v1'
127   # s3Endpoint : URL of connection to S3
128   s3Endpoint: http://172.17.0.2:6007
129   ### Optional parameters
130   # s3RegionName (optional): Region name (default value us-east-1)
131   s3RegionName: us-west-1

```

(suite sur la page suivante)

(suite de la page précédente)

```

132 # s3SignerType (optional): Signing algorithm.
133 #   - signature V4 : 'AWSS3V4SignerType' (default value)
134 #   - signature V2 : 'S3SignerType'
135 s3SignerType: AWSS3V4SignerType
136 # s3PathStyleAccessEnabled (optional): 'true' to access bucket in "path-
↳style", else "virtual-hosted-style" (true by default)
137 s3PathStyleAccessEnabled: true
138 # s3MaxConnections (optional): Max total connection (concurrent_
↳connections) (50 by default)
139 s3MaxConnections: 1000
140 # s3ConnectionTimeout (optional): Max time (in milliseconds) for waiting_
↳to establish connection (10000 by default)
141 s3ConnectionTimeout: 200000
142 # s3SocketTimeout (optional): Max time (in milliseconds) for reading_
↳from a connected socket (50000 by default)
143 s3SocketTimeout: 50000
144 # s3RequestTimeout (optional): Max time (in milliseconds) for a request_
↳(0 by default, disabled)
145 s3RequestTimeout: 0
146 # s3ClientExecutionTimeout (optional): Max time (in milliseconds) for a_
↳request by java client (0 by default, disabled)
147 s3ClientExecutionTimeout: 0
148 # Offer log compaction
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   # Batch processing thread pool size
157   maxBatchThreadPoolSize: 32
158   # Batch metadata computation timeout in seconds
159   batchMetadataComputationTimeout: 600
160 #####
↳###
161 offer-tape-1:
162   provider: tape-library
163   # tapeLibraryConfiguration:
164   #   ...
165   # topology:
166   #   ...
167   # tapeLibraries:
168   #   ...
169   # Offer log compaction
170   offer_log_compaction:
171     ## Expiration, here offer logs 21 days old will be compacted
172     expiration_value: 21
173     ## Choose one of "MILLENNIA", "HALF_DAYS", "MILLIS", "FOREVER", "MICROS
↳", "CENTURIES", "DECADES", "YEARS", "DAYS", "SECONDS", "HOURS", "MONTHS",
↳"WEEKS", "NANOS", "MINUTES", "ERAS"
174     expiration_unit: "DAYS"
175     ## Compaction bulk size here 10 000 offers logs (at most) will be_
↳compacted (Expected value between 1 000 and 200 000)

```

(suite sur la page suivante)

```
176     compaction_size: 10000
177     # Batch processing thread pool size
178     maxBatchThreadPoolSize: 32
179     # Batch metadata computation timeout in seconds
180     batchMetadataComputationTimeout: 600
181     #####
182     ↪###
183     # WARNING: Swift V1 is deprecated
184     # example_swift_v1:
185     #   provider: openstack-swift
186     #   swiftKeystoneAuthUrl: https://keystone/auth/1.0
187     #   swiftDomain: domain
188     #   swiftUser: has to be set in vault-vitam.yml (encrypted) with same_
189     ↪structure => DO NOT COMMENT OUT
190     #   swiftPassword: has to be set in vault-vitam.yml (encrypted) with same_
191     ↪structure => DO NOT COMMENT OUT
192     # THIS PART IS ONLY FOR CLEANING (and mandatory for this use case)
193     #   swiftProjectId: related to OS_PROJECT_ID
194     #   swiftRegionName: related to OS_REGION_NAME
195     #   swiftInterface: related to OS_INTERFACE
196     # example_swift_v3:
197     #   provider: openstack-swift-v3
198     #   swiftKeystoneAuthUrl: https://keystone/v3
199     #   swiftDomain: domaine
200     #   swiftUser: has to be set in vault-vitam.yml (encrypted) with same_
201     ↪structure => DO NOT COMMENT OUT
202     #   swiftPassword: has to be set in vault-vitam.yml (encrypted) with same_
203     ↪structure => DO NOT COMMENT OUT
204     #   swiftProjectName: monTenant
205     #   projectName: monTenant
206     # THIS PART IS ONLY FOR CLEANING (and mandatory for this use case)
207     #   swiftProjectId: related to OS_PROJECT_ID
208     #   swiftRegionName: related to OS_REGION_NAME
209     #   swiftInterface: related to OS_INTERFACE
210
211     #   swiftTrustStore: /chemin_vers_mon_fichier/monSwiftTrustStore.jks
212     #   swiftMaxConnectionsPerRoute: 200
213     #   swiftMaxConnections: 1000
214     #   swiftConnectionTimeout: 200000
215     #   swiftReadTimeout: 60000
216     #   Time (in seconds) to renew a token before expiration occurs
217     #   swiftHardRenewTokenDelayBeforeExpireTime: 60
218     #   swiftSoftRenewTokenDelayBeforeExpireTime: 300
219     #   enableCustomHeaders: false
220     #   customHeaders:
221     #     - key: 'Cookie'
222     #       value: 'Origin=vitam'
```

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.

4.2.3.2.5 Fichier `cots_vars.yml`

La configuration s'effectue dans le fichier `repertoire_inventory/group_vars/all/advanced/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      log_level: WARN # Available log_level are: TRACE, DEBUG, INFO, WARN or
↳ERR
8
9  # Please uncomment and fill values if you want to connect VITAM to external
↳SIEM
10 # external_siem:
11 #     host:
12 #     port:
13
14  elasticsearch:
15      log:
16          host: "elasticsearch-log.service.{{ consul_domain }}"
17          port_http: "9201"
18          groupe: "log"
19          baseuri: "elasticsearch-log"
20          cluster_name: "elasticsearch-log"
21          consul_check_http: 10 # in seconds
22          consul_check_tcp: 10 # in seconds
23          action_log_level: error
24          https_enabled: false
25          indices_fielddata_cache_size: '30%' # related to https://www.elastic.
↳co/guide/en/elasticsearch/reference/7.6/modules-fielddata.html
26          indices_breaker_fielddata_limit: '40%' # related to https://www.
↳elastic.co/guide/en/elasticsearch/reference/7.6/circuit-breaker.html
↳#fielddata-circuit-breaker
27          dynamic_timeout: 30s
28          # default index template
29          index_templates:
30              packetbeat:
31                  shards: 5
32          log_appenders:
33              root:
34                  log_level: "info"
35              rolling:

```

(suite sur la page suivante)

```

36         max_log_file_size: "100MB"
37         max_total_log_size: "5GB"
38         max_files: "50"
39     deprecation_rolling:
40         max_log_file_size: "100MB"
41         max_total_log_size: "1GB"
42         max_files: "10"
43         log_level: "warn"
44     index_search_slowlog_rolling:
45         max_log_file_size: "100MB"
46         max_total_log_size: "1GB"
47         max_files: "10"
48         log_level: "warn"
49     index_indexing_slowlog_rolling:
50         max_log_file_size: "100MB"
51         max_total_log_size: "1GB"
52         max_files: "10"
53         log_level: "warn"
54     # By default, is commented. Should be uncommented if ansible_
↳ computes badly vCPUs number ; values are associated vCPUs numbers ;
↳ please adapt to your configuration
55     # thread_pool:
56     #     index:
57     #         size: 2
58     #     get:
59     #         size: 2
60     #     search:
61     #         size: 2
62     #     write:
63     #         size: 2
64     #     warmer:
65     #         max: 2
66     data:
67         host: "elasticsearch-data.service.{{ consul_domain }}"
68         # default is 0.1 (10%) and should be quite enough in most cases
69         #index_buffer_size_ratio: "0.15"
70         port_http: "9200"
71         groupe: "data"
72         baseuri: "elasticsearch-data"
73         cluster_name: "elasticsearch-data"
74         consul_check_http: 10 # in seconds
75         consul_check_tcp: 10 # in seconds
76         action_log_level: debug
77         https_enabled: false
78         indices_fielddata_cache_size: '30%' # related to https://www.elastic.
↳ co/guide/en/elasticsearch/reference/6.5/modules-fielddata.html
79         indices_breaker_fielddata_limit: '40%' # related to https://www.
↳ elastic.co/guide/en/elasticsearch/reference/6.5/circuit-breaker.html
↳ #fielddata-circuit-breaker
80         dynamic_timeout: 30s
81         # default index template
82         index_templates:
83         log_appenders:
84             root:
85                 log_level: "info"
86             rolling:
87                 max_log_file_size: "100MB"

```

(suite sur la page suivante)

(suite de la page précédente)

```

88         max_total_log_size: "5GB"
89         max_files: "50"
90     deprecation_rolling:
91         max_log_file_size: "100MB"
92         max_total_log_size: "5GB"
93         max_files: "50"
94         log_level: "warn"
95     index_search_slowlog_rolling:
96         max_log_file_size: "100MB"
97         max_total_log_size: "5GB"
98         max_files: "50"
99         log_level: "warn"
100    index_indexing_slowlog_rolling:
101        max_log_file_size: "100MB"
102        max_total_log_size: "5GB"
103        max_files: "50"
104        log_level: "warn"
105    # By default, is commented. Should be uncommented if ansible_
↳computes badly vCPUs number ; values are associated vCPUs numbers ;
↳please adapt to your configuration
106    # thread_pool:
107        # index:
108            # size: 2
109        # get:
110            # size: 2
111        # search:
112            # size: 2
113        # write:
114            # size: 2
115        # warmer:
116            # max: 2
117
118    mongoddb:
119        mongos_port: 27017
120        mongoc_port: 27018
121        mongod_port: 27019
122        mongo_authentication: "true"
123        host: "mongos.service.{{ consul_domain }}"
124        check_consul: 10 # in seconds
125        drop_info_log: false # Drop mongo (I)nformational log, for Verbosity_
↳Level of 0
126        # logs configuration
127        logrotate: enabled # or disabled
128        history_days: 30 # How many days to store logs if logrotate is set to
↳'enabled'
129
130    logstash:
131        host: "logstash.service.{{ consul_domain }}"
132        user: logstash
133        port: 10514
134        rest_port: 20514
135        check_consul: 10 # in seconds
136        # logstash xms & xmx in Megabytes
137        # jvm_xms: 2048
138        # jvm_xmx: 2048
139        # workers_number: 4
140        log_appenders:

```

(suite sur la page suivante)


```

141     rolling:
142         max_log_file_size: "100MB"
143         max_total_log_size: "5GB"
144     json_rolling:
145         max_log_file_size: "100MB"
146         max_total_log_size: "5GB"
147
148 # Prometheus params
149 prometheus:
150     metrics_path: /admin/v1/metrics
151     check_consul: 10 # in seconds
152     prometheus_config_file_target_directory: # Set path where "prometheus.yml
↳ " file will be generated. Example: /tmp/
153     server:
154         port: 9090
155         tsdb_retention_time: "7d"
156         tsdb_retention_size: "5GB"
157     node_exporter:
158         enabled: true
159         port: 9101
160         metrics_path: /metrics
161         log_level: "warn"
162         logrotate: enabled # or disabled
163         history_days: 30 # How many days to store logs if logrotate is set_
↳to 'enabled'
164     consul_exporter:
165         enabled: true
166         port: 9107
167         metrics_path: /metrics
168     elasticsearch_exporter:
169         enabled: true
170         port: 9114
171         metrics_path: /metrics
172         log_level: "warn"
173         logrotate: enabled # or disabled
174         history_days: 30 # How many days to store logs if logrotate is set_
↳to 'enabled'
175     alertmanager:
176         api_port: 9093
177         cluster_port: 9094
178         #receivers: # https://grafana.com/blog/2020/02/25/step-by-step-guide-
↳to-setting-up-prometheus-alertmanager-with-slack-pagerduty-and-gmail/
179         #- name: "slack_alert"
180         # slack_configs:
181         #   - api_url: "https://hooks.slack.com/services/xxxxxxx/
↳xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx"
182         # channel: '#your_alert_channel'
183         # send_resolved: true
184
185 grafana:
186     check_consul: 10 # in seconds
187     http_port: 3000
188     proxy: false
189     grafana_datasources:
190     - name: "Prometheus"
191       type: "prometheus"
192       access: "proxy"

```

(suite sur la page suivante)

(suite de la page précédente)

```

193     url: "http://prometheus-server.service.{{ consul_domain }}:{{
↳prometheus.server.port | default(9090) }}/prometheus"
194     basicAuth: false
195     editable: true
196     - name: "Prometheus AlertManager"
197       type: "camptocamp-prometheus-alertmanager-datasource"
198       access: "proxy"
199       url: "http://prometheus-alertmanager.service.{{ consul_domain }}:{{
↳prometheus.alertmanager.api_port | default(9093) }}"
200     basicAuth: false
201     editable: true
202     jsonData:
203       keepCookies: []
204       severity_critical: "4"
205       severity_high: "3"
206       severity_warning: "2"
207       severity_info: "1"
208     grafana_dashboards:
209     - name: 'vitam-dashboard'
210       orgId: 1
211       folder: ''
212       folderUid: ''
213       type: file
214       disableDeletion: false
215       updateIntervalSeconds: 10
216       allowUiUpdates: true
217       options:
218         path: "/etc/grafana/provisioning/dashboards"
219
220 # Curator units: days
221 curator:
222   log:
223     metricbeat:
224       close: 5
225       delete: 10
226     packetbeat:
227       close: 5
228       delete: 10
229
230 kibana:
231   header_value: "reporting"
232   import_delay: 10
233   import_retries: 10
234   # logs configuration
235   logrotate: enabled # or disabled
236   history_days: 30 # How many days to store logs if logrotate is set to
↳'enabled'
237   log:
238     baseuri: "kibana_log"
239     api_call_timeout: 120
240     groupe: "log"
241     port: 5601
242     default_index_pattern: "logstash-vitam*"
243     check_consul: 10 # in seconds
244     # default shards & replica
245     shards: 1
246     replica: 1

```

(suite sur la page suivante)

```

247     # pour index logstash-*
248     metrics:
249         shards: 1
250         replica: 1
251     # pour index metricbeat-*
252     metricbeat:
253         shards: 3 # must be a factor of 30
254         replica: 1
255     data:
256         baseuri: "kibana_data"
257         # OMA : bugdette : api_call_timeout is used for retries ; should_
↳ceate a separate variable rather than this one
258         api_call_timeout: 120
259         groupe: "data"
260         port: 5601
261         default_index_pattern: "logbookoperation_*"
262         check_consul: 10 # in seconds
263         # index template for .kibana
264         shards: 1
265         replica: 1
266
267     syslog:
268         # value can be syslog-ng or rsyslog
269         name: "rsyslog"
270
271     cerebro:
272         baseuri: "cerebro"
273         port: 9000
274         check_consul: 10 # in seconds
275         # logs configuration
276         logrotate: enabled # or disabled
277         history_days: 30 # How many days to store logs if logrotate is set to
↳'enabled'
278
279     siegfried:
280         port: 19000
281         consul_check: 10 # in seconds
282
283     clamav:
284         port: 3310
285         # logs configuration
286         logrotate: enabled # or disabled
287         history_days: 30 # How many days to store logs if logrotate is set to
↳'enabled'
288         freshclam:
289             # frequency freshclam for database update per day (from 0 to 24 - 24_
↳meaning hourly check)
290             db_update_periodicity: 1
291             private_mirror_address:
292             use_proxy: "no"
293
294     ## Avast Business Antivirus for Linux
295     ## if undefined, the following default values are applied.
296     # avast:
297     #     # logs configuration
298     #     logrotate: enabled # or disabled
299     #     history_days: 30 # How many days to store logs if logrotate is set to
↳'enabled'

```

(suite sur la page suivante)

(suite de la page précédente)

```

300 #   manage_repository: true
301 #   repository:
302 #       state: present
303 #       # For CentOS
304 #       baseurl: http://rpm.avast.com/lin/repo/dists/rhel/release
305 #       gpgcheck: no
306 #       proxy: _none_
307 #       # For Debian
308 #       baseurl: 'deb http://deb.avast.com/lin/repo debian-buster release'
309 #   vps_repository: http://linux-av.u.avcdn.net/linux-av/avast/x86_64
310 #   ## List of sha256 hash of excluded files from antivirus. Useful for
↪test environments.
311 #   whitelist:
312 #       - xxxxxx
313 #       - yyyyyy
314
315 mongo_express:
316     baseuri: "mongo-express"
317
318 ldap_authentication:
319     ldap_protocol: "ldap"
320     ldap_server: "{% if groups['ldap']|length > 0 %}{% groups['ldap']|first }
↪}{% endif %}"
321     ldap_port: "389"
322     ldap_base: "dc=programmevitam,dc=fr"
323     ldap_login: "cn=Manager,dc=programmevitam,dc=fr"
324     uid_field: "uid"
325     ldap_userDn_Template: "uid={0},ou=people,dc=programmevitam,dc=fr"
326     ldap_group_request: "(&(objectClass=groupOfNames)(member={0}))"
327     ldap_admin_group: "cn=admin,ou=groups,dc=programmevitam,dc=fr"
328     ldap_user_group: "cn=user,ou=groups,dc=programmevitam,dc=fr"
329     ldap_guest_group: "cn=guest,ou=groups,dc=programmevitam,dc=fr"
330
331 # Backup tool on storage-offer
332 restic:
333     snapshot_retention: 30 # number of snapshots to keep
334     # default run backup at 23:00 everyday
335     cron:
336         minute: '00'
337         hour: '23'
338         day: '*'
339         month: '*'
340         weekday: '*'
341     # [hosts_storage_offer_default] must be able to connect to the listed
↪databases below to properly backup.
342     backup:
343         # mongo-offer
344         - name: "{{ offer_conf }}"
345           type: mongod
346           host: "{{ offer_conf }}-mongos.service.consul:{{ mongod.mongos_
↪port }}"
347           user: "{{ mongod[offer_conf].admin.user }}"
348           password: "{{ mongod[offer_conf].admin.password }}"
349           # # mongo-data (only if mono-sharded cluster)
350           # - name: mongo-data
351           #   type: mongod
352           #   host: "mongo-data-mongos.service.consul:{{ mongod.mongos_port }}"
↪"

```

(suite sur la page suivante)

(suite de la page précédente)

```

353     # user: "{{ mongodb['mongo-data'].admin.user }}"
354     # password: "{{ mongodb['mongo-data'].admin.password }}"
355     # # mongo-vitamui (only if vitamui is deployed)
356     # - name: mongo-vitamui
357     # type: mongodb
358     # host: mongo-vitamui-mongod.service.consul:{{ mongodb.mongod_port_
↪}}
359     # # Add the following params on environments/group_vars/all/main/
↪vault-vitam.yml
360     # # They can be found under vitamui's deployment sources on_
↪environments/group_vars/all/vault-mongod.yml
361     # user: "{{ mongodb['mongo-vitamui'].admin.user }}"
362     # password: "{{ mongodb['mongo-vitamui'].admin.password }}"

```

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]`.

4.2.3.2.6 Fichier `tenants_vars.yml`

Le fichier `repertoire_inventories/group_vars/all/advanced/tenants_vars.yml` permet de gérer les configurations spécifiques associés aux tenants de la plateforme (liste des tenants, regroupement de tenants, configuration du nombre de shards et replicas, etc. ...).

```

1  ### tenants ###
2  # List of dead / removed tenants that should never be reused / present in_
↪vitam_tenant_ids
3  vitam_removed_tenants: []
4  # Administration tenant
5  vitam_tenant_admin: 1
6
7  ###
8  # Elasticsearch tenant indexation
9  # =====
10 #
11 # Elastic search index configuration settings :
12 # - 'number_of_shards' : number of shards per index. Every ES shard is_
↪stored as a lucene index.
13 # - 'number_of_replicas': number of additional copies of primary shards
14 # The total number of shards : number_of_shards * (1 primary + M number_of_
↪replicas)
15 #
16 # CAUTION : The total number of shards should be lower than or equal to the_
↪number of elasticsearch-data instances in the cluster
17 #
18 # Default settings should be okay for most use cases.
19 # For more data-intensive workloads or deployments with high number of_
↪tenants, custom tenant and/or collection configuration might be specified.
20 #

```

(suite sur la page suivante)

(suite de la page précédente)

```

21 # Tenant list may be specified as :
22 # - A specific tenant                               : eg.
   ↪ '1'
23 # - A tenant range                                 : eg.
   ↪ '10-19'
24 # - A comma-separated combination of specific tenants & tenant ranges : eg.
   ↪ '1, 5, 10-19, 50-59'
25 #
26 # Masterdata collections (accesscontract, filerules...) are indexed as
   ↪ single elasticsearch indexes :
27 # - Index name format : {collection}_{date_time_of_creation}. e.g.
   ↪ accesscontract_20200415_042011
28 # - Index alias name : {collection}. e.g. accesscontract
29 #
30 # Metadata collections (unit & objectgroup), and logbook operation
   ↪ collections are stored on a per-tenant index basis :
31 # - Index name           : {collection}_{tenant}_{date_time_of_creation}. e.g.
   ↪ unit_1_20200517_025041
32 # - Index alias name : {collection}_{tenant}. e.g. unit_1
33 #
34 # Very small tenants (1-100K entries) may be grouped in a "tenant group",
   ↪ and hence, stored in a single elasticsearch index.
35 # This allows reducing the number of indexes & shards that the elasticsearch
   ↪ cluster need to manage :
36 # - Index name           : {collection}_{tenant_group_name}_{date_time_of_
   ↪ creation}. e.g. logbookoperation_grp5_20200517_025041
37 # - Index alias name : {collection}_{tenant_group_name}. e.g.
   ↪ logbookoperation_grp5
38 #
39 # Tenant list can be wide ranges (eg: 100-199), and may contain non-existing
   ↪ (yet) tenants. i.e. tenant lists might be wider that 'vitam_tenant_ids'
   ↪ section
40 # This allows specifying predefined tenant families (whether normal tenants
   ↪ ranges, or tenant groups) to which tenants can be added in the future.
41 # However, tenant lists may not intersect (i.e. a single tenant cannot
   ↪ belong to 2 configuration sections).
42 #
43 # Sizing recommendations :
44 # - 1 shard per 5-10M records for small documents (eg. masterdata
   ↪ collections)
45 # - 1 shard per 1-2M records for larger documents (eg. metadata & logbook
   ↪ collections)
46 # - As a general rule, shard size should not exceed 30GB per shard
47 # - A single ES node should not handle > 200 shards (be it a primary or a
   ↪ replica)
48 # - It is recommended to start small and add more shards when needed (re-
   ↪ sharding requires a re-indexation operation)
49 #
50 # /\ IMPORTANT :
51 # Changing the configuration of an existing tenant requires re-indexation of
   ↪ the tenants and/or tenant groups
52 #
53 # Please refer to documentation for more details.
54 #
55 ###
56 vitam_elasticsearch_tenant_indexation:
57

```

(suite sur la page suivante)

```

58   ###
59   # Default masterdata collection indexation settings (default_config_
↳section) apply for all master data collections
60   # Custom settings can be defined for the following masterdata collections:
61   #   - accesscontract
62   #   - accessionregisterdetail
63   #   - accessionregistersummary
64   #   - accessionregistersymbolic
65   #   - agencies
66   #   - archiveunitprofile
67   #   - context
68   #   - fileformat
69   #   - filerules
70   #   - griffin
71   #   - ingestcontract
72   #   - managementcontract
73   #   - ontology
74   #   - preservationscenario
75   #   - profile
76   #   - securityprofile
77   ###
78   masterdata:
79   # {collection}:
80   #   number_of_shards: 1
81   #   number_of_replicas: 2
82   #   ...
83
84
85   ###
86   # Custom index settings for regular tenants.
87   ###
88   dedicated_tenants:
89   #   - tenants: '1, 3, 11-20'
90   #   unit:
91   #     number_of_shards: 4
92   #     number_of_replicas: 0
93   #   objectgroup:
94   #     number_of_shards: 5
95   #     number_of_replicas: 0
96   #   logbookoperation:
97   #     number_of_shards: 3
98   #     number_of_replicas: 0
99   #   ...
100
101
102
103
104   ###
105   # Custom index settings for grouped tenants.
106   # Group name must meet the following criteria:
107   #   - alphanumeric characters
108   #   - lowercase only
109   #   - not start with a number
110   #   - be less than 64 characters long.
111   #   - NO special characters - / _ | ...
112   ###
113   grouped_tenants:

```

(suite sur la page suivante)

(suite de la page précédente)

```

114 # - name: 'grp1'
115 #   tenants: '5-10'
116 #   unit:
117 #     number_of_shards: 5
118 #     number_of_replicas: 0
119 #   objectgroup:
120 #     number_of_shards: 6
121 #     number_of_replicas: 0
122 #   logbookoperation:
123 #     number_of_shards: 7
124 #     number_of_replicas: 0
125 #   ...
126
127 extendedConfiguration:
128   default:
129     eliminationReportExtraFields: [ ]
130     objectGroupBlackListedFields: ['Filename']
131   custom:
132     # The 'eliminationReportExtraFields' configuration option specifies the
133     ↪ metadata keys that should be included in the report when performing an
134     ↪ elimination.
135     # It determines which additional metadata fields should be retained and
136     ↪ displayed in the elimination report.
137     # You can include any of the following extra fields: "#id", "#version", "
138     ↪ #unitups", "#originating_agency", "#approximate_creation_date", "
139     ↪ #approximate_update_date", "FilePlanPosition", "SystemId",
140     ↪ "OriginatingSystemId", "ArchivalAgencyArchiveUnitIdentifier",
141     ↪ "OriginatingAgencyArchiveUnitIdentifier",
142     ↪ "TransferringAgencyArchiveUnitIdentifier"
143     #
144     # The 'objectGroupBlackListedFields' configuration option specifies the
145     ↪ fields that should not be reported by access-external.
146     #
147     # Example for tenant 0 :
148     #   0:
149     #     eliminationReportExtraFields: ["#id", "FilePlanPosition", "SystemId"]
150     #     objectGroupBlackListedFields: ['Filename']

```

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

Voir aussi :

Se référer au chapitre « Gestion des indexes Elasticsearch dans un contexte massivement multi-tenants » du *DEX* pour plus d'informations sur cette fonctionnalité.

Avvertissement : Attention, en cas de modification de la distribution des tenants, une procédure de réindexation de la base elasticsearch-data est nécessaire. Cette procédure est à la charge de l'exploitation et nécessite un arrêt de service sur la plateforme. La durée d'exécution de cette réindexation dépend de la quantité de données à traiter.

Voir aussi :

Se référer au chapitre « Réindexation » du *DEX* pour plus d'informations.

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/main/vault-vitam.yml`
- `environments/group_vars/all/main/vault-keystores.yml`
- `environments/group_vars/all/main/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 `repertoire_inventory|'group_vars/all/main/vault-vitam.yml'` contient les secrets généraux :

```
1 ---
2 # Vitam platform secret key
3 # Note: It has to be the same on all sites
4 plateforme_secret: change_it_vitamsecret
5
6 # The consul key must be 16-bytes, Base64 encoded: https://www.consul.io/docs/
7 ↪agent/encryption.html
8 # You can generate it with the "consul keygen" command
9 # Or you can use this script: deployment/pki/scripts/generate_consul_key.sh
10 # Note: It has to be the same on all sites
11 consul_encrypt: Biz14ohqN4HtvZmrXp3N4A==
12 mongodb:
```

(suite sur la page suivante)

(suite de la page précédente)

```

13 mongo-data:
14   passphrase: changeitkM4L6zBgK527tWBb
15   admin:
16     user: vitamdb-admin
17     password: change_it_1MpG22m2MywvKW5E
18   localadmin:
19     user: vitamdb-localadmin
20     password: change_it_HycFEVD74g397iRe
21   system:
22     user: vitamdb-system
23     password: change_it_HycFEVD74g397iRe
24   metadata:
25     user: metadata
26     password: change_it_37b97KVADV8YbCwt
27   logbook:
28     user: logbook
29     password: change_it_jVi6q8eX4H1Ce8UC
30   report:
31     user: report
32     password: change_it_jb7TASZbU6n85t8L
33   functionalAdmin:
34     user: functional-admin
35     password: change_it_9eA2zMCL6tm6KF1e
36   securityInternal:
37     user: security-internal
38     password: change_it_m39XvRQWixyDX566
39   scheduler:
40     user: scheduler
41     password: change_it_Q8WEdxhXXOe2NEhp
42   collect:
43     user: collect
44     password: change_it_m39XvRQWixyDX566
45   metadataCollect:
46     user: metadata-collect
47     password: change_it_37b97KVADV8YbCwt
48   offer-fs-1:
49     passphrase: changeitmB5rnk1M5TY61PqZ
50     admin:
51       user: vitamdb-admin
52       password: change_it_FLkM5emt63N73EcN
53     localadmin:
54       user: vitamdb-localadmin
55       password: change_it_QeH8q4e16ah4QKXS
56     system:
57       user: vitamdb-system
58       password: change_it_HycFEVD74g397iRe
59     offer:
60       user: offer
61       password: change_it_pQilT1yT9LAF8au8
62   offer-fs-2:
63     passphrase: changeiteSY1By57qZr4MX2s
64     admin:
65       user: vitamdb-admin
66       password: change_it_84aTMFZ7h8e2NgMe
67     localadmin:
68       user: vitamdb-localadmin
69     password: change_it_AmlB37tGY1w5VfvX

```

(suite sur la page suivante)

```
70 system:
71   user: vitamdb-system
72   password: change_it_HycFEVD74g397iRe
73 offer:
74   user: offer
75   password: change_it_mLDYds957sNQ53mA
76 offer-tape-1:
77   passphrase: changeitmB5rnk1M5TY61PqZ
78   admin:
79     user: vitamdb-admin
80     password: change_it_FLkM5emt63N73EcN
81 localadmin:
82   user: vitamdb-localadmin
83   password: change_it_QeH8q4e16ah4QKXS
84 system:
85   user: vitamdb-system
86   password: change_it_HycFEVD74g397iRe
87 offer:
88   user: offer
89   password: change_it_pQilT1yT9LAF8au8
90 offer-swift-1:
91   passphrase: changeitgYvt42M2pKL6Zx3T
92   admin:
93     user: vitamdb-admin
94     password: change_it_e21hLp51WNa4sJFS
95 localadmin:
96   user: vitamdb-localadmin
97   password: change_it_QB8857SJrGrQh2yu
98 system:
99   user: vitamdb-system
100  password: change_it_HycFEVD74g397iRe
101 offer:
102   user: offer
103   password: change_it_AWJg2Bp3s69P6nMe
104 offer-s3-1:
105   passphrase: changeituF1jVdR9NqdTG625
106   admin:
107     user: vitamdb-admin
108     password: change_it_5b7cSWcS5M1NF4kv
109 localadmin:
110   user: vitamdb-localadmin
111   password: change_it_S9jE24rxHwUZP6y5
112 system:
113   user: vitamdb-system
114   password: change_it_HycFEVD74g397iRe
115 offer:
116   user: offer
117   password: change_it_TuTB1i2k7iQW3zL2
118 offer-tape-1:
119   passphrase: changeituF1jghT9NqdTG625
120   admin:
121     user: vitamdb-admin
122     password: change_it_5b7cSWcab91NF4kv
123 localadmin:
124   user: vitamdb-localadmin
125   password: change_it_S9jE24rxHwUZP5a6
126 system:
```

(suite de la page précédente)

```

127     user: vitamdb-system
128     password: change_it_HycFEVD74g397iRe
129 offer:
130     user: offer
131     password: change_it_TuTB1i2k7iQW3c2a
132
133 vitam_users:
134   - vitam_aadmin:
135     login: aadmin
136     password: change_it_z5MP7GC4qnR8nL9t
137     role: admin
138   - vitam_user:
139     login: user
140     password: change_it_w94Q3jPAT2aJYm8b
141     role: user
142   - vitam_gguest:
143     login: gguest
144     password: change_it_E5v7Tr4h6tYaQG2W
145     role: guest
146   - techadmin:
147     login: techadmin
148     password: change_it_K29E1uHcPZ8zXji8
149     role: admin
150
151 ldap_authentication:
152     ldap_pwd: "change_it_t69Rn5NdUv39EYkC"
153
154 admin_basic_auth_password: change_it_5Yn74JgXwbQ9KdP8
155
156 vitam_offers:
157   offer-swift-1:
158     swiftUser: swift_user
159     swiftPassword: password_change_m44j57aYeRPnPXQ2
160   offer-s3-1:
161     s3AccessKey: accessKey_change_grLS8372Uga5EJSx
162     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/advanced/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'access 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 `repertoire_inventory|group_vars/all/main/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: changeit6kQ16eyDY7QPS9fy
9     collect_external: changeit6kQ16eyDYAoPS9fy
10  client_external:
11    ihm_demo: changeitGT38hhTiA32x1PLy
12    gatling: changeit2sBC5ac7NfGF9Qj7
13    ihm_recette: changeitdAZ9Eq65UhDZd9p4
14    reverse: changeite5XTzb5yVPcEX464
15    vitam_admin_int: changeitz6xZe5gDu7nhDZd9
16    collect_external: changeitz6xZe5gDu7nhDZA12
17  client_storage:
18    storage: changeit647D7LWiyM6qYMnm
19  timestamping:
20    secure_logbook: changeitMn9Skuyx87VYU62U
21    secure_storage: changeite5gDu9Skuy84BLW9
22  truststores:
23    server: changeitxNe4JLfn528PVHj7
24    client_external: changeitJ2eS93DcPH1v4jAp
25    client_storage: changeitHpSCa31aG8ttB87S
26  grantedstores:
27    client_external: changeitLL22HkmDCA2e2vj7
28    client_storage: changeitR3wwp5C8KQS76Vcu
```

Avertissement : Il convient de sécuriser votre environnement en définissant des mots de passe forts.

4.2.3.3.2 Cas des extras

- Le fichier `repertoire_inventory|group_vars/all/main/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 le répertoire `inventory/group_vars/all` commençant par `vault-` doivent être protégés (chiffrés) avec l'utilitaire `ansible-vault`.

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

4.2.3.3.3.1 Générer des fichiers *vaultés* depuis des fichiers en clair

Exemple du fichier `vault-cots.yml`

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

4.2.3.3.3.2 Re-chiffrer un fichier *vaulté* avec un nouveau mot de passe

Exemple du fichier `vault-cots.yml`

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

4.2.3.4 Le mapping Elasticsearch pour Unit et ObjectGroup

Les mappings des indexes elasticsearch pour les collections masterdata Unit et ObjectGroup sont configurables de l'extérieur, plus spécifiquement dans le dossier `inventory/deployment/ansible-vitam/roles/elasticsearch-mapping/files/`, ce dossier contient :

- `deployment/ansible-vitam/roles/elasticsearch-mapping/files/unit-es-mapping.json`
- `deployment/ansible-vitam/roles/elasticsearch-mapping/files/og-es-mapping.json`

Exemple du fichier mapping de la collection ObjectGroup :

```

1  {
2    "dynamic_templates": [
3      {
4        "object": {
5          "match_mapping_type": "object",
6          "mapping": {
7            "type": "object"
8          }
9        }
10     },
11     {
12       "all_string": {
13         "match": "*",
14         "mapping": {
15           "type": "text"
16         }
17       }
18     }
19   }

```

(suite sur la page suivante)

```
19 ],
20 "properties": {
21   "FileInfo": {
22     "properties": {
23       "CreatingApplicationName": {
24         "type": "text"
25       },
26       "CreatingApplicationVersion": {
27         "type": "text"
28       },
29       "CreatingOs": {
30         "type": "text"
31       },
32       "CreatingOsVersion": {
33         "type": "text"
34       },
35       "DateCreatedByApplication": {
36         "type": "date",
37         "format": "strict_date_optional_time"
38       },
39       "Filename": {
40         "type": "text"
41       },
42       "LastModified": {
43         "type": "date",
44         "format": "strict_date_optional_time"
45       }
46     }
47   },
48   "Metadata": {
49     "properties": {
50       "Text": {
51         "type": "object"
52       },
53       "Document": {
54         "type": "object"
55       },
56       "Image": {
57         "type": "object"
58       },
59       "Audio": {
60         "type": "object"
61       },
62       "Video": {
63         "type": "object"
64       }
65     }
66   },
67   "OtherMetadata": {
68     "type": "object",
69     "properties": {
70       "RawMetadata": {
71         "type": "object"
72       }
73     }
74   },
75   "_profil": {
```

(suite sur la page suivante)

(suite de la page précédente)

```

76     "type": "keyword"
77   },
78   "_qualifiers": {
79     "properties": {
80       "_nbc": {
81         "type": "long"
82       },
83       "qualifier": {
84         "type": "keyword"
85       },
86       "versions": {
87         "type": "nested",
88         "properties": {
89           "Compressed": {
90             "type": "text"
91           },
92           "DataObjectGroupId": {
93             "type": "keyword"
94           },
95           "DataObjectVersion": {
96             "type": "keyword"
97           },
98           "DataObjectProfile": {
99             "type": "keyword"
100          },
101          "DataObjectSystemId": {
102            "type": "keyword"
103          },
104          "DataObjectGroupSystemId": {
105            "type": "keyword"
106          },
107          "_opi": {
108            "type": "keyword"
109          },
110          "FileInfo": {
111            "properties": {
112              "CreatingApplicationName": {
113                "type": "text"
114              },
115              "CreatingApplicationVersion": {
116                "type": "text"
117              },
118              "CreatingOs": {
119                "type": "text"
120              },
121              "CreatingOsVersion": {
122                "type": "text"
123              },
124              "DateCreatedByApplication": {
125                "type": "date",
126                "format": "strict_date_optional_time"
127              },
128              "Filename": {
129                "type": "text"
130              },
131              "LastModified": {
132                "type": "date",

```

(suite sur la page suivante)


```
133         "format": "strict_date_optional_time"
134     }
135 }
136 },
137 "FormatIdentification": {
138     "properties": {
139         "FormatId": {
140             "type": "keyword"
141         },
142         "FormatLiteral": {
143             "type": "keyword"
144         },
145         "MimeType": {
146             "type": "keyword"
147         },
148         "Encoding": {
149             "type": "keyword"
150         }
151     }
152 },
153 "MessageDigest": {
154     "type": "keyword"
155 },
156 "Algorithm": {
157     "type": "keyword"
158 },
159 "PhysicalDimensions": {
160     "properties": {
161         "Diameter": {
162             "properties": {
163                 "unit": {
164                     "type": "keyword"
165                 },
166                 "dValue": {
167                     "type": "double"
168                 }
169             }
170         },
171         "Height": {
172             "properties": {
173                 "unit": {
174                     "type": "keyword"
175                 },
176                 "dValue": {
177                     "type": "double"
178                 }
179             }
180         },
181         "Depth": {
182             "properties": {
183                 "unit": {
184                     "type": "keyword"
185                 },
186                 "dValue": {
187                     "type": "double"
188                 }
189             }
190         }
191     }
192 }
```

(suite sur la page suivante)

(suite de la page précédente)

```
190     },
191     "Shape": {
192       "type": "keyword"
193     },
194     "Thickness": {
195       "properties": {
196         "unit": {
197           "type": "keyword"
198         },
199         "dValue": {
200           "type": "double"
201         }
202       }
203     },
204     "Length": {
205       "properties": {
206         "unit": {
207           "type": "keyword"
208         },
209         "dValue": {
210           "type": "double"
211         }
212       }
213     },
214     "NumberOfPage": {
215       "type": "long"
216     },
217     "Weight": {
218       "properties": {
219         "unit": {
220           "type": "keyword"
221         },
222         "dValue": {
223           "type": "double"
224         }
225       }
226     },
227     "Width": {
228       "properties": {
229         "unit": {
230           "type": "keyword"
231         },
232         "dValue": {
233           "type": "double"
234         }
235       }
236     }
237   }
238 },
239 "PhysicalId": {
240   "type": "keyword"
241 },
242 "Size": {
243   "type": "long"
244 },
245 "Uri": {
246   "type": "keyword"
```

(suite sur la page suivante)

```
247     },
248     "_id": {
249       "type": "keyword"
250     },
251     "_storage": {
252       "properties": {
253         "_nbc": {
254           "type": "long"
255         },
256         "offerIds": {
257           "type": "keyword"
258         },
259         "strategyId": {
260           "type": "keyword"
261         }
262       }
263     },
264     "PersistentIdentifier": {
265       "properties": {
266         "PersistentIdentifierType": {
267           "type": "keyword"
268         },
269         "PersistentIdentifierOrigin": {
270           "type": "keyword"
271         },
272         "PersistentIdentifierReference": {
273           "type": "keyword"
274         },
275         "PersistentIdentifierContent": {
276           "type": "keyword"
277         }
278       }
279     },
280     "DataObjectUse": {
281       "type": "keyword"
282     },
283     "DataObjectNumber": {
284       "type": "long"
285     }
286   }
287 }
288 }
289 },
290 "_v": {
291   "type": "long"
292 },
293 "_av": {
294   "type": "long"
295 },
296 "_nbc": {
297   "type": "long"
298 },
299 "_ops": {
300   "type": "keyword"
301 },
302 "_opi": {
303   "type": "keyword"
```

(suite sur la page suivante)

(suite de la page précédente)

```

304     },
305     "_sp": {
306         "type": "keyword"
307     },
308     "_sps": {
309         "type": "keyword"
310     },
311     "_tenant": {
312         "type": "long"
313     },
314     "_up": {
315         "type": "keyword"
316     },
317     "_uds": {
318         "type": "object",
319         "enabled": false
320     },
321     "_us": {
322         "type": "keyword"
323     },
324     "_storage": {
325         "properties": {
326             "_nbc": {
327                 "type": "long"
328             },
329             "offerIds": {
330                 "type": "keyword"
331             },
332             "strategyId": {
333                 "type": "keyword"
334             }
335         }
336     },
337     "_glpd": {
338         "enabled": false
339     },
340     "_acd": {
341         "type": "date",
342         "format": "strict_date_optional_time"
343     },
344     "_aud": {
345         "type": "date",
346         "format": "strict_date_optional_time"
347     }
348 }
349 }

```

Note : Le paramétrage de ce mapping se fait sur les deux composants `metadata` et le composant extra `ihm-recette`.

Prudence : En cas de changement du mapping, il faut veiller à ce que cette mise à jour soit en accord avec l'Ontologie de *VITAM*.

Le mapping est pris en compte lors de la première création des indexes. Pour une nouvelle installation de *VITAM*, les mapping seront automatiquement pris en compte. Cependant, la modification des mappings nécessite

une réindexation via l'API dédiée si VITAM est déjà installé.

4.2.4 Gestion des certificats

Une vue d'ensemble de la gestion des certificats est présentée *dans l'annexe dédiée* (page 122).

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 plateforme* (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/advanced/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.

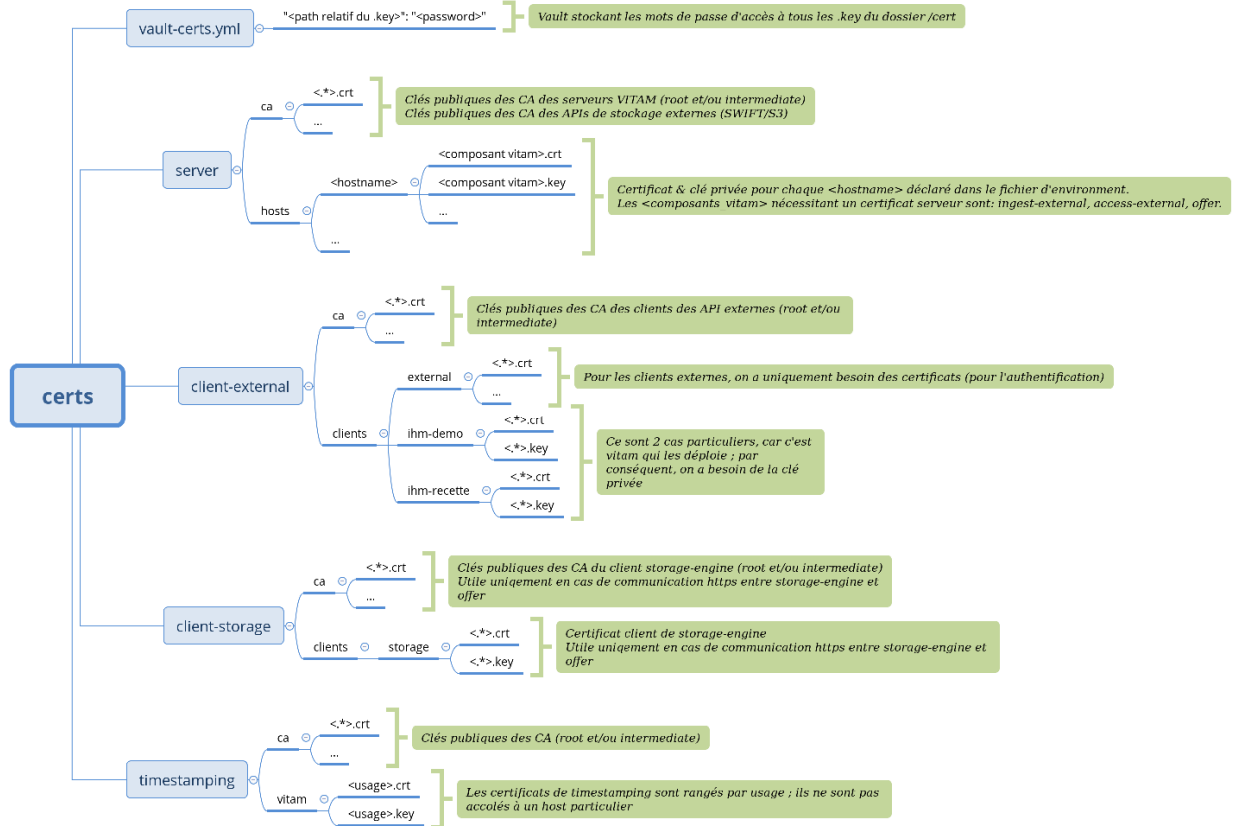


FIG. 3 – Vue détaillée de l'arborescence des certificats

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)¹⁴.

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`).

14. 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/advanced/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/advanced/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/advanced/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/main/vault-keystores.yml`, décrit dans la section *Déclaration des secrets* (page 52).

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/main/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 `deployment/environments/group_vars/all/main/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 griffon 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 `deployment/environments/group_vars/all/main/main.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
vitam-ffmpeg-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.

Éditer le fichier `deployment/environments/group_vars/all/advanced/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 accesslog

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

Éditer le fichier `deployment/environments/group_vars/all/advanced/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-external)

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

- Éditer la variable `vitam.ingestexternal.antivirus` dans le fichier `deployment/environments/group_vars/all/advanced/vitam_vars.yml` pour indiquer le nom de l'antivirus à utiliser.
- Créer un script shell (dont l'extension doit être `.sh`) sous `environments/antivirus/` (norme : `scan-<vitam.ingestexternal.antivirus>.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 : Il subsiste une limitation avec l'antivirus ClamAV qui n'est actuellement pas capable de scanner des fichiers > 4Go. Ainsi, il n'est pas recommandé de conserver cet antivirus en environnement de production.

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` ou de renseigner un [miroir local privé](#)¹⁵).

4.2.5.4.1 Extra : Avast Business Antivirus for Linux

Note : Avast étant un logiciel soumis à licence, Vitam ne fournit pas de support ni de licence nécessaire à l'utilisation de Avast Antivirus for Linux.

Vous trouverez plus d'informations sur le site officiel : [Avast Business Antivirus for Linux](#)¹⁶

À la place de clamAV, il est possible de déployer l'antivirus **Avast Business Antivirus for Linux**.

Pour se faire, il suffit d'éditer la variable `vitam.ingestexternal.antivirus: avast` dans le fichier `deployment/environments/group_vars/all/advanced/vitam_vars.yml`.

Il sera nécessaire de fournir le fichier de licence sous `deployment/environments/antivirus/license.avastlic` pour pouvoir déployer et utiliser l'antivirus Avast.

De plus, il est possible de paramétrer l'accès aux repositories (Packages & Virus definitions database) dans le fichier `deployment/environments/group_vars/all/advanced/cots_vars.yml`.

Si les paramètres ne sont pas définis, les valeurs suivantes sont appliquées par défaut.

15. <https://www.clamav.net/documents/private-local-mirrors>

16. <https://www.avast.com/fr-fr/business/products/linux-antivirus>

```
## Avast Business Antivirus for Linux
## if undefined, the following default values are applied.
avast:
  # logs configuration
  logrotate: enabled # or disabled
  history_days: 30 # How many days to store logs if logrotate is set to 'enabled'
  manage_repository: true
  repository:
    state: present
    # For CentOS
    baseurl: http://rpm.avast.com/lin/repo/dists/rhel/release
    gpgcheck: no
    proxy: _none_
    # For Debian
    baseurl: 'deb http://deb.avast.com/lin/repo debian-buster release'
  vps_repository: http://linux-av.u.avcdn.net/linux-av/avast/x86_64
  ## List of sha256 hash of excluded files from antivirus. Useful for test_
↪environments.
  whitelist:
    - <EMPTY>
```

Avertissement : Vitam gère en entrée les SIPs aux formats : ZIP ou TAR (tar, tar.gz ou tar.bz2); cependant et d'après les tests effectués, il est fortement recommandé d'utiliser le format .zip pour bénéficier des meilleures performances d'analyses avec le scan-avast.sh.

De plus, il faudra prendre en compte un dimensionnement supplémentaire sur les ingest-external afin de pouvoir traiter le scan des fichiers >500Mo.

Dans le cas d'un SIP au format .zip ou .tar, les fichiers >500Mo contenus dans le SIP seront décompressés et scannés unitairement. Ainsi la taille utilisée ne dépassera pas la taille d'un fichier.

Dans le cas d'un SIP au format .tar.gz ou .tar.bz2, les SIPs >500Mo seront intégralement décompressés et scannés. Ainsi, la taille utilisée correspondra à la taille du SIP décompressé.

4.2.5.5 Paramétrage des certificats externes (*-externe)

Se reporter au chapitre dédié à la gestion des certificats : *Gestion des certificats* (page 64)

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/advanced/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/advanced/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/advanced/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 groupes :

- `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](#)¹⁷.

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).

17. <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/main/main.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 `deployment/environments/group_vars/all/advanced/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/advanced/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"
    StorageRule : "5 year"
    ReuseRule : "2 year"
```

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 Augmenter la précision sur le nombre de résultats retournés dépassant 10000

Suite à une évolution d'ElasticSearch (à partir de la version 7.6), le nombre maximum de résultats retournés est limité à 10000. Ceci afin de limiter la consommation de ressources sur le cluster elasticsearch.

Pour permettre de retourner le nombre exact de résultats, il est possible d'éditer le paramètre `vitam.accessexternal.authorizeTrackTotalHits` dans le fichier de configuration `environments/group_vars/all/vitam_vars.yml`

Il sera nécessaire de réappliquer la configuration sur le groupe `hosts_access_external` :

```
ansible-playbook ansible-vitam/vitam.yml --limit hosts_access_external --tags update_
↪vitam_configuration -i environments/hosts.<environnement> --ask-vault-pass
```


Ensuite, si l'API de recherche utilise le type d'entrée de DSL « SELECT_MULTIPLE », il faut ajouter `$track_total_hits : true` au niveau de la partie « filter » de la requête d'entrée.

Ci-dessous, un exemple de requête d'entrée :

```
{
  "$roots": [],
  "$query": [
    {
      "$match": {
        "Title": "héritage"
      }
    }
  ],
  "$filter": {
    "$offset": 0,
    "$limit": 100,
    "$track_total_hits": true
  },
  "$projection": {}
}
```

4.2.5.13 Fichiers complémentaires

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

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

```
1  ---
2
3  # TENANTS
4  # List of active tenants
5  vitam_tenant_ids: [0,1,2,3,4,5,6,7,8,9]
6  # For functional-administration, manage master/slave tenant configuration
7  # http://www.programmevitam.fr/ressources/DocCourante/html/installation/
8  ↪ installation/21-addons.html#passage-des-identifiants-des-referentiels-en-mode-
9  ↪ esclave
10 vitam_tenants_usage_external:
11   - name: 0
12     identifiants:
13       - INGEST_CONTRACT
14       - ACCESS_CONTRACT
15       - MANAGEMENT_CONTRACT
16       - ARCHIVE_UNIT_PROFILE
17   - name: 1
18     identifiants:
19       - INGEST_CONTRACT
20       - ACCESS_CONTRACT
21       - MANAGEMENT_CONTRACT
22       - PROFILE
23       - SECURITY_PROFILE
24       - CONTEXT
25
26 # GRIFFINS
27 # Vitam griffins required to launch preservation scenario
28 # Example:
```

(suite sur la page suivante)

(suite de la page précédente)

```

27 # vitam_griffins: ["vitam-imagemagick-griffin", "vitam-libreoffice-griffin",
↳"vitam-jhove-griffin", "vitam-odfvalidator-griffin", "vitam-siegfried-griffin",
↳"vitam-tesseract-griffin", "vitam-verapdf-griffin", "vitam-ffmpeg-griffin"]
28 vitam_griffins: []
29
30 # CONSUL
31 consul:
32   network: "ip_admin" # Which network to use for consul communications ? ip_admin_
↳or ip_service ?
33 consul_remote_sites:
34 # wan contains the wan addresses of the consul server instances of the external_
↳vitam sites
35 # Exemple, if our local dc is dc1, we will need to set dc2 & dc3 wan conf:
36 #   - dc2:
37 #     wan: ["10.10.10.10", "1.1.1.1"]
38 #   - dc3:
39 #     wan: ["10.10.10.11", "1.1.1.1"]
40
41 # LOGGING
42 # vitam_defaults:
43 #   access_retention_days: 30 # Number of days for file retention
44 #   access_total_size_cap: "10GB" # total acceptable size
45 #   logback_max_file_size: "10MB"
46 #   logback_total_size_cap:
47 #     file:
48 #       history_days: 30
49 #       totalsize: "5GB"
50 #     security:
51 #       history_days: 30
52 #       totalsize: "5GB"
53
54 # ELASTICSEARCH
55 # 'number_of_shards': number of shards per index, every ES shard is stored as a_
↳lucene index
56 # 'number_of_replicas': number of additional copies of primary shards
57 # Total number of shards: number_of_shards * (1 primary + M number_of_replicas)
58 # CAUTION: The total number of shards should be lower than or equal to the number_
↳of elasticsearch-data instances in the cluster
59 # More details in groups_vars/all/advanced/tenants_vars.yml file
60 vitam_elasticsearch_tenant_indexation:
61   default_config:
62     # Default settings for masterdata collections (1 index per collection)
63     masterdata:
64       number_of_shards: 1
65       number_of_replicas: 2
66     # Default settings for unit indexes (1 index per tenant)
67     unit:
68       number_of_shards: 1
69       number_of_replicas: 2
70     # Default settings for object group indexes (1 index per tenant)
71     objectgroup:
72       number_of_shards: 1
73       number_of_replicas: 2
74     # Default settings for logbook operation indexes (1 index per tenant)
75     logbookoperation:
76       number_of_shards: 1
77       number_of_replicas: 2

```

(suite sur la page suivante)

(suite de la page précédente)

```

78   # Default settings for collect_unit indexes
79   collect_unit:
80     number_of_shards: 1
81     number_of_replicas: 2
82   # Default settings for collect_objectgroup indexes
83   collect_objectgroup:
84     number_of_shards: 1
85     number_of_replicas: 2
86
87   collect_grouped_tenants:
88   - name: 'all'
89     # Group all tenants for collect's indexes (collect_unit & collect_objectgroup)
90     tenants: "{ { vitam_tenant_ids | join(',') }}"
91
92   elasticsearch:
93     log:
94       index_templates:
95         default:
96           shards: 1
97           replica: 1
98     data:
99       index_templates:
100        default:
101          shards: 1
102          replica: 2
103   curator:
104     log:
105       metrics:
106         close: 7
107         delete: 30
108       logstash:
109         close: 7
110         delete: 30
111
112   # PACKAGES
113   disable_internet_repositories_install: true # Disable EPEL or Debian backports_
↪repositories install

```

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.

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

```

1   ---
2   ### global ###
3
4   # Vitam deployment mode. Allowed values are :
5   # - "prod" (default): Enforces additional security checks (disallow development/
↪debug tools, reverse proxy does NOT forward traffic to vitam service ports...)
6   # - "dev" (NOT for sensitive / production environments): Allow development/debug_
↪tools, reverse proxy forwards traffic to vitam service ports.
7   deployment_mode: prod
8
9   # TODO MAYBE : permettre la surcharge avec une syntaxe du genre vitamopts.folder_
↪root | default(vitam_default.folder_root) dans les templates ?

```

(suite sur la page suivante)

(suite de la page précédente)

```

10 droid_filename: "DROID_SignatureFile_V109.xml"
11 droid_container_filename: "container-signature-20221102.xml"
12
13 # The global defaults parameters for vitam & vitam components
14 vitam_defaults:
15   folder:
16     root_path: /vitam
17     folder_permission: "0750"
18     conf_permission: "0440"
19     folder_upload_permission: "0770"
20     script_permission: "0750"
21   users:
22     vitam: "vitam"
23     vitamdb: "vitamdb"
24     group: "vitam"
25   services:
26     # Default log level for vitam components: logback values (TRACE, DEBUG, INFO,
↳WARN, ERROR, OFF)
27     log_level: WARN
28     start_timeout: 300
29     stop_timeout: 3600
30     port_service_timeout: 86400
31     api_call_timeout: 120
32     api_long_call_timeout: 300
33     status_retries_number: 60
34     status_retries_delay: 5
35     at_boot: false
36     ### Trust X-SSL-CLIENT-CERT header for external api auth ? true | false
↳(default)
37     # Should only be enabled when accessing to vitam externals through a Reverse
↳Proxy that does "SSL offloading"
38     # NGINX configuration           : proxy_set_header X-SSL-CLIENT-CERT $ssl_client
↳escaped_cert;
39     # Apache httpd configuration : RequestHeader set X-SSL-CLIENT-CERT "%{SSL_
↳CLIENT_CERT}s"
40     # Important : When enabled, special care must be taken to ensure firewall rules
↳are properly set to ensure only
41     #           reverse proxy can access vitam external applications through
↳their respective port_service to avoid
42     #           malicious header injection.
43     trust_client_certificate_header: false
44     ### Force chunk mode : set true if chunk header should be checked
45     vitam_force_chunk_mode: false
46     # syslog_facility
47     syslog_facility: local0
48
49 #####
50 ### Default Components parameters
51 ### Uncomment them if you want to update the default value applied on all
↳components
52
53 ### Ontology cache settings (max entries in cache & retention timeout in
↳seconds)
54 # ontologyCacheMaxEntries: 100
55 # ontologyCacheTimeoutInSeconds: 300
56 ### Elasticsearch scroll timeout in milliseconds settings
57 # elasticSearchScrollTimeoutInMilliseconds: 300000

```

(suite sur la page suivante)

```

58
59   ### The following values can be overwritten for each components in vitam: ↵
↵ parameters.
60   jvm_log: false
61   performance_logger: false
62
63   # consul_business_check: 10 # value in seconds
64   # consul_admin_check: 10 # value in seconds
65
66
67   ### Logs configuration for reconstruction services (INFO or DEBUG for active ↵
↵ logs).
68   ### Logs will be present only on secondary site.
69   ### Available for the following components: logbook, metadata & functional- ↵
↵ administration.
70   reconstruction:
71     log_level: INFO
72
73   # Used in ingest, unitary update, mass-update
74   classificationList: [ "Non protégé", "Secret Défense", "Confidentiel Défense" ]
75   # Used in ingest, unitary update, mass-update
76   classificationLevelOptional: true
77   # Packages install retries
78   packages_install_retries_number: 1
79   packages_install_retries_delay: 10
80
81   # Request time check settings. Do NOT update except if required by Vitam support
82   # Max acceptable time desynchronization between machines (in seconds).
83   acceptableRequestTime: 10
84   # Critical time desynchronization between machines (in seconds).
85   criticalRequestTime: 60
86   # Request time alert throttling Delay (in seconds)
87   requestTimeAlertThrottlingDelay: 60
88
89   # Reconstruction config
90   restoreBulkSize: 10000
91
92   vitam_timers:
93     # /\ \ IMPORTANT :
94     # Please ensure timer execution is spread so that not all timers run ↵
↵ concurrently (eg. *:05:00, *:35:00, *:50:00..),
95     # Special care for heavy-load timers that run on same machines or use same ↵
↵ resources (eg. vitam-traceability-*).
96     #
97     # Quartz cron nomenclature
98     #   minutely → 0 * * * * ?
99     #   hourly → 0 0 * * * ?
100    #   daily → 0 0 0 * * ?
101    #   monthly → 0 0 0 1 * ?
102    #   weekly → 0 0 0 ? * MON *
103    #   yearly → 0 0 0 1 1 ?
104    #   quarterly → 0 0 0 1 1/3 ?
105    #   semiannually → 0 0 0 1 1/6 ?
106    logbook: # all have to run on only one machine
107     # Sécurisation des journaux des opérations
108     frequency_traceability_operations: "* 05 0/1 * * ?" # every hour
109     # Sécurisation des journaux du cycle de vie des groupes d'objets

```

(suite sur la page suivante)

(suite de la page précédente)

```

110 frequency_traceability_lfc_objectgroup: "* 15 0/1 * * ?" # every hour
111 # Sécurisation des journaux du cycle de vie des unités archivistiques
112 frequency_traceability_lfc_unit: "* 35 0/1 * * ?" # every hour
113 # Audit de traçabilité
114 frequency_traceability_audit: "0 55 00 * * ?"
115 # Reconstruction (uniquement sur site secondaire)
116 frequency_logbook_reconstruction: "0 0/5 * * * ?"
117 storage:
118 # Sécurisation du journal des écritures
119 frequency_traceability_log: "0 40 0/4 * * ?" # every 4 hours
120 # Sauvegarde des journaux d'accès
121 vitam_storage_accesslog_backup: "0 10 0/4 * * ?" # every 4 hours
122 # Sauvegarde des journaux des écritures
123 vitam_storage_log_backup: "0 15 0/4 * * ?" # every 4 hours
124
125 functional_administration:
126 frequency_create_accession_register_symbolic: "0 50 0 * * ?"
127 frequency_accession_register_reconstruction: "0 0/5 * * * ?"
128 frequency_rule_management_audit: "0 40 * * * ?"
129 frequency_reconstruction: "0 0/5 * * * ?"
130 frequency_integrity_audit: "0 0 0 1 JAN ? 2020"
131 frequency_existence_audit: "0 0 0 1 JAN ? 2020"
132 metadata:
133 frequency_store_graph: "0 10/30 * * * ?"
134 frequency_reconstruction: "0 0/5 * * * ?"
135 frequency_computed_inherited_rules: "0 30 2 * * ?"
136 frequency_purge_dip: "0 0 * * * ?"
137 frequency_purge_transfers_sip: "0 25 2 * * ?"
138 frequency_audit_mongodb_es: "0 0 0 1 JAN ? 2020"
139 offer:
140 # Compaction offer logs
141 frequency_offerlog_compaction: "0 40 * * * ?"
142
143 scheduler:
144 job_parameters:
145 integrity_audit:
146 operations_delay_in_minutes: 1440
147 existence_audit:
148 operations_delay_in_minutes: 1440
149
150
151 ### consul ###
152 # WARNING: consul_domain should be a supported domain name for your organization
153 # You will have to generate server certificates with the same domain,
154 ↪ name and the service subdomain name
155 # Example: consul_domain=vitam means you will have to generate some,
156 ↪ certificates with .service.vitam domain
157 # access-external.service.vitam, ingest-external.service.vitam,
158 ↪ ...
159 consul_domain: consul
160 consul_folder_conf: "{{ vitam_defaults.folder.root_path }}/conf/consul"
161
162 # Workspace should be useless but storage have a dependency to it...
163 # elastic-kibana-interceptor is present as kibana is present, if kibana-data &
164 ↪ interceptor are not needed in the secondary site, just do not add them in the
165 ↪ hosts file
166 vitam_secondary_site_components: [ "scheduler", "logbook", "metadata",
167 ↪ "functional-administration", "storage", "storageofferdefault", (suite sur la page suivante)
168 ↪ "elasticsearch-log", "elasticsearch-data", "logstash", "kibana", "mongoc",
169 ↪ "mongod", "mongos", "elastic-kibana-interceptor", "consul" ]

```

```

162
163 # containers list
164 containers_list: [ 'units', 'objects', 'objectgroups', 'logbooks', 'reports',
↳ 'manifests', 'profiles', 'storagelog', 'storageaccesslog', 'storagetraceability
↳ ', 'rules', 'dip', 'agencies', 'backup', 'backupoperations', 'unitgraph',
↳ 'objectgroupgraph', 'distributionreports', 'accessionregistersdetail',
↳ 'accessionregisterssymbolic', 'tmp', 'archivaltransferreply' ]
165
166 ### Composants Vitam ###
167 vitam:
168   ### All available parameters for each components are described in the vitam_
↳ defaults variable
169
170   ### Example
171   # component:
172   #   at_boot: false
173   #   logback_rolling_policy: true
174   ## Force the log level for this component. Available logback values are (TRACE, _
↳ DEBUG, INFO, WARN, ERROR, OFF)
175   ## If this var is not set, the default one will be used (vitam_defaults.
↳ services.log_level)
176   #   log_level: "DEBUG"
177
178 accessexternal:
179   # Component name: do not modify
180   vitam_component: access-external
181   # DNS record for the service:
182   # Modify if ihm-demo is not using consul (typical production deployment)
183   host: "access-external.service.{{ consul_domain }}"
184   port_admin: 28102
185   port_service: 8444
186   baseuri: "access-external"
187   https_enabled: true
188   # Use platform secret for this component ? : do not modify
189   secret_platform: "false"
190   authorizeTrackTotalHits: false # if false, limit results to 10K. if true, _
↳ authorize results overs 10K (can overload elasticsearch-data)
191 accessinternal:
192   vitam_component: access-internal
193   host: "access-internal.service.{{ consul_domain }}"
194   port_service: 8101
195   port_admin: 28101
196   baseuri: "access-internal"
197   https_enabled: false
198   secret_platform: "true"
199 functional_administration:
200   vitam_component: functional-administration
201   host: "functional-administration.service.{{ consul_domain }}"
202   port_service: 8004
203   port_admin: 18004
204   baseuri: "adminmanagement"
205   https_enabled: false
206   secret_platform: "true"
207   cluster_name: "{{ elasticsearch.data.cluster_name }}"
208   # Number of AccessionRegisterSymbolic creation threads that can be run in _
↳ parallel.
209   accessionRegisterSymbolicThreadPoolSize: 16

```

(suite de la page précédente)

```

210 # Number of rule audit threads that can be run in parallel.
211 ruleAuditThreadPoolSize: 16
212 # Reconstruction metrics cache in minutes (secondary site)
213 reconstructionMetricsCacheDurationInMinutes: 15
214 scheduler:
215   vitam_component: scheduler
216   host: "scheduler.service.{{ consul_domain }}"
217   port_service: 8799
218   port_admin: 28799
219   baseuri: "scheduler"
220   https_enabled: false
221   secret_platform: "true"
222   schedulerThreadSize: 8
223 elasticsearchinterceptor:
224   vitam_component: elastic-kibana-interceptor
225   host: "elastic-kibana-interceptor.service.{{ consul_domain }}"
226   port_service: 8014
227   port_admin: 18014
228   baseuri: ""
229   https_enabled: false
230   secret_platform: "false"
231   cluster_name: "{{ elasticsearch.data.cluster_name }}"
232 batchreport:
233   vitam_component: batch-report
234   host: "batch-report.service.{{ consul_domain }}"
235   port_service: 8015
236   port_admin: 18015
237   baseuri: "batchreport"
238   https_enabled: false
239   secret_platform: "false"
240 ingestexternal:
241   vitam_component: ingest-external
242   # DNS record for the service:
243   # Modify if ihm-demo is not using consul (typical production deployment)
244   host: "ingest-external.service.{{ consul_domain }}"
245   port_admin: 28001
246   port_service: 8443
247   baseuri: "ingest-external"
248   https_enabled: true
249   secret_platform: "false"
250   antivirus: "clamav" # or avast
251   #scantimeout: 1200000 # value in milliseconds; increase this value if huge
↳files need to be analyzed in more than 20 min (default value)
252   # Directory where files should be placed for local ingest
253   upload_dir: "/vitam/data/ingest-external/upload"
254   # Directory where successful ingested files will be moved to
255   success_dir: "/vitam/data/ingest-external/upload/success"
256   # Directory where failed ingested files will be moved to
257   fail_dir: "/vitam/data/ingest-external/upload/failure"
258   # Action done to file after local ingest (see below for further information)
259   upload_final_action: "MOVE"
260   # upload_final_action can be set to three different values (lower or upper
↳case does not matter)
261   # 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
262   # DELETE : After upload, the local file will be deleted if the upload
↳succeeded

```

(suite sur la page suivante)

(suite de la page précédente)

```

263 # 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)
264 ingestinternal:
265   vitam_component: ingest-internal
266   host: "ingest-internal.service.{{ consul_domain }}"
267   port_service: 8100
268   port_admin: 28100
269   baseuri: "ingest"
270   https_enabled: false
271   secret_platform: "true"
272 ihm_demo:
273   vitam_component: ihm-demo
274   host: "ihm-demo.service.{{ consul_domain }}"
275   port_service: 8446
276   port_admin: 28002
277   baseurl: "/ihm-demo"
278   static_content: "{{ vitam_defaults.folder.root_path }}/app/ihm-demo/v2"
279   baseuri: "ihm-demo"
280   https_enabled: true
281   secret_platform: "false"
282   # User session timeout in milliseconds (for shiro)
283   session_timeout: 1800000
284   secure_cookie: true
285   # Specify here the realms you want to use for authentication in ihm-demo
286   # You can set multiple realms, one per line
287   # With multiple realms, the user will be able to choose between the allowed
↳realms
288   # Example: authentication_realms:
289   #           - x509Realm
290   #           - ldapRealm
291   # Authorized values:
292   # x509Realm: certificate
293   # iniRealm: ini file
294   # ldapRealm: ldap
295   authentication_realms:
296     # - x509Realm
297     - iniRealm
298     # - ldapRealm
299   allowedMediaTypes:
300     - type: "application"
301       subtype: "pdf"
302     - type: "text"
303       subtype: "plain"
304     - type: "image"
305       subtype: "jpeg"
306     - type: "image"
307       subtype: "tiff"
308 logbook:
309   vitam_component: logbook
310   host: "logbook.service.{{ consul_domain }}"
311   port_service: 9002
312   port_admin: 29002
313   baseuri: "logbook"
314   https_enabled: false
315   secret_platform: "true"
316   cluster_name: "{{ elasticsearch.data.cluster_name }}"
317   # Temporization delay (in seconds) for recent logbook operation events.

```

(suite sur la page suivante)

(suite de la page précédente)

```

318 # Set it to a reasonable delay to cover max clock difference across servers +_
↳VM/GC pauses
319 operationTraceabilityTemporizationDelay: 300
320 # Max delay between 2 logbook operation traceability operations.
321 # A new logbook operation traceability is generated after this delay, even if_
↳tenant has no
322 # new logbook operations to secure
323 # Unit can be in DAYS, HOURS, MINUTES, SECONDS
324 # Hint: Set it to 690 MINUTES (11 hours and 30 minutes) to force new_
↳traceability after +/- 12 hours (supposing
325 # logbook operation traceability timer run every hour +/- some clock delays)
326 operationTraceabilityMaxRenewalDelay: 690
327 operationTraceabilityMaxRenewalDelayUnit: MINUTES
328 # Number of logbook operations that can be run in parallel.
329 operationTraceabilityThreadPoolSize: 16
330 # Temporization delay (in seconds) for recent logbook lifecycle events.
331 # Set it to a reasonable delay to cover max clock difference across servers +_
↳VM/GC pauses
332 lifecycleTraceabilityTemporizationDelay: 300
333 # Max delay between 2 lifecycle traceability operations.
334 # A new unit/objectgroup lifecycle traceability is generated after this delay,
↳ even if tenant has no
335 # new unit/objectgroups to secure
336 # Unit can be in DAYS, HOURS, MINUTES, SECONDS
337 # Hint: Set it to 690 MINUTES (11 hours and 30 minutes) to force new_
↳traceability after +/- 12 hours (supposing
338 # LFC traceability timers run every hour +/- some clock delays)
339 lifecycleTraceabilityMaxRenewalDelay: 690
340 lifecycleTraceabilityMaxRenewalDelayUnit: MINUTES
341 # Max entries selected per (Unit or Object Group) LFC traceability operation
342 lifecycleTraceabilityMaxEntries: 100000
343 # Reconstruction metrics cache in minutes (secondary site)
344 reconstructionMetricsCacheDurationInMinutes: 15
345 metadata:
346 vitam_component: metadata
347 host: "metadata.service.{{ consul_domain }}"
348 port_service: 8200
349 port_admin: 28200
350 baseuri: "metadata"
351 https_enabled: false
352 secret_platform: "true"
353 cluster_name: "{{ elasticsearch.data.cluster_name }}"
354 # Archive Unit Profile cache settings (max entries in cache & retention_
↳timeout in seconds)
355 archiveUnitProfileCacheMaxEntries: 100
356 archiveUnitProfileCacheTimeoutInSeconds: 300
357 # Schema validator cache settings (max entries in cache & retention timeout_
↳in seconds)
358 schemaValidatorCacheMaxEntries: 100
359 schemaValidatorCacheTimeoutInSeconds: 300
360 # DIP cleanup delay (in minutes)
361 dipTimeToLiveInMinutes: 10080 # 7 days
362 criticalDipTimeToLiveInMinutes: 1440 # 1 day
363 transfersSIPTimeToLiveInMinutes: 10080 # 7 days
364 unitsStreamThreshold: 1000000 # 1 million
365 unitsStreamExecutionLimit: 3 # 3 times
366 objectsStreamThreshold: 1000000 # 1 million

```

(suite sur la page suivante)

```

367   objectsStreamExecutionLimit: 3 # 3 times
368   workspaceFreespaceThreshold: 25 # when below use critical time to live when_
↳above use normal time to live
369   elasticsearch_mapping_dir: "{ vitam_defaults.folder.root_path }/conf/
↳metadata/mapping" # Directory of elasticsearch metadata mapping
370   ##### Audit data consistency MongoDB-ES #####
371   isDataConsistencyAuditRunnable: false
372   dataConsistencyAuditOplogMaxSize: 100
373   # Reconstruction metrics cache in minutes (secondary site)
374   reconstructionMetricsCacheDurationInMinutes: 15
375   context_path: "/metadata"
376   processing:
377     vitam_component: processing
378     host: "processing.service.{{ consul_domain }}"
379     port_service: 8203
380     port_admin: 28203
381     baseuri: "processing"
382     https_enabled: false
383     secret_platform: "true"
384     maxDistributionInMemoryBufferSize: 100000
385     maxDistributionOnDiskBufferSize: 100000000
386   security_internal:
387     vitam_component: security-internal
388     host: "security-internal.service.{{ consul_domain }}"
389     port_service: 8005
390     port_admin: 28005
391     baseuri: "security-internal"
392     https_enabled: false
393     secret_platform: "true"
394   storageengine:
395     vitam_component: storage
396     host: "storage.service.{{ consul_domain }}"
397     port_service: 9102
398     port_admin: 29102
399     baseuri: "storage"
400     https_enabled: false
401     secret_platform: "true"
402     storageTraceabilityOverlapDelay: 300
403     restoreBulkSize: 1000
404     # Storage write/access log backup max thread pool size
405     storageLogBackupThreadPoolSize: 16
406     # Storage write log traceability thread pool size
407     storageLogTraceabilityThreadPoolSize: 16
408     # Offer synchronization batch size & thread pool size
409     offerSynchronizationBulkSize: 1000
410     offerSyncThreadPoolSize: 32
411     # Retries attempts on failures
412     offerSyncNumberOfRetries: 3
413     # Retry wait delay on failures (in seconds)
414     offerSyncFirstAttemptWaitingTime: 15
415     offerSyncWaitingTime: 30
416     # Offer synchronization wait delay (in seconds) for async offers_
↳(synchronization from a tape-storage offer)
417     offerSyncAccessRequestCheckWaitingTime: 10
418     logback_total_size_cap:
419     offersync:
420       history_days: 30

```

(suite de la page précédente)

```

421     totalsize: "5GB"
422     offerdiff:
423         history_days: 30
424         totalsize: "5GB"
425     # unit time per kB (in ms) used while calculating the timeout of an http_
↪request between storage and offer.
426     timeoutMsPerKB: 100
427     # minimum timeout (in ms) for writing objects to offers
428     minWriteTimeoutMs: 60000
429     # minimum timeout per object (in ms) for bulk writing objects to offers
430     minBulkWriteTimeoutMsPerObject: 10000
431     storageofferdefault:
432         vitam_component: "offer"
433         port_service: 9900
434         port_admin: 29900
435         baseuri: "offer"
436         https_enabled: false
437         secret_platform: "true"
438         logback_total_size_cap:
439             offer_tape:
440                 history_days: 30
441                 totalsize: "5GB"
442             offer_tape_backup:
443                 history_days: 30
444                 totalsize: "5GB"
445     worker:
446         vitam_component: worker
447         host: "worker.service.{{ consul_domain }}"
448         port_service: 9104
449         port_admin: 29104
450         baseuri: "worker"
451         https_enabled: false
452         secret_platform: "true"
453         api_output_index_tenants: [ 0,1,2,3,4,5,6,7,8,9 ]
454         rules_index_tenants: [ 0,1,2,3,4,5,6,7,8,9 ]
455         # Archive Unit Profile cache settings (max entries in cache & retention_
↪timeout in seconds)
456         archiveUnitProfileCacheMaxEntries: 100
457         archiveUnitProfileCacheTimeoutInSeconds: 300
458         # Schema validator cache settings (max entries in cache & retention timeout_
↪in seconds)
459         schemaValidatorCacheMaxEntries: 100
460         schemaValidatorCacheTimeoutInSeconds: 300
461         # Batch size for bulk atomic update
462         queriesThreshold: 100000
463         # Bulk atomic update batch size
464         bulkAtomicUpdateBatchSize: 100
465         # Max threads that can be run in concurrently is thread pool for bulk atomic_
↪update
466         bulkAtomicUpdateThreadPoolSize: 8
467         # Number of jobs that can be queued in memory before blocking for bulk atomic_
↪update
468         bulkAtomicUpdateThreadPoolQueueSize: 16
469         # Dip/transfer threshold file size
470         binarySizePlatformThreshold: 1
471         binarySizePlatformThresholdSizeUnit: "GIGABYTE"
472     workspace:

```

(suite sur la page suivante)

```

473   vitam_component: workspace
474   host: "workspace.service.{{ consul_domain }}"
475   port_service: 8201
476   port_admin: 28201
477   baseuri: "workspace"
478   https_enabled: false
479   secret_platform: "true"
480   context_path: "/workspace"
481   collect_internal:
482     vitam_component: collect-internal
483     host: "collect-internal.service.{{ consul_domain }}"
484     port_service: 8038
485     port_admin: 28038
486     baseuri: "collect-internal"
487     https_enabled: false
488     secret_platform: "true"
489     transactionStatusThreadPoolSize: 4
490     statusTransactionThreadFrequency: 5
491   collect_external:
492     vitam_component: collect-external
493     host: "collect-external.service.{{ consul_domain }}"
494     port_service: 8030
495     port_admin: 28030
496     baseuri: "collect-external"
497     https_enabled: true
498     secret_platform: "false"
499     authorizeTrackTotalHits: false # if false, limit results to 10K. if true,
↳authorize results overs 10K (can overload elasticsearch-data)
500   metadata_collect:
501     vitam_component: metadata-collect
502     host: "metadata-collect.service.{{ consul_domain }}"
503     port_service: 8290
504     port_admin: 28290
505     baseuri: "metadata-collect"
506     https_enabled: false
507     secret_platform: "true"
508     cluster_name: "{{ elasticsearch.data.cluster_name }}"
509     # Archive Unit Profile cache settings (max entries in cache & retention,
↳timeout in seconds)
510     archiveUnitProfileCacheMaxEntries: 100
511     archiveUnitProfileCacheTimeoutInSeconds: 300
512     # Schema validator cache settings (max entries in cache & retention timeout,
↳in seconds)
513     schemaValidatorCacheMaxEntries: 100
514     schemaValidatorCacheTimeoutInSeconds: 300
515     # DIP cleanup delay (in minutes)
516     dipTimeToLiveInMinutes: 10080 # 7 days
517     criticalDipTimeToLiveInMinutes: 1440 # 1 day
518     transfersSIPTimeToLiveInMinutes: 10080 # 7 days
519     workspaceFreespaceThreshold: 25 # when below use critical time to live when,
↳above use normal time to live
520     elasticsearch_mapping_dir: "{{ vitam_defaults.folder.root_path }}/conf/
↳metadata-collect/mapping" # Directory of elasticsearch metadata mapping
521     ##### Audit data consistency MongoDB-ES #####
522     isDataConsistencyAuditRunnable: false
523     dataConsistencyAuditOpllogMaxSize: 100
524     context_path: "/metadata-collect"

```

(suite de la page précédente)

```

525 workspace_collect:
526   vitam_component: workspace-collect
527   host: "workspace-collect.service.{{ consul_domain }}"
528   port_service: 8291
529   port_admin: 28291
530   baseuri: "workspace-collect"
531   https_enabled: false
532   secret_platform: "true"
533   context_path: "/workspace-collect"
534
535 # http://www.programmevitam.fr/ressources/DocCourante/html/installation/
536 ↪installation/21-addons.html#durees-minimales-permettant-de-controler-les-
537 ↪valeurs-saisies
538 vitam_tenant_rule_duration:
539 # - name: 2 # applied tenant
540 #   rules:
541 #     - AppraisalRule : "1 year" # rule name : rule value
542
543 # If you want to deploy vitam in a single VM, add the vm name in this array
544 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.

- `deployment/environments/group_vars/all/advanced/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   log_level: WARN # Available log_level are: TRACE, DEBUG, INFO, WARN or ERR
8
9 # Please uncomment and fill values if you want to connect VITAM to external SIEM
10 # external_siem:
11 #   host:
12 #   port:
13
14 elasticsearch:
15   log:
16     host: "elasticsearch-log.service.{{ consul_domain }}"
17     port_http: "9201"
18     groupe: "log"

```

(suite sur la page suivante)

```

19     baseuri: "elasticsearch-log"
20     cluster_name: "elasticsearch-log"
21     consul_check_http: 10 # in seconds
22     consul_check_tcp: 10 # in seconds
23     action_log_level: error
24     https_enabled: false
25     indices fielddata_cache_size: '30%' # related to https://www.elastic.co/
↳guide/en/elasticsearch/reference/7.6/modules-fielddata.html
26     indices_breaker_fielddata_limit: '40%' # related to https://www.elastic.
↳co/guide/en/elasticsearch/reference/7.6/circuit-breaker.html#fielddata-circuit-
↳breaker
27     dynamic_timeout: 30s
28     # default index template
29     index_templates:
30         packetbeat:
31             shards: 5
32     log_appenders:
33         root:
34             log_level: "info"
35         rolling:
36             max_log_file_size: "100MB"
37             max_total_log_size: "5GB"
38             max_files: "50"
39         deprecation_rolling:
40             max_log_file_size: "100MB"
41             max_total_log_size: "1GB"
42             max_files: "10"
43             log_level: "warn"
44         index_search_slowlog_rolling:
45             max_log_file_size: "100MB"
46             max_total_log_size: "1GB"
47             max_files: "10"
48             log_level: "warn"
49         index_indexing_slowlog_rolling:
50             max_log_file_size: "100MB"
51             max_total_log_size: "1GB"
52             max_files: "10"
53             log_level: "warn"
54     # By default, is commented. Should be uncommented if ansible computes
↳badly vCPUs number ; values are associated vCPUs numbers ; please adapt to
↳your configuration
55     # thread_pool:
56     #     index:
57     #         size: 2
58     #     get:
59     #         size: 2
60     #     search:
61     #         size: 2
62     #     write:
63     #         size: 2
64     #     warmer:
65     #         max: 2
66     data:
67         host: "elasticsearch-data.service.{{ consul_domain }}"
68         # default is 0.1 (10%) and should be quite enough in most cases
69         #index_buffer_size_ratio: "0.15"
70         port_http: "9200"

```

(suite de la page précédente)

```

71     groupe: "data"
72     baseuri: "elasticsearch-data"
73     cluster_name: "elasticsearch-data"
74     consul_check_http: 10 # in seconds
75     consul_check_tcp: 10 # in seconds
76     action_log_level: debug
77     https_enabled: false
78     indices fielddata_cache_size: '30%' # related to https://www.elastic.co/
↪guide/en/elasticsearch/reference/6.5/modules-fielddata.html
79     indices breaker fielddata_limit: '40%' # related to https://www.elastic.
↪co/guide/en/elasticsearch/reference/6.5/circuit-breaker.html#fielddata-circuit-
↪breaker
80     dynamic_timeout: 30s
81     # default index template
82     index_templates:
83     log_appenders:
84         root:
85             log_level: "info"
86         rolling:
87             max_log_file_size: "100MB"
88             max_total_log_size: "5GB"
89             max_files: "50"
90         deprecation_rolling:
91             max_log_file_size: "100MB"
92             max_total_log_size: "5GB"
93             max_files: "50"
94             log_level: "warn"
95         index_search_slowlog_rolling:
96             max_log_file_size: "100MB"
97             max_total_log_size: "5GB"
98             max_files: "50"
99             log_level: "warn"
100        index_indexing_slowlog_rolling:
101            max_log_file_size: "100MB"
102            max_total_log_size: "5GB"
103            max_files: "50"
104            log_level: "warn"
105        # By default, is commented. Should be uncommented if ansible computes
↪badly vCPUs number ; values are associated vCPUs numbers ; please adapt to
↪your configuration
106        # thread_pool:
107        #     index:
108        #         size: 2
109        #     get:
110        #         size: 2
111        #     search:
112        #         size: 2
113        #     write:
114        #         size: 2
115        #     warmer:
116        #         max: 2
117
118     mongodb:
119         mongos_port: 27017
120         mongoc_port: 27018
121         mongod_port: 27019
122         mongo_authentication: "true"

```

(suite sur la page suivante)


```

123  host: "mongos.service.{{ consul_domain }}"
124  check_consul: 10 # in seconds
125  drop_info_log: false # Drop mongo (I)nformational log, for Verbosity Level of
↪ 0
126  # logs configuration
127  logrotate: enabled # or disabled
128  history_days: 30 # How many days to store logs if logrotate is set to 'enabled'
↪ '
129
130  logstash:
131  host: "logstash.service.{{ consul_domain }}"
132  user: logstash
133  port: 10514
134  rest_port: 20514
135  check_consul: 10 # in seconds
136  # logstash xms & xmx in Megabytes
137  # jvm_xms: 2048
138  # jvm_xmx: 2048
139  # workers_number: 4
140  log_appenders:
141    rolling:
142      max_log_file_size: "100MB"
143      max_total_log_size: "5GB"
144    json_rolling:
145      max_log_file_size: "100MB"
146      max_total_log_size: "5GB"
147
148  # Prometheus params
149  prometheus:
150  metrics_path: /admin/v1/metrics
151  check_consul: 10 # in seconds
152  prometheus_config_file_target_directory: # Set path where "prometheus.yml"
↪ file will be generated. Example: /tmp/
153  server:
154    port: 9090
155    tsdb_retention_time: "7d"
156    tsdb_retention_size: "5GB"
157  node_exporter:
158    enabled: true
159    port: 9101
160    metrics_path: /metrics
161    log_level: "warn"
162    logrotate: enabled # or disabled
163    history_days: 30 # How many days to store logs if logrotate is set to
↪ 'enabled'
164  consul_exporter:
165    enabled: true
166    port: 9107
167    metrics_path: /metrics
168  elasticsearch_exporter:
169    enabled: true
170    port: 9114
171    metrics_path: /metrics
172    log_level: "warn"
173    logrotate: enabled # or disabled
174    history_days: 30 # How many days to store logs if logrotate is set to
↪ 'enabled'

```

(suite sur la page suivante)

(suite de la page précédente)

```

175 alertmanager:
176   api_port: 9093
177   cluster_port: 9094
178   #receivers: # https://grafana.com/blog/2020/02/25/step-by-step-guide-to-
↪ setting-up-prometheus-alertmanager-with-slack-pagerduty-and-gmail/
179   #- name: "slack_alert"
180   # slack_configs:
181   # - api_url: "https://hooks.slack.com/services/xxxxxxx/
↪ "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx"
182   # channel: '#your_alert_channel'
183   # send_resolved: true
184
185 grafana:
186   check_consul: 10 # in seconds
187   http_port: 3000
188   proxy: false
189   grafana_datasources:
190     - name: "Prometheus"
191       type: "prometheus"
192       access: "proxy"
193       url: "http://prometheus-server.service.{{ consul_domain }}:{{ prometheus.
↪ server.port | default(9090) }}/prometheus"
194     basicAuth: false
195     editable: true
196     - name: "Prometheus AlertManager"
197       type: "camptocamp-prometheus-alertmanager-datasource"
198       access: "proxy"
199       url: "http://prometheus-alertmanager.service.{{ consul_domain }}:{{
↪ prometheus.alertmanager.api_port | default(9093) }}"
200     basicAuth: false
201     editable: true
202     jsonData:
203       keepCookies: []
204       severity_critical: "4"
205       severity_high: "3"
206       severity_warning: "2"
207       severity_info: "1"
208     grafana_dashboards:
209     - name: 'vitam-dashboard'
210       orgId: 1
211       folder: ''
212       folderUId: ''
213       type: file
214       disableDeletion: false
215       updateIntervalSeconds: 10
216       allowUiUpdates: true
217       options:
218         path: "/etc/grafana/provisioning/dashboards"
219
220 # Curator units: days
221 curator:
222   log:
223     metricbeat:
224       close: 5
225       delete: 10
226     packetbeat:
227       close: 5

```

(suite sur la page suivante)

```
228         delete: 10
229
230 kibana:
231     header_value: "reporting"
232     import_delay: 10
233     import_retries: 10
234     # logs configuration
235     logrotate: enabled # or disabled
236     history_days: 30 # How many days to store logs if logrotate is set to 'enabled
↪ '
237     log:
238         baseuri: "kibana_log"
239         api_call_timeout: 120
240         groupe: "log"
241         port: 5601
242         default_index_pattern: "logstash-vitam*"
243         check_consul: 10 # in seconds
244         # default shards & replica
245         shards: 1
246         replica: 1
247         # pour index logstash-*
248         metrics:
249             shards: 1
250             replica: 1
251             # pour index metricbeat-*
252         metricbeat:
253             shards: 3 # must be a factor of 30
254             replica: 1
255     data:
256         baseuri: "kibana_data"
257         # OMA : bugdette : api_call_timeout is used for retries ; should ceate a
↪ separate variable rather than this one
258         api_call_timeout: 120
259         groupe: "data"
260         port: 5601
261         default_index_pattern: "logbookoperation_*"
262         check_consul: 10 # in seconds
263         # index template for .kibana
264         shards: 1
265         replica: 1
266
267 syslog:
268     # value can be syslog-ng or rsyslog
269     name: "rsyslog"
270
271 cerebro:
272     baseuri: "cerebro"
273     port: 9000
274     check_consul: 10 # in seconds
275     # logs configuration
276     logrotate: enabled # or disabled
277     history_days: 30 # How many days to store logs if logrotate is set to 'enabled
↪ '
278
279 siegfried:
280     port: 19000
281     consul_check: 10 # in seconds
```

(suite sur la page suivante)

(suite de la page précédente)

```

282
283 clamav:
284     port: 3310
285     # logs configuration
286     logrotate: enabled # or disabled
287     history_days: 30 # How many days to store logs if logrotate is set to 'enabled
↳ '
288     freshclam:
289         # frequency freshclam for database update per day (from 0 to 24 - 24
↳ meaning hourly check)
290         db_update_periodicity: 1
291         private_mirror_address:
292         use_proxy: "no"
293
294 ## Avast Business Antivirus for Linux
295 ## if undefined, the following default values are applied.
296 # avast:
297 #     # logs configuration
298 #     logrotate: enabled # or disabled
299 #     history_days: 30 # How many days to store logs if logrotate is set to
↳ 'enabled'
300 #     manage_repository: true
301 #     repository:
302 #         state: present
303 #         # For CentOS
304 #         baseurl: http://rpm.avast.com/lin/repo/dists/rhel/release
305 #         gpgcheck: no
306 #         proxy: _none_
307 #         # For Debian
308 #         baseurl: 'deb http://deb.avast.com/lin/repo debian-buster release'
309 #         vps_repository: http://linux-av.u.avcdn.net/linux-av/avast/x86_64
310 #         ## List of sha256 hash of excluded files from antivirus. Useful for test
↳ environments.
311 #         whitelist:
312 #             - xxxxxx
313 #             - yyyyyyy
314
315 mongo_express:
316     baseuri: "mongo-express"
317
318 ldap_authentication:
319     ldap_protocol: "ldap"
320     ldap_server: "{% if groups['ldap']|length > 0 %}{{ groups['ldap']|first }}{%
↳ endif %}"
321     ldap_port: "389"
322     ldap_base: "dc=programmevitam,dc=fr"
323     ldap_login: "cn=Manager,dc=programmevitam,dc=fr"
324     uid_field: "uid"
325     ldap_userDn_Template: "uid={0},ou=people,dc=programmevitam,dc=fr"
326     ldap_group_request: "(&(objectClass=groupOfNames)(member={0}))"
327     ldap_admin_group: "cn=admin,ou=groups,dc=programmevitam,dc=fr"
328     ldap_user_group: "cn=user,ou=groups,dc=programmevitam,dc=fr"
329     ldap_guest_group: "cn=guest,ou=groups,dc=programmevitam,dc=fr"
330
331 # Backup tool on storage-offer
332 restic:
333     snapshot_retention: 30 # number of snapshots to keep

```

(suite sur la page suivante)

(suite de la page précédente)

```

334 # default run backup at 23:00 everyday
335 cron:
336     minute: '00'
337     hour: '23'
338     day: '*'
339     month: '*'
340     weekday: '*'
341 # [hosts_storage_offer_default] must be able to connect to the listed
↪databases below to properly backup.
342 backup:
343     # mongo-offer
344     - name: "{{ offer_conf }}"
345       type: mongod
346       host: "{{ offer_conf }}-mongos.service.consul:{{ mongodb.mongos_port }}"
347       user: "{{ mongodb[offer_conf].admin.user }}"
348       password: "{{ mongodb[offer_conf].admin.password }}"
349     # # mongo-data (only if mono-sharded cluster)
350     # - name: mongo-data
351       # type: mongod
352       # host: "mongo-data-mongos.service.consul:{{ mongodb.mongos_port }}"
353       # user: "{{ mongodb['mongo-data'].admin.user }}"
354       # password: "{{ mongodb['mongo-data'].admin.password }}"
355     # # mongo-vitamui (only if vitamui is deployed)
356     # - name: mongo-vitamui
357       # type: mongod
358       # host: mongo-vitamui-mongod.service.consul:{{ mongodb.mongod_port }}
359     # # Add the following params on environments/group_vars/all/main/vault-
↪vitam.yml
360     # # They can be found under vitamui's deployment sources on
↪environments/group_vars/all/vault-mongod.yml
361     # user: "{{ mongodb['mongo-vitamui'].admin.user }}"
362     # password: "{{ mongodb['mongo-vitamui'].admin.password }}"

```

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).

- deployment/environments/group_vars/all/advanced/jvm_opts.yml, comme suit :

```

1 ---
2
3 # Default values if unset
4 # jvm_opts.memory: "-Xms512m -Xmx512m"
5 # gc: "-Xlog:gc*,gc+age=trace,safepoint:file={{ vitam_folder_log }}/gc.
↪log:utctime,pid,tags:filecount=32,filesize=64m"
6 # java: ""
7
8 vitam:
9     accessinternal:
10         jvm_opts:
11             # memory: "-Xms512m -Xmx512m"
12             # gc: ""

```

(suite sur la page suivante)

(suite de la page précédente)

```

13         # java: ""
14 accessexternal:
15     jvm_opts:
16         # memory: "-Xms512m -Xmx512m"
17         # gc: ""
18         # java: ""
19 elastickibanainterceptor:
20     jvm_opts:
21         # memory: "-Xms512m -Xmx512m"
22         # gc: ""
23         # java: ""
24 batchreport:
25     jvm_opts:
26         # memory: "-Xms512m -Xmx512m"
27         # gc: ""
28         # java: ""
29 ingestinternal:
30     jvm_opts:
31         # memory: "-Xms512m -Xmx512m"
32         # gc: ""
33         # java: ""
34 ingestexternal:
35     jvm_opts:
36         # memory: "-Xms512m -Xmx512m"
37         # gc: ""
38         # java: ""
39 metadata:
40     jvm_opts:
41         # memory: "-Xms512m -Xmx512m"
42         # gc: ""
43         # java: ""
44 ihm_demo:
45     jvm_opts:
46         # memory: "-Xms512m -Xmx512m"
47         # gc: ""
48         # java: ""
49 ihm_recette:
50     jvm_opts:
51         # memory: "-Xms512m -Xmx512m"
52         # gc: ""
53         # java: ""
54 logbook:
55     jvm_opts:
56         # memory: "-Xms512m -Xmx512m"
57         # gc: ""
58         # java: ""
59 workspace:
60     jvm_opts:
61         # memory: "-Xms512m -Xmx512m"
62         # gc: ""
63         # java: ""
64 processing:
65     jvm_opts:
66         # memory: "-Xms512m -Xmx512m"
67         # gc: ""
68         # java: ""
69 worker:

```

(suite sur la page suivante)

```
70     jvm_opts:
71         # memory: "-Xms512m -Xmx512m"
72         # gc: ""
73         # java: ""
74     storageengine:
75         jvm_opts:
76             # memory: "-Xms512m -Xmx512m"
77             # gc: ""
78             # java: ""
79     storageofferdefault:
80         jvm_opts:
81             # memory: "-Xms512m -Xmx512m"
82             # gc: ""
83             # java: ""
84     functional_administration:
85         jvm_opts:
86             # memory: "-Xms512m -Xmx512m"
87             # gc: ""
88             # java: ""
89     scheduler:
90         jvm_opts:
91             # memory: "-Xms512m -Xmx512m"
92             # gc: ""
93             # java: ""
94     security_internal:
95         jvm_opts:
96             # memory: "-Xms512m -Xmx512m"
97             # gc: ""
98             # java: ""
99     library:
100         jvm_opts:
101             memory: "-Xms32m -Xmx128m"
102             # gc: ""
103             # java: ""
104     collect_internal:
105         jvm_opts:
106             # memory: "-Xms512m -Xmx512m"
107             # gc: ""
108             # java: ""
109     collect_external:
110         jvm_opts:
111             # memory: "-Xms512m -Xmx512m"
112             # gc: ""
113             # java: ""
114     metadata_collect:
115         jvm_opts:
116             # memory: "-Xms512m -Xmx512m"
117             # gc: ""
118             # java: ""
119     workspace_collect:
120         jvm_opts:
121             # memory: "-Xms512m -Xmx512m"
122             # gc: ""
123             # java: ""
```

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

défini dans ansible.

4.2.5.14 Paramétrage de l'Offre Froide (bibliothèques de cartouches)

Voir aussi :

Les principes de fonctionnement de l'offre froide sont décrits dans la documentation externe dédiée (« Archivage sur Offre Froide »).

La bibliothèque et les lecteurs doivent déjà être configurés sur la machine devant supporter une instance de ce composant (avec login automatique en cas de reboot).

La commande `lsscsi -g` peut permettre de vérifier si des périphériques sont détectés.

Note : Une offre froide est mono-instantiable uniquement. Elle ne peut être déployée en haut-disponibilité.

Le paramétrage de l'offre froide se fait via la configuration du fichier `deployment/environments/group_vars/all/offer_opts.yml`. L'ensemble des clés disponibles est listé dans le fichier `deployment/environments/group_vars/all/offer_opts.yml.example`

L'offre froide doit être configurée avec le flag `AsyncRead` défini à `True` dans la stratégie par défaut de Vitam via `vitam_strategy` ou dans une stratégie additionnelle `other_strategies`.

Exemple :

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

Une offre froide 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 section `tapeLibraryConfiguration` décrit le paramétrage général de l'offre froide.

- **maxTarEntrySize** Taille maximale (en octets) au-delà de la laquelle les fichiers entrants seront découpés en segments. Typiquement 1 Go, maximum 8 Go.
- **maxTarFileSize** Taille maximale (en octets) des *tars* à constituer. Typiquement 10 Go.
- **forceOverrideNonEmptyCartridges** Permet de passer outre le contrôle vérifiant que les bandes nouvellement introduites sont vides. Par défaut à *false*. Ne doit être défini à *true* que sur un environnement de recette où l'écrasement d'une bande de test est sans risque.
- **cachedTarMaxStorageSpaceInMB** Permet de définir la taille maximale du cache disque (en Mo) (Ex. 10 To pour un env de production)
- **cachedTarEvictionStorageSpaceThresholdInMB** Permet de définir la taille critique du cache disque (en Mo). Une fois ce seuil atteint, les archives non utilisées sont purgées (selon la date de dernier accès). Doit être plus petit que la taille maximale **cachedTarMaxStorageSpaceInMB**. (Ex. 8 To pour un env de production)

- **cachedTarSafeStorageSpaceThresholdInMB** Seuil « confortable » d'utilisation du cache (en Mo). Le processus d'éviction des archives du cache s'arrête lorsque ce seuil est atteint. Doit être plus petit que la taille critique **cachedTarEvictionStorageSpaceThresholdInMB**. (Ex. 6 To pour un env de production)
- **maxAccessRequestSize** Définit un seuil technique du nombre d'objets que peut cibler une demande d'accès. Par défaut de 10000. À ne pas modifier.
- **readyAccessRequestExpirationDelay** Valeur du délais d'expiration des demandes d'accès. Une fois une demande d'accès à des objets est prête, l'accès immédiat aux objets est garantie durant cette période.
- **readyAccessRequestExpirationUnit** Unité du délais d'expiration des demandes d'accès (une valeur parmi « SECONDS » / « MINUTES » / « HOURS » / « DAYS » / « MONTHS »).
- **readyAccessRequestPurgeDelay** Valeur du délais de purge complète des demandes d'accès.
- **readyAccessRequestPurgeUnit** Unité du délais de purge complète des demandes d'accès (une valeur parmi « SECONDS » / « MINUTES » / « HOURS » / « DAYS » / « MONTHS »).
- **accessRequestCleanupTaskIntervalDelay** Valeur de la fréquence de nettoyage des demandes d'accès.
- **accessRequestCleanupTaskIntervalUnit** Unité de la fréquence de nettoyage des demandes d'accès (une valeur parmi « SECONDS » / « MINUTES » / « HOURS » / « DAYS » / « MONTHS »).

Note : maxTarEntrySize doit être strictement inférieur à maxTarFileSize

Note : cachedTarEvictionStorageSpaceThresholdInMB doit être strictement inférieur à cachedTarMaxStorageSpaceInMB

Note : cachedTarSafeStorageSpaceThresholdInMB doit être strictement inférieur à cachedTarEvictionStorageSpaceThresholdInMB

Note : Se référer à la documentation *DAT* pour les éléments de dimensionnement du cache.

Note : La durée de purge des demandes d'accès doit être strictement supérieure à leur durée d'expiration.

Note : Le monitoring de l'offre froide est for est **fortement recommandé** afin de s'assurer du bon fonctionnement de l'offre, et du dimensionnement du disque local. Un dashboard dédié à l'offre froide de Vitam est déployé avec les composants « extra » prometheus et grafana.

Exemple :

```
inputFileStorageFolder: "/vitam/data/offer/offer/inputFiles"
inputTarStorageFolder: "/vitam/data/offer/offer/inputTars"
tmpTarOutputStorageFolder: "/vitam/data/offer/offer/tmpTarOutput"
cachedTarStorageFolder: "/vitam/data/offer/offer/cachedTars"
maxTarEntrySize: 10000000
maxTarFileSize: 10000000000
ForceOverrideNonEmptyCartridge: false
cachedTarMaxStorageSpaceInMB: 1_000_000
cachedTarEvictionStorageSpaceThresholdInMB: 800_000
cachedTarSafeStorageSpaceThresholdInMB: 700_000
```

(suite sur la page suivante)

(suite de la page précédente)

```

maxAccessRequestSize: 10_000
readyAccessRequestExpirationDelay: 30
readyAccessRequestExpirationUnit: DAYS
readyAccessRequestPurgeDelay: 60
readyAccessRequestPurgeUnit: DAYS
accessRequestCleanupTaskIntervalDelay: 15
accessRequestCleanupTaskIntervalUnit: MINUTES

topology:
  ...
tapeLibraries:
  ...

```

Le paragraphe `topology` décrit 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 une archive TAR peut rester ouverte (durée maximale d'accumulation des objets dans un TAR) avant que le TAR soit finalisé / planifié pour écriture sur bande.

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

```

Note : Tous les tenants doivent être affectés à un et un seul bucket.

Prudence : L'affectation d'un tenant à un bucket est définitive. i.e. Il est impossible de modifier le bucket auquel un tenant a été déjà affecté car les données ont déjà été écrites sur bandes. Il est possible cependant, lors de l'ajout d'un tout nouveau tenant à Vitam, d'affecter ce nouveau tenant à un bucket existant.

La section `tapeLibraries` permet de définir le paramétrage des bibliothèques de bandes pilotées par l'offre froide.

Note : Une offre de stockage Vitam ne peut manipuler qu'une seule bibliothèque de bandes. Afin de piloter plusieurs bibliothèques de bandes, il convient d'utiliser des offres Vitam différentes.

Une bibliothèque de bandes est composée d'un robot (bras articulé), et d'un ensemble de lecteurs.

Note : Seul un robot doit être configuré pour piloter une librairie de bandes. La configuration de plusieurs robots pour une même librairie de bandes n'est actuellement PAS supportée.

La commande `ls -l /dev/tape/by-id/` permet de lister les chemins des périphériques (lecteurs et bras articulés) à configurer.

Exemple :

```
$ ls -l /dev/tape/by-id/
total 0
lrwxrwxrwx 1 root root  9 Mar  7 11:07 scsi-1HP_EML_E-Series_B4B0AC0000 -> ../../sg1
lrwxrwxrwx 1 root root  9 Mar  7 11:07 scsi-SHP_DLT_VS80_B4B0A00001 -> ../../st0
lrwxrwxrwx 1 root root 10 Mar  7 11:07 scsi-SHP_DLT_VS80_B4B0A00001-nst -> ../../nst0
lrwxrwxrwx 1 root root  9 Mar  7 11:07 scsi-SHP_DLT_VS80_B4B0A00002 -> ../../st1
lrwxrwxrwx 1 root root 10 Mar  7 11:07 scsi-SHP_DLT_VS80_B4B0A00002-nst -> ../../nst1
lrwxrwxrwx 1 root root  9 Mar  7 11:07 scsi-SHP_DLT_VS80_B4B0A00003 -> ../../st2
lrwxrwxrwx 1 root root 10 Mar  7 11:07 scsi-SHP_DLT_VS80_B4B0A00003-nst -> ../../nst2
lrwxrwxrwx 1 root root  9 Mar  7 11:07 scsi-SHP_DLT_VS80_B4B0A00004 -> ../../st3
lrwxrwxrwx 1 root root 10 Mar  7 11:07 scsi-SHP_DLT_VS80_B4B0A00004-nst -> ../../nst3
```

Prudence : Ne pas utiliser les chemins `/dev/*` dont l'index peut changer en cas de redémarrage. Utiliser les chemins `/dev/tape/by-id/*` (qui utilisent le numéro de série du device cible).

Prudence : Seuls les devices de lecteurs de type `/dev/nstX` (par exemple : `/dev/tape/by-id/scsi-SHP_DLT_VS80_B4B0A00001-nst -> /dev/nst0`) peuvent être utilisés dans Vitam. Les devices de lecteurs de type `/dev/stX` (par exemple : `/dev/tape/by-id/scsi-SHP_DLT_VS80_B4B0A00001 -> /dev/st0`) ne doivent PAS être utilisés (car ils causent à rebobinage automatique de la bande après chaque opération).

- **robots :** Définition du bras robotique de la librairie.
 - **device :** Chemin du fichier de périphérique scsi générique associé au bras. (ex. `/dev/tape/by-id/scsi-1HP_EML_E-Series_B4B0AC0000`)
 - **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. (ex. `/dev/tape/by-id/scsi-SHP_DLT_VS80_B4B0A00001-nst`)
 - **mtPath :** Chemin vers la commande Linux de manipulation des lecteurs.
 - **ddPath :** Chemin vers la commande Linux de copie de bloc de données.
 - **timeoutInMilliseconds :** timeout en millisecondes à appliquer aux ordres du lecteur.
- **fullCartridgeDetectionThresholdInMB** Seuil de détection de bande pleine (en Mo) Permet pour détecter en cas d'erreur d'écriture sur bande, la cause probable de l'erreur :
 - Si le volume des données écrites sur bande > seuil : La bande est considérée comme pleine
 - Si le volume des données écrites sur bande < seuil : La bande est considérée comme corrompueTypiquement, 90% de la capacité théorique de stockage des cartouches (hors compression).

Exemple :

```

tapeLibraries:
  TAPE_LIB_1:
    robots:
      -
        device: /dev/tape/by-id/scsi-1HP_EML_E-Series_B4B0AC0000
        mtPath: "/usr/sbin/mtx"
        timeoutInMilliseconds: 3600000
    drives:
      -
        index: 0
        device: /dev/tape/by-id/scsi-SHP_DLT_VS80_B4B0A00001-nst
        mtPath: "/bin/mt"
        ddPath: "/bin/dd"
        timeoutInMilliseconds: 3600000
      -
        index: 1
        device: /dev/tape/by-id/scsi-SHP_DLT_VS80_B4B0A00002-nst
        mtPath: "/bin/mt"
        ddPath: "/bin/dd"
        timeoutInMilliseconds: 3600000
      -
        index: 2
        device: /dev/tape/by-id/scsi-SHP_DLT_VS80_B4B0A00003-nst
        mtPath: "/bin/mt"
        ddPath: "/bin/dd"
        timeoutInMilliseconds: 3600000
      -
        index: 3
        device: /dev/tape/by-id/scsi-SHP_DLT_VS80_B4B0A00004-nst
        mtPath: "/bin/mt"
        ddPath: "/bin/dd"
        timeoutInMilliseconds: 3600000

    fullCartridgeDetectionThresholdInMB : 2_000_000

```

4.2.5.15 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 user's 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'aux 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

4.2.5.16 Installation de la stack Prometheus

Note : Si vous disposez d'un serveur Prometheus et alertmanager, vous pouvez installer uniquement les exporters souhaités.

Prometheus server et alertmanager sont des addons dans la solution *VITAM*.

Voici à quoi correspond une configuration qui permettra d'installer toute la stack prometheus.

```
prometheus:
  metrics_path: /admin/v1/metrics
  check_consul: 10 # in seconds
  prometheus_config_file_target_directory: # Set path where "prometheus.yml" file_
  ↳ will be generated. Example: /tmp/
  server:
    port: 9090
    tsdb_retention_time: "7d"
    tsdb_retention_size: "5GB"
  node_exporter:
    enabled: true
    port: 9101
    metrics_path: /metrics
  consul_exporter:
    enabled: true
```

(suite sur la page suivante)

(suite de la page précédente)

```

port: 9107
metrics_path: /metrics
elasticsearch_exporter:
  enabled: true
  port: 9114
  metrics_path: /metrics
alertmanager:
  api_port: 9093
  cluster_port: 9094

```

- L'adresse d'écoute de ces composants est celle de la patte d'administration.
- Vous pouvez surcharger la valeur de certaines de ces variables (Par exemple le port d'écoute, le path de l'API).
- Pour générer uniquement le fichier de configuration prometheus.yml à partir du fichier d'inventaire de l'environnement en question, il suffit de spécifier le répertoire destination dans la variable `prometheus_config_file_target_directory`

4.2.5.16.1 Playbooks ansible

Veillez vous référer à la documentation d'exploitation pour plus d'information.

- Installer prometheus et alertmanager

```

ansible-playbook ansible-vitam-extra/prometheus.yml -i environments/hosts.
↪ <environnement> --ask-vault-pass

```

- Générer le fichier de conf prometheus.yml dans le dossier prometheus_config_file_target_directory

```

ansible-playbook ansible-vitam-extra/prometheus.yml -i environments/hosts.
↪ <environnement> --ask-vault-pass

```

```

--tags gen_prometheus_config ..

```

4.2.5.17 Installation de Grafana

Note : Si vous disposez déjà d'un Grafana, vous pouvez l'utiliser pour l'interconnecter au serveur Prometheus.

Grafana est un addon dans la solution *VITAM*.

Grafana sera déployé sur l'ensemble des machines renseignées dans le groupe `[hosts_grafana]` de votre fichier d'inventaire.

Pour se faire, il suffit d'exécuter le playbook associée :

```

ansible-playbook ansible-vitam-extra/grafana.yml -i environments/hosts.<environnement>
↪ --ask-vault-pass

```

4.2.5.17.1 Configuration

Les paramètres de configuration de ce composant se trouvent dans le fichier `environments/group_vars/all/advanced/cots_vars.yml`. Vous pouvez adapter la configuration en fonction de vos besoins.

4.2.5.17.2 Configuration spécifique derrière un proxy

Si Grafana est déployé derrière un proxy, vous devez apporter des modifications au fichier de configuration `ansible-vitam-extra/roles/grafana/templates/grafana.ini.j2`

Voici les variables modifiées par la solution *VITAM* pour permettre le fonctionnement de Grafana derrière un proxy apache.

```
[server]
root_url = http://{{ ip_admin }}:{{ grafana.http_port | default(3000) }}/grafana
serve_from_sub_path = true

[auth.basic]
enabled = false
```

Avertissement : Lors de la première connexion, vous devrez changer le mot de passe par défaut (login : admin ; password : aadmin1234), configurer le datasource et créer/importer les dashboards manuellement.

4.2.5.18 Installation de restic

restic est un add-on (beta) de la solution *VITAM*.

restic sera déployé sur l'ensemble des machines du groupe `[hosts_storage_offer_default]` qui possèdent le paramètre `restic_enabled=true`. Attention à ne renseigner qu'une seule fois ce paramètre par `offer_conf`.

Pour se faire, il suffit d'exécuter le playbook associé :

```
ansible-playbook --vault-password-file vault_pass.txt ansible-vitam-extra/restic.yml -
↪i environments/hosts.<environnement>
```

4.2.5.18.1 Configuration

Les paramètres de configuration de ce composant se trouvent dans les fichiers `environments/group_vars/all/advanced/cots_vars.yml` et `environments/group_vars/all/main/vault-cots.yml`. Vous pouvez adapter la configuration en fonction de vos besoins.

4.2.5.18.2 Limitations actuelles

restic est fourni en tant que fonctionnalité beta. À ce titre, il ne peut se substituer à des vérifications régulières de l'état des sauvegardes de vos bases.

restic ne fonctionne pas avec les providers *openstack-swift*, *openstack-swift-v2* et *tape-library*.

restic ne fonctionne pas avec un cluster mongo multi-shardé. Ainsi, mongo-data ne peut être sauvegardé via restic que dans de petites instances de Vitam.

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](#)¹⁸ et [Section Virus Database](#)¹⁹

4.2.6.1.2 Fichier de mot de passe des vaults ansible

Par défaut, le mot de passe des `vault` sera demandé à chaque exécution d'ansible avec l'utilisation de l'option `--ask-vault-pass` de la commande `ansible-playbook`.

Pour simplifier l'exécution des commandes `ansible-playbook`, vous pouvez utiliser un fichier `repertoire_deploiement/vault_pass.txt` contenant le mot de passe des fichiers vault. Ainsi, vous pouvez utiliser l'option `--vault-password-file=vault_pass.txt` à la place de l'option `--ask-vault-pass` dans les différentes commandes de cette documentation.

Avertissement : Il est déconseillé de conserver le fichier `vault_pass.txt` sur la machine de déploiement ansible car ce fichier permet d'avoir accès à l'ensemble des secrets de *VITAM*.

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 `repertoire_inventory/group_vars/all/main/repositories.yml` à partir du modèle suivant (décommenter également les lignes) :

```

1 ---
2
3 # vitam_repositories:
4 #   - key: repol # Mandatory: Only on CentOS/AlmaLinux
5 #     value: http://path_to_repol # Mandatory: Path to the repository
6 #     gpgcheck: 1 # Optionnal: Default to 0 (equivalent as [trusted=yes] on Debian)
7 #     gpgkey: path_to_custom_key # Optionnal: Only if gpgcheck is enabled; Default to
8 #       ↳ official Vitam GPG Key
9 #     subtree: "./" # Optionnal: Only on Debian; Default to ./
#     proxy: http://proxy_url # Optionnal: Only on CentOS/AlmaLinux; Default to _none_

```

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/hosts.
↳ <environnement> --ask-vault-pass

```

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

19. <https://www.clamav.net/downloads>

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 `repertoire_playbook_ansible/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-exploitation/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 `repertoire_inventory/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 Tests d'infrastructure

Il est possible de lancer une série de tests d'infrastructure en amont du déploiement, ceci afin de se prémunir d'éventuelles erreurs durant l'installation.

Les tests sont basés sur des prérequis de la solution *VITAM* et sont génériques. De ce fait, des « faux-positifs » peuvent être remontés dû à une configuration spécifique de votre environnement. Il est à votre charge d'analyser le rapport à l'issue des tests et de juger de la pertinence des résultats.

Les tests sont les suivants :

- Version d'Ansible
- Accès aux recursors (serveurs DNS)
- Présence de Java
- Accès aux repositories
- Accès aux offres objet

Comme pour le déploiement, les tests s'effectuent depuis la machine *ansible*. La commande pour les effectuer est la suivante :

```
ansible-playbook ansible-vitam/checks_infra.yml -i environments/hosts.<environnement>_
↪--ask-vault-pass
```

4.2.6.1.6 Déploiement

Une fois les étapes précédentes correctement effectuées (en particulier, la section *Génération des magasins de certificats* (page 69)), 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/hosts.<environnement> --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).

Note : Il est possible d'effectuer les tests d'infrastructure décrits dans la partie précédente en ajoutant le paramètre `-e checks_infra=yes`. Un rapport s'affichera à l'issue des tests et il sera donné la possibilité de poursuivre ou non le déploiement.

Note : Il est également possible de forcer la suppression de profils de sécurité et de leurs données associées (contextes applicatifs et certificats) en ajoutant le paramètre `-e delete_security_profiles=yes`. Cela peut éventuellement être utile dans le cas d'un nouveau lancement de l'installation suite à un échec.

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 Éléments *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 `|repertoire_inventory|'group_vars/all/advanced/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     elasticsearch_mapping_dir: "{{ vitam_defaults.folder.root_path }}/conf/ihm-
↪recette/mapping"
21     library:
22       vitam_component: library
23       host: "library.service.{{ consul_domain }}"
24       port_service: 8090
25       port_admin: 28090
26       baseuri: "doc"
27       https_enabled: false
28       secret_platform: "false"
29       consul_business_check: 30 # value in seconds
30       consul_admin_check: 30 # value in seconds
31
32 tenant_to_clean_before_tnr: ["0", "1"]
33
34 # Period units in seconds
35 metricbeat:
36   enabled: false
37   system:
38     period: 10

```

(suite sur la page suivante)

(suite de la page précédente)

```

39  mongodb:
40      period: 10
41  elasticsearch:
42      period: 10
43
44  packetbeat:
45      enabled: false
46
47  browser:
48      enabled: false
49
50  docker_opts:
51      registry_httponly: yes
52      vitam_docker_tag: latest
53      ## Custom CIDR address for docker bridge networks
54      # docker_bip: 192.168.191.1/24
55      ## Custom CIDR address settings for docker internal networks
56      # docker_address_pools_cidr: 192.168.192.1/18
57      # docker_address_pools_size: 24
58
59  gatling_install: false
60  docker_install: false # whether or not install docker & docker images

```

Avertissement : À 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/advanced/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 `!repertoire_inventory!group_vars/all/main/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/hosts.  
↪ <environnement> --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/hosts.<environnement> -  
↪ -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`, et dans toutes les directives concernées (ex. `api_output_index_tenants`, `rules_index_tenants`, `vitam_removed_tenants`, `dedicated_tenants`, `grouped_tenants`...)

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

Note : Si une offre froide est configurée, la liste des buckets configurés doit être mise à jour en conséquence.

5.2 Cas d'une modification des paramètres JVM

Se référer à *Tuning JVM* (page 69)

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 ansible-vitam/vitam.yml -i environments/hosts.<environnement> --ask-  
↪ vault-pass --tags update_jvmoptions_vitam  
ansible-playbook ansible-vitam-extra/extra.yml -i environments/hosts.<environnement> -  
↪ -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/main/main.yml`.

Note : Dans le cas d'une montée de version des composants *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 `repertoire_inventory/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 pronom²⁰](#)) 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 discussion²¹](#), 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)

20. <http://www.nationalarchives.gov.uk/aboutapps/pronom/droid-signature-files.htm>

21. <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 :

- Centos²²
- Debian²³

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`):

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

23. <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](#)²⁴.

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](#)²⁵. 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}'
```

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

25. <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`²⁶ et `mongotop`²⁷.

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 dus à la réplication](#)²⁸ :

```
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`²⁹) :

```
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 :

26. <https://docs.mongodb.com/manual/reference/program/mongostat/>
 27. <https://docs.mongodb.com/manual/reference/program/mongotop/>
 28. <https://docs.mongodb.com/manual/tutorial/troubleshoot-replica-sets>
 29. <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.RequestAuthorizationValidator.  
↪checkTimestamp(AuthorizationWrapper.java:102) : [vitam-env-int8-app-04.vitam-  
↪env:storage:239079175] Timestamp check failed. 16s  
fr.gouv.vitam.common.security.filter.RequestAuthorizationValidator.  
↪checkTimestamp(AuthorizationWrapper.java:107) : [vitam-env-int8-app-04.vitam-  
↪env:storage:239079175] Critical timestamp check failure. 61s
```

Il faut vérifier / corriger l'heure des machines hébergeant la solution logicielle *VITAM*. .. caution : : Si un *delta* de temps important (10s par défaut) a été détecté entre les machines, des erreurs sont tracées dans les logs et une alerte est remontée dans le dashboard Kibana des Alertes de sécurité. Au delà d'un seuil critique (60s par défaut) d'écart de temps entre les machines, les requêtes sont systématiquement rejetées, ce qui peut causer des dysfonctionnements majeurs de la solution.

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

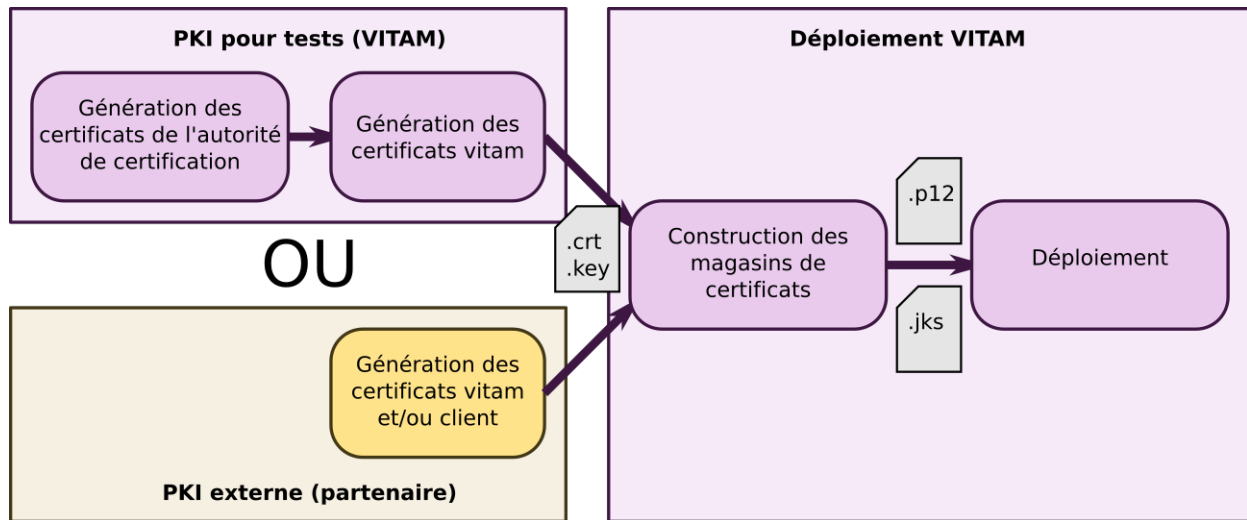


FIG. 1 – Vue d'ensemble de la gestion des certificats au déploiement

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 126)

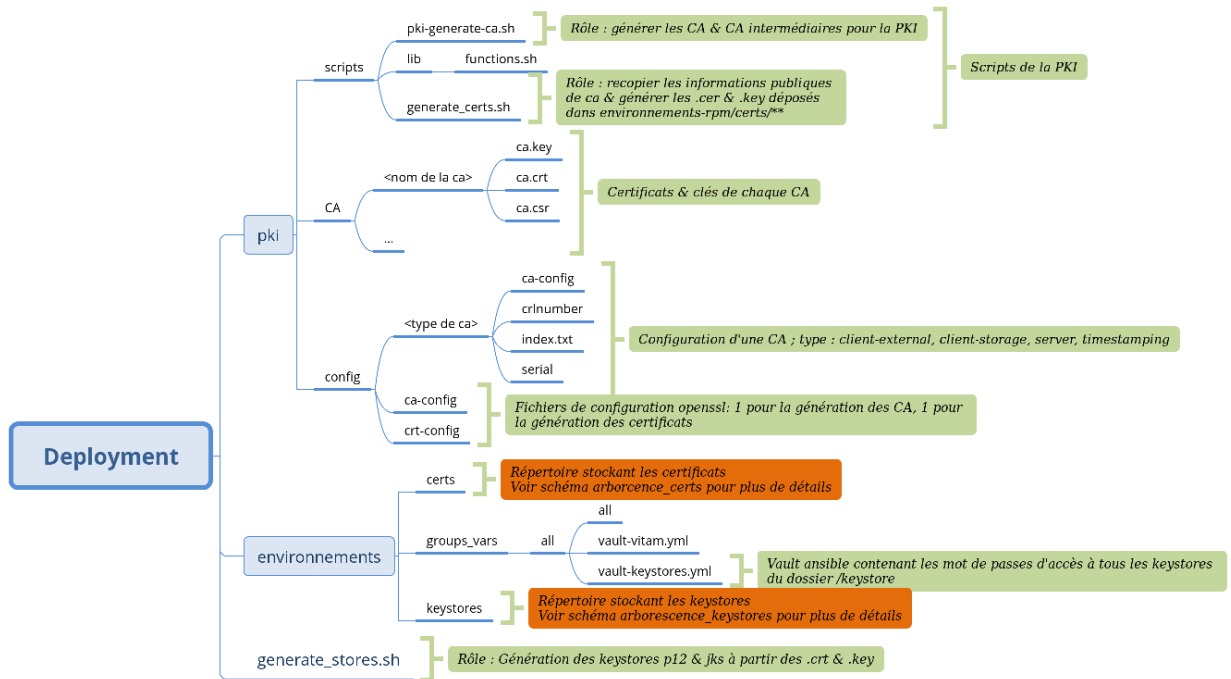


FIG. 2 – Vue l'arborescence de la PKI Vitam

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

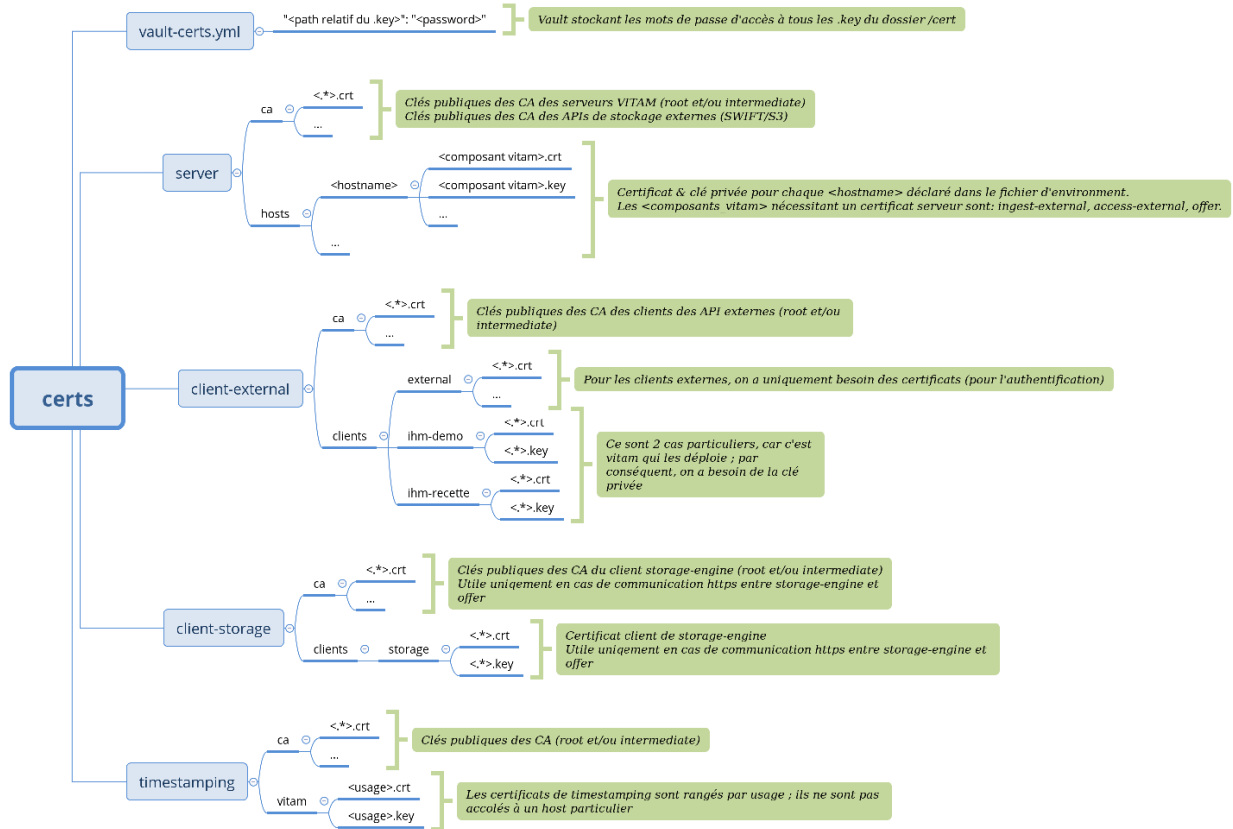


FIG. 3 – Vue détaillée de l'arborescence des certificats

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

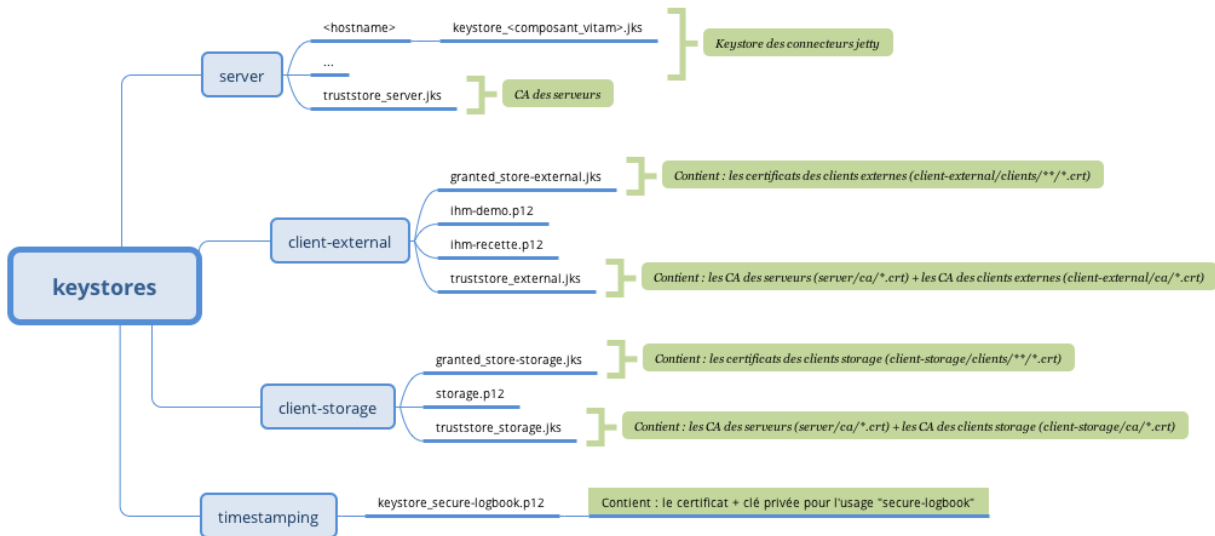


FIG. 4 – Vue détaillée de l'arborescence des keystores

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 bon 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 125)).

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/advanced/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	15
2	Vue détaillée des certificats entre le storage et l'offre en multi-site	21
3	Vue détaillée de l'arborescence des certificats	67
1	Vue d'ensemble de la gestion des certificats au déploiement	123
2	Vue l'arborescence de la <i>PKI</i> Vitam	124
3	Vue détaillée de l'arborescence des certificats	125
4	Vue détaillée de l'arborescence des keystores	126

Liste des tableaux

1	Documents de référence VITAM	2
1	Matrice de compétences	7
1	Description des identifiants de référentiels	74
2	Description des règles	75

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

TNA, 5

TNR, 5

TTL, 5

U

UDP, 5

UID, 5

V

VITAM, 5

VM, 5

W

WAF, 5

WAN, 5