# VITAM - Documentation d'installation

*Version 0.10.0-SNAPSHOT*

**VITAM**

**21 oct. 2016**

Table des matières

**Prudence :** Cette documentation est un travail en cours ; elle est susceptible de changer de manière conséquente.

CHAPITRE 1

# Introduction

## 1.1 But de cette documentation

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

## 1.2 Destinataires de ce document

Ce document s'adresse à des exploitants du secteur informatique ayant de bonnes connaissances en environnement Linux.

## 1.3 Information concernant les licences

Le logiciel *VITAM* est publié sous la license CeCILL 2.1 [1] ; la documentation associée (comprenant le présent document) est publiée sous license CC-BY-SA 3.0 [2].

## 1.4 Documents de référence

### 1.4.1 Documents internes

Tableau 1.1 – Documents de référence VITAM

| Nom | Lien |
|---|---|
| *DAT* | (à renseigner) |
| *DIN* | (à renseigner) |
| *DEX* | (à renseigner) |
| Release notes | (à renseigner) |

1. http://www.cecill.info/licences/Licence_CeCILL_V2.1-fr.html
2. https://creativecommons.org/licenses/by-sa/3.0/fr/legalcode

### 1.4.2 Référentiels externes

**Référentiel Général d'Interopérabilité [RGI]** V1.0 du 12 juin 2009 approuvé par arrêté du Premier ministre du 9 novembre 2009

Règles d'interopérabilité (format, protocoles, encodages, etc.) rentrant dans le champ d'application de l'ordonnance n°2005-1516 du 8 décembre 2005 relative aux échanges électroniques entre les usagers et les autorités administratives et entre les autorités administratives.

https://references.modernisation.gouv.fr/rgi-interoperabilite

**Référentiel Général de Sécurité [RGS]** V2.0 du 13 juin 2014 approuvé par arrêté du Premier ministre du 13 juin 2014

Le RGS précise les règles de sécurité s'imposant aux autorités administratives dans la sécurisation de leur SI et notamment sur les dispositifs de sécurité relatifs aux mécanismes cryptographiques et à l'utilisation de certificats électroniques et contremarques de temps. Le RGS propose également des bonnes pratiques en matière de SSI. Le RGS découle de l'application de l'ordonnance n°2005-1516 du 8 décembre 2005 relative aux échanges électroniques entre les usagers et les autorités administratives et entre les autorités administratives.

https://references.modernisation.gouv.fr/rgs-securite

**Norme OAIS (ISO 14721 :2012 – 1 septembre 2012)** Systèmes de transfert desinformations et données spatiales – Système ouvert d'archivage d'information (SOAI) - Modèle de référence

**Standard d'échange de données pour l'archivage (SEDA)** Transfert, communication, élimination, restitution, modification – Version 1.0 – Septembre 2012

Cadre normatif pour les différents échanges d'informations entre les services d'archives publics et leurs partenaires : entités productrices des archives, entités gestionnaires, entités de contrôle des processus, et enfin entités qui utilisent ces archives. Il concerne également les échanges entre plusieurs services d'archives (services publics d'archives, prestataires d'archivage, archivage intermédiaire, archivage définitif).

http://www.archivesdefrance.culture.gouv.fr/seda/

## 1.5 Glossaire

**COTS** Component Off The Shelves ; 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.

**DIN** Dossier d'Installation

**DEX** Dossier d'EXploitation

**DAT** Dossier d'Architecture Technique

**IHM** Interface Homme Machine

**VITAM** Valeurs Immatérielles Transférées aux Archives pour Mémoire

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

**API** Application Programming Interface

**BDD** Base De Données

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

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

**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.

**NoSQL** Base de données non-basée sur un paradigme classique des base relationnelles. Référence [3]

**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. Référence [4]

**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). Référence [5]

3. https://fr.wikipedia.org/wiki/NoSQL
4. https://fr.wikipedia.org/wiki/Attaque_de_l'homme_du_milieu
5. https://fr.wikipedia.org/wiki/Domain_Name_System_Security_Extensions

CHAPITRE 2

# Architecture de la solution logicielle VITAM

Le schéma ci-dessous représente une solution *VITAM* :

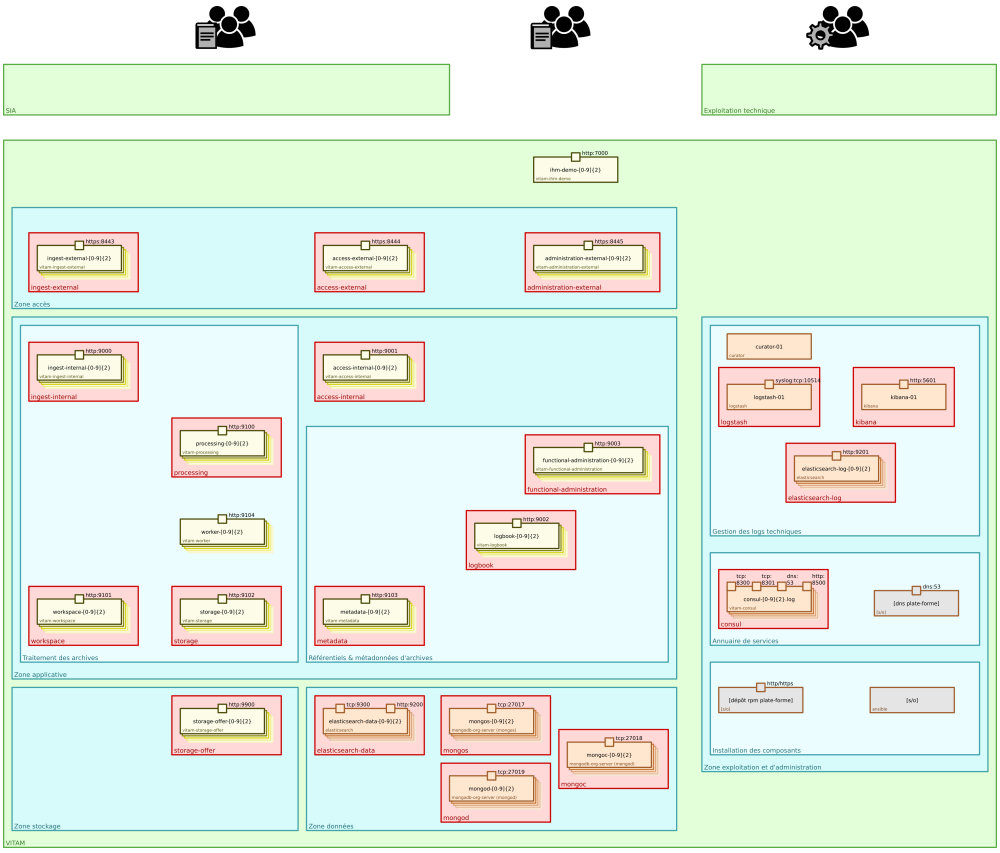


Fig. 2.1 – Vue d'ensemble d'un déploiement VITAM : zones, composants

**Voir aussi :**

Se référer au *DAT* (et notamment le chapitre dédié à l'architecture technique) pour plus de détails, en particulier concernant les flux entre les composants.

CHAPITRE 3

# Pré-requis

## 3.1 Description

- Plate-forme Linux CentOS 7
- Packages VITAM (au moment de la rédaction du document, aucun formalisme n'a été défini pour ce point.)
- Solution de déploiement vitam

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

- ansible (version 2.0.2 minimale et conseillée)
- présence du package openssh-clients (client SSH utilisé par ansible)
- un accès ssh vers un utilisateur d'administration avec élévation de privilèges vers les droits root sur les serveurs cibles
- Le compte utilisé sur le serveur d'administration doit avoir confiance dans les serveurs cibles (fichier ~/.ssh/authorized_keys rempli)
- il est vivement conseillé d'avoir configuré une authentification ssh par certificat vers les serveurs cibles
- présence du package java-1.8.0-openjdk & openssl (du fait de la génération de certificats / stores, l'utilitaire `keytool` est nécessaire)
- tous les serveurs cibles doivent avoir accès au registry docker vitam (docker.programmevitam.fr) => A commenter, car cela n'est nécessaire que dans une installation docker.

## 3.2 Matériel

Les prérequis matériel sont définis dans le *DAT* ; à l'heure actuelle, le minimum recommandé pour la solution Vitam est 2 CPUs et 512Mo de RAM disponible par composant applicatif installé sur chaque machine.

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

- default-offer
- solution de centralisation des logs (elasticsearch)
- workspace
- elasticsearch des données Vitam

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

CHAPITRE 4

# Dépendances aux services d'infrastructures

## 4.1 Ordonnanceurs techniques / batchs

Sans objet pour cette version de VITAM.

**Note :** Curator permet d'effectuer des opérations périodiques de maintenance sur les index elasticsearch ; cependant, il gère lui-même le déclenchement de ses actions, et ne nécessite donc pas la configuration d'un ordonnanceur externe.

**Note :** Des batchs d'exploitation seront disponibles dans les versions ultérieures de la solution VITAM (ex : validation périodique de la validité des certificats clients)

## 4.2 Socles d'exécution

### 4.2.1 OS

- CentOS 7

**Prudence :** SELinux doit être configuré en mode `permissive` ou `disabled`

### 4.2.2 Middlewares

- Java : JRE 8 ; les versions suivantes ont été testées :
  - OpenJDK 1.8.0, dans la version présente dans les dépôts officiels CentOS 7 au moment de la parution de la version VITAM (actuellement : 1.8.0.101)

CHAPITRE 5

# Fiche type de déploiement VITAM

## 5.1 Fiche-type VITAM

**Prudence :** cette liste a pour but d'évoluer et s'étoffer au fur et à mesure des mises à jour des composants et du contenu des fichiers de déploiement de VITAM.

Tableau 5.1 – Tableau récapitulatif des informations à renseigner pour VITAM

| Nom du omposant | Descriptif | Valeur d'exemple | Valeur choisie | Si HA ? |
|---|---|---|---|---|
| IHM-demo machine | interface web | vitam-prod-app-1.internet.agri | | |
| ingest-external machine | interface web | vitam-prod-app-1.internet.agri | | |
| ingest-internal machine | interface web | vitam-prod-app-1.internet.agri | | |
| access-external machine | interface web | vitam-prod-app-1.internet.agri | | |
| access-internal machine | interface web | vitam-prod-app-1.internet.agri | | |
| logbook machine | interface web | vitam-prod-app-1.internet.agri | | |
| metadata machine | interface web | vitam-prod-app-1.internet.agri | | |
| mongodb machine(s) | base de données | vitam-prod-app-1.internet.agri | | |
| processing machine(s) | base de données | vitam-prod-app-1.internet.agri | | |
| storage-engine machine(s) | xxxx | vitam-prod-app-1.internet.agri | | |
| storage-offer-default machine(s) | implémentation de pilote de stockage | vitam-prod-app-1.internet.agri | | |
| Consul servers | implémentation de Consul pour un DNS applicatif (nécessite 3 serveurs minimum ; règle (2*n+1) ) | vitam-prod-app-1.internet.agri, vitam-prod-app-2.internet.agri, vitam-prod-app-3.internet.agri | | |
| elasticsearch data machine(s) | Cluster ElasticSearch de données VITAM (3 machines) | vitam-prod-ela-1.internet.agri,vitam-prod-ela-2.internet.agri,vitam-prod-ela-3.internet.agri | | |
| log central machine(s) | Centralisation des logs | vitam-prod-log-1.internet.agri | | |

**À faire**

ajouter section issue du DAT sur les préconisations de colocalisation, ... et nombre de machines pour chaque composant.

CHAPITRE 6

# Récupération de la version

Au moment de la rédaction du document, aucun formalisme n'a été défini pour ce point.

CHAPITRE 7

# Procédures d'installation / mise à jour : Package RPM

## 7.1 Pré-requis supplémentaire

Tous les serveurs cible doivent avoir accès aux dépôts rpm contenant les paquets des logiciels 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.

## 7.2 Procédures

### 7.2.1 Configuration de sécurité

En fonction de la méthode d'authentification sur les serveurs et d'élevation 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

#### Authentification du compte utilisateur utilisé pour la connexion SSH

Pour le login du compte utilisateur, voir le paragraphe décrivant le fichier d'inventaire

#### 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.

#### Par login/mot de passe

Dans le cas d'une authentificatioon 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

#### Par clé SSH sans passphrase

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

#### 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.

#### Elevation de privilèges

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

#### Par sudo avec mot de passe

Dans ce cas, il faut rajouter les options –become-method=sudo –ask-sudo-pass

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

#### 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

#### Par sudo sans mot de passe

Dans ce cas, il faut rajouter l'option –become-method=sudo

#### Déjà Root

Dans ce cas, il n'y a pas de paramétrages supplémentaire

### 7.2.2 PKI

#### Action préalable

Le fichier `environnements-rpm/group_vars/all/vault.yml` a été généré avec un mot de passe ; le changer par la commande :

```
ansible-vault rekey environnements-rpm/group_vars/all/vault.yml
```

Puis éditer le fichier `environnements-rpm/<votre fichier d'inventaire>` et le mettre en conformité de l'environnement souhaité.

#### Génération des autorités de certification

#### Cas d'une PKI inexistante

Dans le répertoire de déploiement, lancer le script : `./pki-generate-ca.sh`

Ce script génère sous `PKI/CA` les certificats CA et intermédiaires pour client et server.

---

**Note :** bien noter les dates de création et de fin de validité des CA.

---

**Prudence :** 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.

#### Cas d'une PKI existante

Si le client possède déjà une PKI, ou ne compte pas utilise la PKI fournie par VITAM, il convient de positionner les fichiers `ca.crt` et `ca.key` sous PKI/CA/<usage>‘‘, où usage est :

- server
- server_intermediate
- client
- client_intermediate

---

**À faire**

droits Unix à vérifier

---

#### Génération des certificats

#### Cas de certificats inexistants

**Avertissement :** cette étape n'est à effectuer que pour les clients ne possédant pas de certificats.

Editer le fichier `environnements-rpm/group_vars/all/vault.yml` ( qui est un fichier protégé par mot de passe ) pour indiquer les mots de passe nécessaires.

Editer le fichier `environnements-rpm/<inventaire>` pour indiquer les serveurs associé à chaque service.

Puis, dans le répertoire de déploiement, lancer le script : `./generate_certs <environnement>`

---

**Note :** Ce script utilise le fichier `environnements-rpm/group_vars/all/vault.yml`. Le mot de passe de ce fichier sera demandé plusieurs fois et génèrera des certificats et stores adéquats au contenu du fichier yml.

---

Ce script génère sous `PKI/certificats` les certificats (format p12) nécessaires pour un bon fonctionnement dans VITAM.

**Prudence :** Les certificats générés à l'issue ont une durée de validité de (à vérifier).

### Cas de certificats déjà créés par le client

---

**À faire**

procédure à écrire

---

### Génération des stores

Editer le fichier `environnements-rpm/group_vars/all/vault.yml` ( qui est un fichier protégé par mot de passe ) pour indiquer les mots de passe nécessaires.

Editer le fichier `environnements-rpm/<inventaire>` pour indiquer les serveurs associé à chaque service.

Puis, dans le répertoire de déploiement, lancer le script : `./generate_stores.sh <environnement>`

---

**Note :** Ce script utilise le fichier `environnements-rpm/group_vars/all/vault.yml`. Le mot de passe de ce fichier sera demandé plusieurs fois et génèrera des certificats et stores adéquats au contenu du fichier yml.

---

Ce script génère sous `PKI/certificats` les les stores (jks) associés pour un bon fonctionnement dans VITAM.

### Recopie des bons fichiers dans l'ansiblerie

Dans le répertoire de déploiement, lancer le script : `./copie_fichiers_vitam.sh <environnement>`

Ce script recopie les fichiers nécessaires (certificats, stores) aux bons endroits de l'ansiblerie (sous `ansible-vitam-rpm/roles/vitam/files/<composant>`).

### Cas des SIA

Pour le moment, la prise en charge des certificats des SIA n'est pas effective.

### 7.2.3 Première installation

Les fichiers de déploiement sont disponibles dans la version VITAM livrée dans le sous-répertoire `deployment` . Ils consistent en 2 parties :

- le playbook ansible, présent dans le sous-répertoire `ansible-vitam-rpm`, qui est indépendant de l'environnement à déployer
- les fichiers d'inventaire (1 par environnement à déployer) ; des fichiers d'exemple sont disponibles dans le sous-répertoire `environments-rpm`

#### Configuration du déploiement

#### Informations "plate-forme"

Pour configurer le déploiement, il est nécessaire de créer (dans n'importe quel répertoire en dehors du répertoire `environments-rpm` un nouveau fichier d'inventaire comportant les informations suivantes :

```
# Group definition ; DO NOT MODIFY
[hosts]

# Group definition ; DO NOT MODIFY
[hosts:children]
vitam
reverse
library
hosts-mongo-express


########### Tests environments specifics ###########

# EXTRA : Front reverse-proxy (test environments ONLY) ; add machine name after
[reverse]


########### Extra VITAM applications ###########

[library]
# TODO: Put here servers where this service will be deployed : library

[hosts-mongo-express]
# TODO: Put here servers where this service will be deployed : mongo-express

########### VITAM services ###########

# Group definition ; DO NOT MODIFY
[vitam:children]
zone-external
zone-access
zone-applicative
zone-storage
zone-data
zone-admin


##### Zone externe


```

```
[zone-external:children]
hosts-ihm-demo

[hosts-ihm-demo]
# TODO: Put here servers where this service will be deployed : ihm-demo


##### Zone access

# Group definition ; DO NOT MODIFY
[zone-access:children]
hosts-ingest-external
hosts-access-external

[hosts-ingest-external]
# TODO: Put here servers where this service will be deployed : ingest-external


[hosts-access-external]
# TODO: Put here servers where this service will be deployed : access-external


##### Zone applicative

# Group definition ; DO NOT MODIFY
[zone-applicative:children]
hosts-ingest
hosts-processing
hosts-worker
hosts-access
hosts-metadata
hosts-functional-administration
hosts-logbook
hosts-workspace
hosts-storage-engine

[hosts-logbook]
# TODO: Put here servers where this service will be deployed : logbook


[hosts-workspace]
# TODO: Put here servers where this service will be deployed : workspace


[hosts-ingest]
# TODO: Put here servers where this service will be deployed : ingest-internal


[hosts-access]
# TODO: Put here servers where this service will be deployed : access-internal


[hosts-metadata]
# TODO: Put here servers where this service will be deployed : metadata


[hosts-functional-administration]
# TODO: Put here servers where this service will be deployed : functional-
↪administration
```

```
99
100
101 [hosts-processing]
102 # TODO: Put here servers where this service will be deployed : processing
103
104
105 [hosts-storage-engine]
106 # TODO: Put here servers where this service will be deployed : storage-engine
107
108
109 [hosts-worker]
110 # TODO: Put here servers where this service will be deployed : worker
111
112
113 ##### Zone storage
114
115 [zone-storage:children] # DO NOT MODIFY
116 hosts-storage-offer-default
117
118
119 [hosts-storage-offer-default]
120 # TODO: Put here servers where this service will be deployed : storage-offer-default
121
122
123
124 ##### Zone data
125
126 # Group definition ; DO NOT MODIFY
127 [zone-data:children]
128 hosts-elasticsearch-data
129 mongo_common
130
131
132 [hosts-elasticsearch-data]
133 # TODO: Put here servers where this service will be deployed : elasticsearch-data␣
    ↪cluster
134
135
136 # Group definition ; DO NOT MODIFY
137 [mongo_common:children]
138 mongos
139 mongoc
140 mongod
141
142 [mongos]
143 # TODO: Put here servers where this service will be deployed : mongos cluster ; add␣
    ↪after name shard_id=0
144 # Example : vitam-iaas-mongos-01.int shard_id=0
145
146 [mongoc]
147 # TODO: Put here servers where this service will be deployed : mongoc cluster
148
149
150 [mongod] # mongod declaration ; add machines name after ; add after shard_id=0 & rs_
    ↪member_id=<increasing number, starting from 0, for each line>
151 # TODO: Put here servers where this service will be deployed : mongod cluster ; add␣
    ↪after name shard_id=0
152 # Example : vitam-iaas-db-01.int rs_member_id=0 shard_id=0
```

```
# Example : vitam-iaas-db-02.int rs_member_id=1 shard_id=0
# Example : vitam-iaas-db-03.int rs_member_id=2 shard_id=0

###### Zone admin

# Group definition ; DO NOT MODIFY
[zone-admin:children]
hosts-consul-server
hosts-log-server
hosts-elasticsearch-log

[hosts-consul-server]
# TODO: Put here servers where this service will be deployed : consul


[hosts-log-server]
# TODO: Put here servers where this service will be deployed : log-server (kibana/
 ↪logstash)


[hosts-elasticsearch-log]
# TODO: Put here servers where this service will be deployed : elasticsearch-log␣
 ↪cluster




########### Global vars ###########

[hosts:vars]
# Declare user for ansible on target machines
ansible_ssh_user=

# Can target user become as root ? true/false
ansible_become=

# DEPRECATED
# TODO: remove
local_user=

# Environment (defines consul environment name ; in extra on homepage)
environnement=

# For extra ; used when VITAM is under a reverse proxy (provides configuration for␣
 ↪reverse proxy && displayed in header page)
vitam_reverse_domain=

# DEPRECATED
# TODO : remove
vitam_ihm_demo_external_dns=

# Version that has to be deployed (defined in the release note)
# Example: rpm_version=0.9.0-RC1*
rpm_version=

# Configuration for Curator
#        Days before deletion on log management cluster; 365 for production␣
 ↪environment
days_to_delete=
```

```
207
208 #        Days before claosing "old" indexes on log management cluster; 30 for␣
    ↪production environment
209 days_to_close=
210
211 #        Days before deletion for topbeat index only on log management cluster; 365␣
    ↪for production environment
212 days_to_delete_topbeat=
213
214 # Related to Consul ; apply in a table your DNS server(s)
215 # Example : dns_servers=["8.8.8.8","8.8.4.4"]
216 dns_servers=
```

Pour chaque type de "host" (lignes 2 à 176), indiquer le(s) serveur(s) défini(s) pour chaque fonction. Une colocalisation de composants est possible.

**Avertissement :** indiquer les contre-indications !

Ensuite, dans la section `hosts:vars` (lignes 179 à 216), renseigner les valeurs comme décrit :

A titre informatif, le positionnement des variables ainsi que des dérivations des déclarations de variables sont effectuées sous `environments-rpm /group_vars/all/all`, comme suit :

```
1  ---
2
3  vitam_folder_root: /vitam
4  docker_registry_httponly: yes
5  vitam_docker_tag: latest
6  port_http_timeout: 99999999
7
8  syslog_facility: local0
9
10 # Composants colocalisés
11 vitam_access_host: "access.service.{{consul_domain}}"
12 vitam_access_port: 8101
13 vitam_access_baseurl: "http://{{vitam_access_host}}:{{vitam_access_port}}"
14
15 vitam_ingest_host: "ingest-internal.service.{{consul_domain}}"
16 vitam_ingest_port: 8100
17 vitam_ingest_baseurl: "http://{{vitam_ingest_host}}:{{vitam_ingest_port}}"
18
19 vitam_metadata_host: "metadata.service.{{consul_domain}}"
20 vitam_metadata_port: 8200
21 vitam_metadata_baseurl: "http://{{vitam_metadata_host}}:{{vitam_metadata_port}}"
22
23 vitam_ihm_demo_host: "{{groups['hosts-ihm-demo'][0]}}"
24 vitam_ihm_demo_port: 8002
25 vitam_ihm_demo_baseurl: /ihm-demo
26 vitam_ihm_demo_static_content: webapp
27
28 vitam_ingestexternal_host: "ingest-external.service.{{consul_domain}}"
29 vitam_ingestexternal_port: 8001
30 vitam_ingestexternal_port_https: 8443
31 vitam_ingestexternal_baseurl: "http://{{vitam_ingestexternal_host}}:{{vitam_
    ↪ingestexternal_port}}"
32
```

```
33
34 # Internal components communication configuration
35 vitam_logbook_host: "logbook.service.{{consul_domain}}"
36 vitam_logbook_port: 9002
37 vitam_logbook_baseurl: "http://{{vitam_logbook_host}}:{{vitam_logbook_port}}"
38
39 vitam_workspace_host: "workspace.service.{{consul_domain}}"
40 vitam_workspace_port: 8201
41 vitam_workspace_baseurl: "http://{{vitam_workspace_host}}:{{vitam_workspace_port}}"
42
43 vitam_processing_host: "processing.service.{{consul_domain}}"
44 vitam_processing_port: 8203
45 vitam_processing_baseurl: "http://{{vitam_processing_host}}:{{vitam_processing_port}}"
46
47 vitam_worker_port: 9104
48
49 vitam_storageengine_host: "storage.service.{{consul_domain}}"
50 vitam_storageengine_port: 9102
51 vitam_storageengine_baseurl: "http://{{vitam_storageengine_host}}:{{vitam_
   ↪storageengine_port}}"
52
53 vitam_storageofferdefault_host: "storage-offer-default.service.{{consul_domain}}"
54 vitam_storageofferdefault_port: 9900
55 vitam_storageofferdefault_baseurl: "http://{{vitam_storageofferdefault_host}}:{{vitam_
   ↪storageofferdefault_port}}"
56
57 vitam_functional_administration_host: "functional-administration.service.{{consul_
   ↪domain}}"
58 vitam_functional_administration_port: 8004
59 vitam_functional_administration_baseurl: "http://{{vitam_functional_administration_
   ↪host}}:{{vitam_functional_administration_port}}"
60
61 # Normally no need for the host ? Maybe use the same strategy as data ?
62 elasticsearch_log_host: "{{groups['hosts-elasticsearch-log'][0]}}"
63 elasticsearch_log_http_port: "9201"
64 elasticsearch_log_tcp_port: "9301"
65
66 elasticsearch_data_http_port: "9200"
67 elasticsearch_data_tcp_port: "9300"
68
69 mongo_base_path: "{{vitam_folder_root}}"
70 mongos_port: 27017
71 mongoc_port: 27018
72 mongod_port: 27019
73 mongo_user: "vitamdb"
74
75 vitam_mongodb_host: "{{ groups['mongos'][0] }}"
76 vitam_mongodb_port: "{{mongos_port}}"
77
78 vitam_logstash_host: "logstash.service.{{consul_domain}}"
79 vitam_logstash_port: 10514
80
81 # Normally no need for the host ?
82 vitam_kibana_host: "{{groups['hosts-log-server'][0]}}"
83 vitam_kibana_port: 5601
84
85 vitam_curator_host: "{{groups['hosts-log-server'][0]}}"
86
```

```
vitam_library_port: 8090

vitam_siegfried_port: 19000

vitam_user: vitam
vitam_group: vitam

consul_domain=consul

vitam_folder_permission=0750

vitam_conf_permission=0640
```

Le `vault.yml` est également présent sous `environments-rpm /group_vars/all/all` et contient les secrets ; ce fichier est encrypté par `ansible-vault` et doit être paramétré avant le lancement de l'orchestration de déploiement.

```
KeyStorePassword_ingest_external: <mot de passe du keystore de ingest-external>
KeyManagerPassword_ingest_external: <mot de passe manager du keystore de ingest-
↪external>
TrustStorePassword_ingest_external: <mot de passe du truststore de ingest-external>
grantedKeyStorePassphrase_ingest_external: <mot de passe du grantedStore de ingest-
↪external>
p12_ihm_demo_password: <mot de passe du certificat IHM-demo>
KeyStorePassword_access_external: <mot de passe du keystore de access-external>
KeyManagerPassword_access_external: <mot de passe manager du keystore de access-
↪external>
TrustStorePassword_access_external: <mot de passe du truststore de access-external>
grantedKeyStorePassphrase_access_external: <mot de passe du grantedStore de access-
↪external>
plateforme_secret: <secret de plate-forme>

```

Le déploiement s'effectue depuis la machine "ansible" et va distribuer la solution VITAM selon l'inventaire correctement renseigné.

**Avertissement :** 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.

### Paramétrage de l'antivirus (ingest-externe)

**À faire**

A rédiger plus correctement. L'idée est de créer un autre shell sous `ansible-vitam-rpm/roles/vitam/templates/ingest-` prendre comme modèle le ficheir `scan-clamav.sh.j2`. Il faudra aussi modifier le fichier `ansible-vitam-rpm/roles/vitam/templates/ingest-external/ingest-external.conf.j2` en pointant sur le nouveau fichier.

#### Paramétrage des certificats (*-externe)

Se reporter à l'étape "PKI" du déploiement, décrite plus bas.

#### Test de la configuration

Pour tester le déploiement de VITAM, il faut se placer dans le répertoire `deployment` et entrer la commande suivante :

```
ansible-playbook ansible-vitam-rpm /vitam.yml -i environments-rpm /<ficher d'inventaire> --check
```

**Note :** cette commande n'est pas recommandée, du fait de limitations de check.

#### Déploiement

#### PKI

1. paramétrer le fichier `environnements-rpm/group_vars/all/vault.yml``et le fichier d'inventaire de la plate-forme sous ``environnements-rpm` (se baser sur le fichier hosts.example)
2. Lancer le script `pki-generate-ca.sh` : en cas d'absence de PKI, il permet de générer une PKI, ainsi que des certificats pour les échanges https entre composants. Se reporter au chapitre PKI si le client préfère utiliser sa propre PKI.
3. Lancer la script `generate_certs.sh <environnement>` . Basé sur le contenu du fichier `vault.yml`, ce script génère des certificats nécessaires au bon fonctionnement de VITAM.

3. Lancer la script `generate_stores.sh <environnement>` . Basé sur le contenu du fichier `vault.yml`, ce script génère des stores nécessaires au bon fonctionnement de VITAM.
4. Lancer le script `copie_fichiers_vitam.sh <environnement>` pour recopier dans les bons répertoires d'ansiblerie les certificats et stores précédemment créés.

#### Déploiement

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

ansible-playbook `ansible-vitam-rpm`/vitam.yml -i `environments-rpm`/<ficher d'inventaire> –ask-vault-pass

Il sera alors demandé le mot de passe correspondant au fichier `vault.yml`.

### 7.2.4 Mise à niveau

Cette section décrit globalement le processus de mise à niveau d'une solution VITAM déjà en place et ne peut se substituer aux recommandations effectuées dans la "release note" associée à la fourniture des composants mis à niveau.

La mise à jour peut actuellement être effectuée comme une "première installation".

**À faire**

faire également référence à la release note, si procédure supplémentaire particulière

CHAPITRE 8

# Validation de la procédure

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

## 8.1 Validation manuelle

Chaque service VITAM (en dehors de bases de données) expose des URL de statut présente à l'adresse suivante : `<protocole web https ou https>://<host>:<port>/<composant>/v1/status` Cette URL doit retourner une réponse HTTP 200 (sans body) sur une requête HTTP GET.

`<protocole web https ou https>://<host>:<port>/admin/v1/status` => renvoie un statut HTTP 204 si OK

## 8.2 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 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.

## 8.3 Validation via SoapUI

**À faire**

penser à ajouter la partie liée à SoapUI. Définition du formalisme.

## 8.4 Validation via IHM technique

**À faire**

pour le moment, cette IHM n'existe pas. Penser aux copies écran quand...

## 8.5 Post-installation : administration fonctionnelle

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

- le référentiel PRONOM ( lien vers pronom [6] ) 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. http://www.nationalarchives.gov.uk/aboutapps/pronom/droid-signature-files.htm

CHAPITRE 9

# Contacts et support

## 9.1 Contacts

**À faire**

La procédure pour contacter l'équipe support VITAM n'est pas définie à ce jour.

CHAPITRE 10

# Annexes

Table des figures

Liste des tableaux