



VITAM - Documentation d'exploitation

Version 2.6.1

VITAM

juin 04, 2019

Table des matières

1	Introduction	1
1.1	But de cette documentation	1
1.2	Destinataires 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	2
2.3	Glossaire	2
3	Expertises requises	5
4	Architecture de la solution logicielle VITAM	6
5	Exploitation globale	8
5.1	Gestion des accès	8
5.1.1	API	8
5.1.2	IHM de démonstration	8
5.1.3	IHM d'administration	8
5.2	Portails d'administration	9
5.2.1	Technique	9
5.2.2	Fonctionnel	9
5.3	Paramétrage & configuration	9
5.3.1	Mise à niveau de la configuration de l'environnement	9
5.3.1.1	Mise à jour du nombre de tenants	9
5.3.1.2	Mise à jour des paramètres JVM	9
5.4	Déploiement / mises à jour	9
5.4.1	Mise à jour des certificats	9
5.4.2	Mise à jour de la solution logicielle VITAM	10
5.4.3	Ajouter un/des instances de composants VITAM	10
5.4.4	Modifier la fréquence de lancement de certains <i>timers</i> systemd	11
5.5	Interruption / maintenance	11
5.5.1	Procédure d'arrêt complet	11
5.5.2	Procédure de démarrage complet	11
5.5.3	Procédure de statut	12
5.5.4	Autres cas	12

5.5.4.1	Procédure de maintenance / indisponibilité de VITAM	12
5.5.4.2	Procédure de maintenance liée aux <i>timers systemD</i>	12
5.5.4.3	Procédure de maintenance sur les composants d'administration	12
5.5.4.4	Procédure de maintenance des <i>IHM</i>	12
5.5.4.5	Procédure de maintenance des <i>Bases de données métier</i>	13
5.6	Sauvegarde / restauration	13
5.6.1	Sauvegarde	13
5.6.1.1	mongodb	13
5.6.1.2	Elasticsearch	14
5.6.2	Restauration	14
5.6.2.1	mongodb	14
5.6.2.2	Elasticsearch	14
5.7	Batches et traitements	15
5.7.1	Curator	15
5.7.2	<i>Timers systemD</i>	15
5.7.2.1	Sécurisation des journaux d'opérations	15
5.7.2.2	Sécurisation des cycles de vie	15
5.7.2.3	Sécurisation des offres de stockages	15
5.7.2.4	Autres <i>timers</i>	16
5.8	Sauvegarde des données graphe (Log shipping)	16
5.8.1	Déclenchement de la sauvegarde	16
5.8.2	Reconstruction des données <i>graphe</i>	16
5.9	Recalcul des données graphe	17
5.9.1	Déclenchement	17
5.10	Montée de version du fichier de signature de Siegfried	18
5.11	Griffins	18
5.11.1	Ajout de nouveaux / mise à jour de <i>griffins</i>	19
5.11.1.1	Ajout de <i>griffins</i>	19
5.11.1.2	Mise à jour des <i>griffins</i>	19
5.11.1.3	Préparation du système	19
5.11.1.4	Prise en compte technique par VITAM	19
5.12	Reconstruction	19
5.12.1	Procédure multi-sites	20
5.12.1.1	Cas du site primaire	20
5.12.1.2	Cas du site secondaire	20
5.12.2	Procédure mono-site	21
5.13	Plan de Reprise d'Activité (PRA)	21
5.13.1	Déclenchement	21
5.13.2	Retour en situation nominale	22
5.13.2.1	Déclenchement	22
5.14	Resynchronisation d'une offre	23
5.14.1	Cas de l'ajout d'une nouvelle offre	23
5.14.2	Cas de la resynchronisation d'une offre temporairement indisponible	25
5.15	Procédure d'exploitation suite à la création ou la modification d'une ontologie	26
5.15.1	Création d'une ontologie	26
5.15.2	Changement de type d'une ontologie existante	27
5.16	Procédure d'exploitation pour la mise en pause forcée d'une opération	28
5.16.1	Mise en pause forcée	28
5.16.2	Sortie de la mise en pause forcée	28
5.17	Réindexation	29
5.17.1	Déclenchement	29
5.18	Procédure d'exploitation pour la révocation des certificats SIA et Personae	30
5.19	Activation/désactivation d'une offre	31
5.20	Nettoyage d'un environnement	32

5.20.1	Etat des lieux après purge	33
5.20.2	Limitations	33
6	Suivi de l'état du système	34
6.1	Veille et patchs sécurité	34
6.2	Métriques	34
6.2.1	Configuration	34
6.2.1.1	Activation/désactivation	34
6.2.1.2	Registres	34
6.2.1.3	Reporters	35
6.2.1.4	Fichier de configuration	35
6.2.2	Métier	35
6.2.3	Métriques techniques	35
6.2.3.1	Métriques système critiques	35
6.2.3.2	Indicateurs de SLA	35
6.2.3.3	Indicateurs de performance	35
6.2.4	Visualisation	36
6.2.4.1	Discover	36
6.2.4.2	Visualize	36
6.2.4.3	Dashboards	37
6.3	API de de supervision	38
6.3.1	Patte d'administration	38
6.3.1.1	/admin/v1/status	38
6.3.1.2	/admin/v1/version	39
6.3.1.3	/admin/v1/autotest	40
6.3.2	Patte de service	41
6.4	Logs	41
6.4.1	Changement des règles de log	41
6.4.2	Rétention des index sous elasticsearch-log	42
6.5	Audit	43
6.6	Gestion de la capacité	43
6.7	Suivi de l'état de sécurité	43
6.8	Alerting	43
6.8.1	Système	43
6.8.2	Applicatif	43
6.9	Suivi des Workflows	43
6.9.1	Suivi	44
6.9.1.1	IHM	44
6.9.1.2	Appels REST	44
6.9.2	Cas des <i>workflows</i> en FATAL	44
6.9.2.1	Plugins et Handlers	44
6.9.2.2	Distributeur	45
6.9.2.3	Processing - State Machine	45
6.9.3	Redémarrer un processus en cas de pause	46
6.9.3.1	Trouver la cause	46
6.9.3.2	Relancer le Workflow	46
6.9.3.2.1	Vérifier les inputs	46
6.9.3.2.2	Rejouer une étape	46
6.9.3.2.3	Prochaine étape	46
6.9.3.2.4	Finaliser le workflow	46
6.10	Cohérence des journaux	47
6.10.1	Lancement	47
6.10.2	Résultat	47
6.11	Liste des <i>timers</i> systemd	47

6.11.1	Timers de maintenance des index elasticsearch-log	47
6.11.1.1	vitam-curator-metrics-indexes	47
6.11.1.2	vitam-curator-close-old-indexes	48
6.11.1.3	vitam-curator-delete-old-indexes	48
6.11.2	Timers de gestion des journaux (preuve systématique)	48
6.11.2.1	vitam-storage-log-backup	48
6.11.2.2	vitam-storage-accesslog-backup	49
6.11.2.3	vitam-storage-log-traceability	49
6.11.2.4	vitam-traceability-operations	49
6.11.2.5	vitam-traceability-lfc-unit	49
6.11.2.6	vitam-traceability-lfc-objectgroup	50
6.11.3	Timers d'audit interne VITAM	50
6.11.3.1	vitam-traceability-audit	50
6.11.3.2	vitam-rule-management-audit	50
6.11.4	Timer relatif aux liens symboliques de <i>accession register</i>	50
6.11.4.1	vitam-create-accession-register-symbolic	50
6.11.5	Timers de reconstruction VITAM	51
6.11.5.1	vitam-functional-administration-reconstruction	51
6.11.5.2	vitam-logbook-reconstruction	51
6.11.5.3	vitam-metadata-reconstruction	51
6.11.5.4	vitam-metadata-store-graph	52
7	Exploitation des COTS de la solution logicielle VITAM	53
7.1	Généralités	53
7.2	COTS	53
7.2.1	Cerebro	53
7.2.1.1	Présentation	53
7.2.1.2	Configuration / fichiers utiles	53
7.2.1.2.1	Fichier /vitam/conf/cerebro/application.conf	53
7.2.1.3	Opérations	55
7.2.2	consul	55
7.2.2.1	Présentation	55
7.2.2.1.1	Cas serveur	56
7.2.2.1.2	Cas agent	56
7.2.2.2	Configuration / fichiers utiles	56
7.2.2.2.1	Cas des applicatifs monitorés par Consul	56
7.2.2.2.1.1	Fichier /vitam/conf/consul/service-<composant>.json	56
7.2.2.3	Opérations	57
7.2.3	Kibana interceptor	58
7.2.3.1	Présentation	58
7.2.3.2	Configuration / fichiers utiles	58
7.2.3.2.1	Fichier elastic-kibana-interceptor.conf	58
7.2.3.3	Opérations	59
7.2.4	elasticsearch chaîne de log	59
7.2.4.1	Présentation	59
7.2.4.2	Configuration / fichiers utiles	59
7.2.4.2.1	Fichier /vitam/conf/elasticsearch-log/log4j2.properties	59
7.2.4.2.2	Fichier /vitam/conf/elasticsearch-log/jvm.options	61
7.2.4.2.3	Fichier /vitam/conf/elasticsearch-log/elasticsearch.yml	64
7.2.4.2.4	Fichier /vitam/conf/elasticsearch-log/sysconfig/elasticsearch	66

7.2.4.2.5	Fichier /usr/lib/tmpfiles.d/elasticsearch-log.conf . . .	68
7.2.4.3	Opérations	68
7.2.5	elasticsearch Vitam	68
7.2.5.1	Présentation	68
7.2.5.2	Configuration / fichiers utiles	69
7.2.5.2.1	Fichier log4j2.properties	69
7.2.5.2.2	Fichier jvm.options	71
7.2.5.2.3	Fichier elasticsearch.yml	73
7.2.5.2.4	Fichier sysconfig/elasticsearch	76
7.2.5.2.5	Fichier /usr/lib/tmpfiles.d/elasticsearch-data.conf . . .	77
7.2.5.3	Opérations	78
7.2.6	Kibana	78
7.2.6.1	Présentation	78
7.2.6.2	Configuration / fichiers utiles	78
7.2.6.3	Opérations	79
7.2.7	log server	79
7.2.7.1	Présentation	79
7.2.7.2	Configuration / fichiers utiles	79
7.2.7.3	Opérations	79
7.2.8	mongoC	80
7.2.8.1	Présentation	80
7.2.8.2	Configuration / fichiers utiles	80
7.2.8.2.1	Fichier mongoc.conf	80
7.2.8.2.2	Fichier keyfile	81
7.2.8.3	Opérations	81
7.2.9	mongoD	82
7.2.9.1	Présentation	82
7.2.9.2	Configuration / fichiers utiles	82
7.2.9.2.1	Fichier mongod.conf	82
7.2.9.2.2	Fichier keyfile	83
7.2.9.3	Opérations	83
7.2.10	mongoS	83
7.2.10.1	Présentation	83
7.2.10.2	Configuration / fichiers utiles	83
7.2.10.2.1	Fichier mongos.conf	83
7.2.10.2.2	Fichier keyfile	84
7.2.10.3	Opérations	84
7.2.11	siegfried	85
7.2.11.1	Présentation	85
7.2.11.2	Configuration / fichiers utiles	85
7.2.11.3	Opérations	85
8	Exploitation des composants de la solution logicielle VITAM	86
8.1	Généralités	86
8.2	Composants	86
8.2.1	Fichiers communs	86
8.2.1.1	Fichier /vitam/conf/<composant>/sysconfig/java_opts	86
8.2.1.2	Fichier /vitam/conf/<composant>/logback.xml	87
8.2.1.3	Fichier /vitam/conf/<composant>/logback-access.xml	88
8.2.1.4	Fichier /vitam/conf/<composant>/jetty-config.xml	91
8.2.1.5	Fichier /vitam/conf/<composant>/logbook-client.conf	97
8.2.1.6	Fichier /vitam/conf/<composant>/server-identity.conf	97
8.2.1.7	Fichier /vitam/conf/<composant>/antisamy-esapi.xml	97
8.2.1.8	Fichier /vitam/conf/<composant>/vitam.conf	111

8.2.1.9	Fichier /vitam/conf/<composant>/vitam.metrics.conf	111
8.2.1.10	Fichier /vitam/conf/<composant>/java.security	112
8.2.2	Access	113
8.2.2.1	access external	113
8.2.2.1.1	Présentation	113
8.2.2.1.2	Configuration / fichiers utiles	113
8.2.2.1.2.1	Fichier access-external.conf	113
8.2.2.1.2.2	Fichier access-internal-client.conf	113
8.2.2.1.2.3	Fichier functional-administration-client.conf	113
8.2.2.1.2.4	Fichier ingest-internal-client.conf	113
8.2.2.1.2.5	Fichier internal-security-client.conf	113
8.2.2.1.3	Opérations	114
8.2.2.2	access-internal	114
8.2.2.2.1	Présentation du composant	114
8.2.2.2.2	Configuration / fichiers utiles	114
8.2.2.2.2.1	Fichier access-internal.conf	115
8.2.2.2.2.2	Fichier storage-client.conf	115
8.2.2.2.2.3	Fichier metadata-client.conf	115
8.2.2.2.2.4	Fichier functional-administration-client.conf	115
8.2.2.2.3	Opérations	115
8.2.3	Batch-Report	116
8.2.3.1	Présentation	116
8.2.3.2	Configuration	116
8.2.3.2.1	Fichier batch-report.conf	116
8.2.3.3	Client batch-report	116
8.2.3.4	Opérations	116
8.2.4	common-plugin	117
8.2.4.1	Présentation du composant	117
8.2.4.2	Classes utiles	117
8.2.4.2.1	Classe Item Status	117
8.2.4.2.2	Classe VitamAutoCloseable	117
8.2.4.2.3	Classe ParameterHelper	117
8.2.4.2.4	Classe VitamParameter	118
8.2.4.2.5	Classe ProcessingException	118
8.2.4.2.6	Classe IOParameter	118
8.2.4.2.7	Classe ProcessingUri	118
8.2.4.2.8	Classe UriPrefix	118
8.2.4.2.9	Classe AbstractWorkerParameters	118
8.2.4.2.10	Classe DefaultWorkerParameters	118
8.2.4.2.11	Classe WorkerParameterName	118
8.2.4.2.12	Classe WorkerParameters	118
8.2.4.2.13	Classe WorkerParametersDeserializer	118
8.2.4.2.14	Classe WorkerParametersFactory	119
8.2.4.2.15	Classe WorkerParametersSerializer	119
8.2.4.2.16	Interface HandlerIO	119
8.2.4.2.17	Classe WorkerAction	119
8.2.4.2.18	Classe HandlerIOImpl	119
8.2.5	Common	119
8.2.5.1	Présentation	119
8.2.5.2	Format Identifiers	119
8.2.5.2.1	Configuration des services d'identification des formats	120
8.2.6	Functional administration	120
8.2.6.1	Présentation	120
8.2.6.2	Configuration / fichiers utiles	120

8.2.6.2.1	Fichier <code>functional-administration.conf</code>	121
8.2.6.2.2	Passage des identifiants des référentiels en mode esclave	121
8.2.6.2.3	Configuration du Functional administration	123
8.2.6.3	Opérations	123
8.2.7	Hello World Plugin	123
8.2.7.1	Présentation	123
8.2.7.1.1	Comment intégrer votre plugins dans vitam ?	124
8.2.7.1.2	Créer un nouveau workflow	124
8.2.7.1.3	Comment ajouter un nouveau workflow dans vitam ?	125
8.2.7.1.4	Comment ajouter la traduction de clés des Plugins ?	125
8.2.7.1.5	Comment appeler le nouveau workflow ?	126
8.2.7.1.6	Remarques	126
8.2.7.1.7	Securité	126
8.2.8	ihm-demo	126
8.2.8.1	Présentation	126
8.2.8.2	Configuration / fichiers utiles	126
8.2.8.2.1	Fichier <code>access-external-client.conf</code>	127
8.2.8.2.2	Fichier <code>ihm-demo.conf</code>	127
8.2.8.2.3	Fichier <code>ingest-external-client.conf</code>	127
8.2.8.2.4	Fichier <code>shiro.ini</code>	128
8.2.8.3	Configuration de apache shiro	130
8.2.8.4	Présentation authentification via LDAP et via certificat	130
8.2.8.5	Décryptage de <code>shiro.ini</code>	131
8.2.8.6	Opérations	131
8.2.9	ihm-recette	132
8.2.9.1	Présentation	132
8.2.9.2	Configuration / fichiers utiles	132
8.2.9.2.1	Fichier <code>access-external-client.conf</code>	132
8.2.9.2.2	Fichier <code>driver-location.conf</code>	133
8.2.9.2.3	Fichier <code>driver-mapping.conf</code>	133
8.2.9.2.4	Fichier <code>functional-administration-client.conf</code>	133
8.2.9.2.5	Fichier <code>ihm-recette-client.conf</code>	133
8.2.9.2.6	Fichier <code>ihm-recette.conf</code>	133
8.2.9.2.7	Fichier <code>ingest-external-client.conf</code>	134
8.2.9.2.8	Fichier <code>shiro.ini</code>	134
8.2.9.2.9	Fichier <code>static-offer.json</code>	136
8.2.9.2.10	Fichier <code>static-strategy.json</code>	136
8.2.9.2.11	Fichier <code>storage-client.conf</code>	137
8.2.9.2.12	Fichier <code>storage.conf</code>	137
8.2.9.2.13	Fichier <code>storage-offer.conf</code>	137
8.2.9.2.14	Fichier <code>tnr.conf</code>	137
8.2.9.3	Opérations	137
8.2.10	Ingest	138
8.2.10.1	Introduction	138
8.2.10.2	ingest-external	138
8.2.10.2.1	Présentation	138
8.2.10.2.2	Configuration / fichiers utiles	138
8.2.10.2.2.1	Fichier <code>ingest-external.conf</code>	139
8.2.10.2.2.2	Fichier <code>ingest-internal-client.conf</code>	139
8.2.10.2.2.3	Fichier <code>internal-security-client.conf</code>	139
8.2.10.2.2.4	Fichier <code>format-identifiers.conf</code>	139
8.2.10.2.2.5	Fichier <code>functional-administration-client.conf</code>	140
8.2.10.2.2.6	Fichier <code>scan-clamav.sh</code>	140
8.2.10.2.3	Opérations	141

8.2.10.3	ingest-internal	142
8.2.10.3.1	Présentation	142
8.2.10.3.2	Configuration / fichiers utiles	142
8.2.10.3.2.1	Fichier ingest-internal.conf	142
8.2.10.3.2.2	Fichier storage-client.conf	142
8.2.10.3.3	Opérations	142
8.2.11	Security-Internal	143
8.2.11.1	Introduction	143
8.2.11.2	security-internal-exploitation	143
8.2.11.2.1	Fichier security-internal.conf	143
8.2.11.2.2	Fichier personal-certificate-permissions.conf	144
8.2.11.3	Opérations	146
8.2.12	Logbook	147
8.2.12.1	Présentation	147
8.2.12.2	Logbook Exploitation	147
8.2.12.2.1	Configuration du Logbook	147
8.2.12.2.2	Fichier logbook.conf	147
8.2.12.2.3	Fichier functional-administration-client.conf	149
8.2.12.2.4	Fichier logbook-client.conf	149
8.2.12.2.5	Fichier securisationDaemon.conf	149
8.2.12.2.6	Fichier storage-client.conf	149
8.2.12.2.7	Fichier traceabilityAudit.conf	149
8.2.12.3	Opérations	149
8.2.13	Metadata	150
8.2.13.1	Présentation	150
8.2.13.2	Configuration / fichiers utiles	150
8.2.13.2.1	Fichier metadata.conf	150
8.2.13.2.2	Fichier functional-administration-client.conf	151
8.2.13.2.3	Fichier storage-client.conf	151
8.2.13.3	Opérations	151
8.2.14	Processing	151
8.2.14.1	Introduction	151
8.2.14.1.1	But de cette documentation	151
8.2.14.2	Processing	151
8.2.14.2.1	Configuration du worker	152
8.2.14.2.2	Supervision du service	152
8.2.14.3	Configuration / fichiers utiles	152
8.2.14.3.1	Fichier processing.conf	152
8.2.14.3.2	Fichier version.conf	152
8.2.14.3.3	Fichier storage-client.conf	153
8.2.14.3.4	Fichier metadata-client.conf	153
8.2.14.4	Opérations	153
8.2.15	Storage	153
8.2.15.1	Introduction	153
8.2.15.1.1	But de cette documentation	153
8.2.15.2	storage-engine	154
8.2.15.2.1	Présentation	154
8.2.15.2.2	Storage Engine	154
8.2.15.2.2.1	Configuration du moteur de stockage	154
8.2.15.2.2.2	Configuration du driver de l'offre de stockage par défaut	156
8.2.15.2.2.3	Supervision du service	156
8.2.15.2.3	Configuration / fichiers utiles	156
8.2.15.2.3.1	Fichier driver-location.conf	156
8.2.15.2.3.2	Fichier driver-mapping.conf	156

8.2.15.2.3.3	Fichier <code>static-offer.json</code>	157
8.2.15.2.3.4	Fichier <code>static-strategy.json</code>	157
8.2.15.2.3.5	Fichier <code>storage-engine.conf</code>	157
8.2.15.2.4	Opérations	158
8.2.15.2.4.1	<code>access-log</code>	159
8.2.15.3	<code>offer</code>	159
8.2.15.3.1	Présentation	159
8.2.15.3.2	Storage Offer Default	160
8.2.15.3.2.1	Configuration de l’offre de stockage	160
8.2.15.3.2.2	Supervision du service	160
8.2.15.3.3	Configuration / fichiers utiles	160
8.2.15.3.3.1	Fichier <code>default-offer.conf</code>	161
8.2.15.3.3.2	Fichier <code>default-storage.conf</code>	161
8.2.15.3.4	Opérations	163
8.2.16	Technical administration	163
8.2.16.1	Présentation	163
8.2.17	Worker	163
8.2.17.1	Introduction	163
8.2.17.2	Configuration / fichiers utiles	163
8.2.17.2.1	Fichier <code>batch-report-client.conf</code>	164
8.2.17.2.2	Fichier <code>format-identifiers.conf</code>	164
8.2.17.2.3	Fichier <code>functional-administration-client.conf.j2</code>	164
8.2.17.2.4	Fichier <code>metadata-client.conf</code>	164
8.2.17.2.5	Fichier <code>storage-client.conf</code>	164
8.2.17.2.6	Fichier <code>verify-timestamp.conf</code>	164
8.2.17.2.7	Fichier <code>version.conf</code>	165
8.2.17.2.8	Fichier <code>worker.conf</code>	165
8.2.17.3	Opérations	165
8.2.18	Workspace	166
8.2.18.1	Présentation	166
8.2.18.2	Configuration / fichiers utiles	166
8.2.18.2.1	Fichier <code>workspace.conf</code>	166
8.2.18.3	Opérations	167
9	Intégration d’une application externe dans Vitam	168
9.1	Prérequis	168
9.2	Intégration de certificats clients de VITAM	168
9.2.1	Authentification applicative SIA	168
9.2.2	Authentification <i>personae</i>	169
9.2.2.1	Ajout d’un certificat pour l’authentification <i>Personae</i>	169
9.2.2.2	Suppression d’un certificat pour l’authentification <i>Personae</i>	169
9.3	Déploiement des keystores	169
9.3.1	Vitam n’est pas encore déployé	169
9.3.2	Vitam est déjà déployé	169
10	Aide à l’exploitation	170
10.1	Analyse de premier niveau	170
10.1.1	Etat par Consul	170
10.1.2	Etat par Kibana	171
10.2	Playbook ansible pour échanger avec le support	171
11	Questions Fréquemment Posées	173
11.1	Présentation	173
11.2	Retour d’expérience / cas rencontrés	173

11.2.1	Crash rsyslog, code killed, signal : BUS	173
11.2.2	Mongo-express ne se connecte pas à la base de données associée	173
11.2.3	Elasticsearch possède des shard non alloués (état « UNASSIGNED »)	173
11.2.4	Elasticsearch possède des shards non initialisés (état « INITIALIZING »)	174
11.2.5	MongoDB semble lent	174
11.2.6	Les shards de MongoDB semblent mal équilibrés	175
11.2.7	L'importation initiale (profil de sécurité, certificats) retourne une erreur	176
11.2.8	Problème d'ingest et/ou d'access	176
11.2.9	Erreur d'inconsistance des données MongoDB / ES	176
12	Annexes	177
12.1	Cycle de vie des certificats	177
12.2	Gestion des anomalies en production	179
12.2.1	Numérotation des versions	179
12.2.2	Mise à disposition du logiciel	179
12.2.3	Gestion des patches	179
	Index	183

1.1 But de cette documentation

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

2.1 Information concernant les licences

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

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

2.2.2 Référentiels externes

2.3 Glossaire

API Application Programming Interface

AU *Archive Unit*, unité archivistique

http://www.cecill.info/licences/Licence_CeCILL_V2.1-fr.html

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

BDD Base De Données

CA *Certificate Authority*, autorité de certification

CAS Content Addressable 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.

DAT Dossier d'Architecture Technique

DC Data Center

DEX Dossier d'EXploitation

DIN Dossier d'INstallation

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*³

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

ELK *Elasticsearch Logstash Kibana*

IHM Interface Homme Machine

IP *Internet Protocol*

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*

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*⁴

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

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

NoSQL Base de données non-basée sur un paradigme classique des bases relationnelles. [Définition NoSQL](#)⁵

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.

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](#)⁶

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](#)⁷

RGI Référentiel Général d'Interopérabilité

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

SAE Système d'Archivage Électronique

SEDA Standard d'Échange de Données pour l'Archivage

SGBD Système de Gestion de Base de Données

SIA Système d'Informations Archivistique

SIEM Security Information and Event Management

SIP Submission Information Package

SSH *Secure SHell*

Swift OpenStack Object Store project

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

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

<https://fr.wikipedia.org/wiki/NoSQL>
https://fr.wikipedia.org/wiki/Infrastructure_%C3%A0_cl%C3%A9s_publicques
https://fr.wikipedia.org/wiki/Representational_state_transfer

CHAPITRE 3

Expertises requises

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

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

CHAPITRE 4

Architecture de la solution logicielle VITAM

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

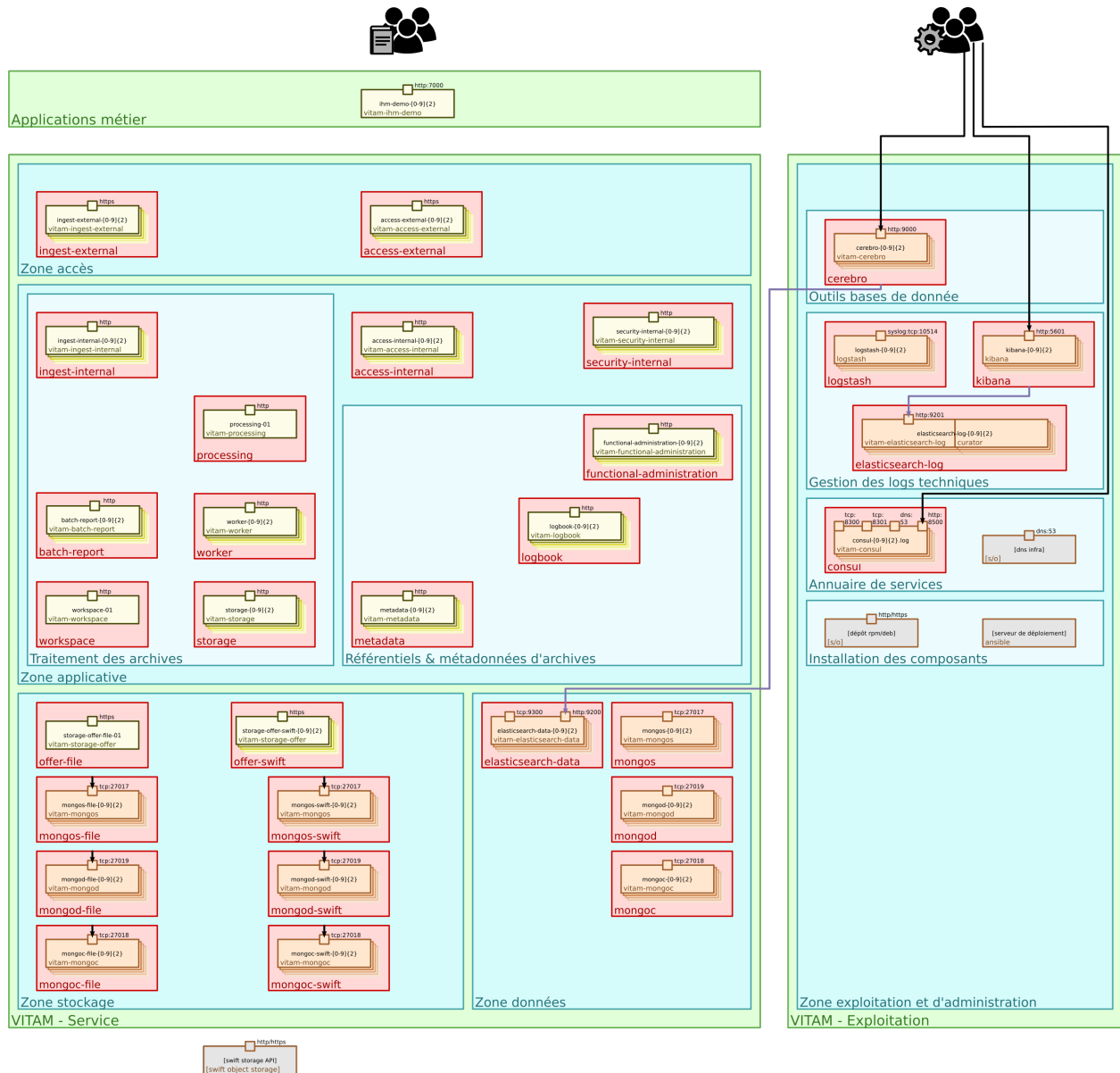


Fig. 1 – Vue d'ensemble d'un déploiement *VITAM* : zones, composants

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.

5.1 Gestion des accès

5.1.1 API

La gestion des accès à aux *API* externes se fait via les *granted stores* (cf. *Fichiers communs* (page 86) pour les fichiers communs, cf. *Configuration / fichiers utiles* (page 113) pour access-external et *Configuration / fichiers utiles* (page 142) pour ingest-external).

5.1.2 IHM de démonstration

Dans cette version, la gestion des utilisateurs se fait par configuration d'un fichier plat (cf. *Configuration de apache shiro* (page 130)).

Note : Il est également possible de connecter cette *IHM* à un annuaire LDAP ; se reporter au *DIN* pour plus d'informations.

5.1.3 IHM d'administration

Dans cette version, la gestion des utilisateurs se fait par configuration d'un fichier plat (cf. *exploitihmrecette*).

Note : Il est également possible de connecter cette *IHM* à un annuaire LDAP ; se reporter au *DIN* pour plus d'informations.

5.2 Portails d'administration

5.2.1 Technique

Aucun portail d'administration technique n'est prévu dans cette version de la solution logicielle *VITAM*.

5.2.2 Fonctionnel

Le portail d'administration fonctionnel est intégré au composant **ihm-demo** dans cette version de la solution logicielle *VITAM* (cf. *Présentation* (page 126)).

5.3 Paramétrage & configuration

L'étape de paramétrage et la configuration sont essentiellement liées à la mise en place ou la mise à niveau de la solution logicielle *VITAM* (ansible / inventaire).

Voir aussi :

Plus d'informations, et notamment les paramètres d'installation, sont disponibles dans le *DIN*.

5.3.1 Mise à niveau de la configuration de l'environnement

5.3.1.1 Mise à jour du nombre de tenants

Note : se référer au *DIN*

5.3.1.2 Mise à jour des paramètres JVM

Note : se référer au *DIN*

5.4 Déploiement / mises à jour

5.4.1 Mise à jour des certificats

Pour mettre à jour les certificats (avant expiration par exemple), il suffit de les mettre à jour dans les répertoires de déploiement, puis de régénérer les stores (dans `environments/keystores`) et lancer leur redéploiement via cette commande ansible :

Cas où le fichier de mot de passe pour vault n'existe pas

```
# Si le mot de passe du vault n'est pas renseigné dans le fichier vault_pass.txt
ansible-playbook ansible-vitam/vitam.yml -i environments/<fichier d'inventaire> --ask-
↪ vault-pass --tags update_vitam_certificates
ansible-playbook ansible-vitam-extra/extra.yml -i environments/<fichier d'inventaire>
↪ --ask-vault-pass --tags update_vitam_certificates
```

Cas où le fichier de mot de passe pour vault existe

```
# Si le mot de passe du vault est renseigné dans le fichier vault_pass.txt
ansible-playbook ansible-vitam/vitam.yml -i environments/<fichier d'inventaire> --
↳ vault-password-file vault_pass.txt --tags update_vitam_certificates
ansible-playbook ansible-vitam-extra/extra.yml -i environments/<fichier d'inventaire>
↳ --vault-password-file vault_pass.txt --tags update_vitam_certificates
```

Voir aussi :

Le cycle de vie des certificats est rappelé dans les annexes. Une vue d'ensemble est également présentée dans le [DIN](#).

5.4.2 Mise à jour de la solution logicielle VITAM

Pour la mise à jour de la solution logicielle [VITAM](#) (tout comme pour sa première installation), se référer au [DIN](#), au [DMV](#), ainsi qu'à la « release note » associée à toute version.

Ces documents détaillent les pré-requis, la configuration des fichiers et les procédures éventuelles de migration de données pour effectuer une mise à jour applicative. Le [DMV](#) explique également comment valider une montée de version applicative de la solution logicielle [VITAM](#).

Voir aussi :

Plus d'informations, et notamment les paramètres d'installation, sont disponibles dans le [DIN](#).

Voir aussi :

Dans le cadre d'une montée de version, se référer également au [DMV](#).

5.4.3 Ajouter un/des instances de composants VITAM

Dans le cas où le *sizing* initial ne donne pas pleinement satisfaction, il est possible de rajouter à une solution logicielle [VITAM](#) existante une/des instances supplémentaires de composants.

Prudence : Dans le cas d'ajout d'une offre, il est nécessaire de suivre la procédure de resynchronisation des offres.

Avertissement : Seul le composant « vitam-processing » n'est pas multi-instanciable.

Avertissement : Le composant « vitam-workspace » est multi-instanciable, si son répertoire `/vitam/data/workspace` est partagé entre les différentes instances du composant. Bien penser, dans ce cas, à la problématique de droits d'accès aux fichiers.

1. Modifier l'inventaire avec la/les VM supplémentaire(s)
2. Lancer un déploiement comme indiqué dans le [DIN](#) en rajoutant la directive `-l <liste de/des VM(s) supplémentaire(s)>`

5.4.4 Modifier la fréquence de lancement de certains *timers* systemd

Par défaut, la solution logicielle *VITAM* déploie et active, selon l'usage (site primaire / site secondaire), des *timers* systemd. Le *playbook* ansible d'installation de vitam ansible-vitam/vitam.yml, permet d'uniquement modifier la fréquence des *timers* en rajoutant le tag `update_timers_frequency`.

Pour cela, il faut éditer la section `vitam_timers` dans le fichier `environments/group_vars/all/vitam_vars.yml`.

A l'issue, lancer le *playbook* avec la commande

```
ansible-playbook -i <inventaire> ansible-vitam/vitam.yml --tags update_timers_
↪frequency --ask-vault-pass
```

ou bien, si vous utilisez le fichier `vault_pass.txt`

```
ansible-playbook -i <inventaire> ansible-vitam/vitam.yml --tags update_timers_
↪frequency --vault-password-file vault_pass.txt
```

5.5 Interruption / maintenance

5.5.1 Procédure d'arrêt complet

Un *playbook* ansible d'arrêt complet de la solution logicielle *VITAM* est fourni, sous `deployment/ansible-vitam-exploitation` (fichier de *playbook* `stop_vitam.yml`), pour réaliser de façon automatisée les actions nécessaires.

Avertissement : Ce script, en l'état, permet un *EMERGENCY BREAK*, autrement dit un arrêt brutal des composants, ne permettant pas de garantir, à l'issue, une cohérence des données. Il est donc fortement recommandé de positionner les traitements courants en pause avant de lancer la procédure d'arrêt.

Note : Une confirmation est demandée pour lancer ce script d'arrêt de la solution logicielle *VITAM*.

Un *playbook* ansible d'arrêt des *timers* systemd *VITAM* est également fourni, sous `deployment/ansible-vitam-exploitation` (fichier de *playbook* `stop_vitam_timers.yml`), pour réaliser de façon automatisée les actions nécessaires. Ce script est à lancer une fois l'arrêt des services correctement réalisé.

5.5.2 Procédure de démarrage complet

Le pré-requis est le bon fonctionnement des partitions hébergeant la solution logicielle *VITAM*.

Un *playbook* ansible de démarrage de la solution logicielle *VITAM* est fourni, sous `deployment/ansible-vitam-exploitation` (fichier de *playbook* `start_vitam.yml`), pour réaliser de façon automatisée les actions nécessaires.

Un *playbook* ansible de démarrage des *timers* systemd *VITAM* est également fourni, sous `deployment/ansible-vitam-exploitation` (fichier de *playbook* `start_vitam_timers.yml`), pour réaliser de façon automatisée les actions nécessaires. Ce script est à lancer une fois le démarrage des services correctement réalisé.

5.5.3 Procédure de statut

Un playbook ansible de démarrage de VITAM est fourni, sous `deployment/ansible-vitam-exploitation` (fichier de *playbook* `status_vitam.yml`), pour réaliser de façon automatisée les tests « autotest » intégrés dans la solution logicielle *VITAM*.

5.5.4 Autres cas

5.5.4.1 Procédure de maintenance / indisponibilité de VITAM

Deux *playbooks* sont fournis dans `deployment/ansible-vitam-exploitation` :

- fichier de *playbook* `stop_external.yml`, permettant d'arrêter sélectivement les composants *VITAM ingest-external* et *access-external*
- fichier de *playbook* `start_external.yml`, permettant de démarrer sélectivement *VITAM ingest-external* et *access-external*

Ces scripts permettent d'empêcher l'usage de la solution logicielle *VITAM* par les services versants, tout en laissant opérationnel le reste de la solution logicielle. Ces *playbooks* peuvent être utiles, voire nécessaires, dans le cadre d'une migration de données ou de maintenance de la solution logicielle *VITAM*.

5.5.4.2 Procédure de maintenance liée aux *timers systemd*

Deux *playbooks* sont fournis dans `deployment/ansible-vitam-exploitation` :

- fichier de *playbook* `stop_vitam_timers.yml`, permettant d'arrêter sélectivement les *timers systemd*
- fichier de *playbook* `start_vitam_timers.yml`, permettant de démarrer sélectivement les *timers systemd*

5.5.4.3 Procédure de maintenance sur les composants d'administration

Deux *playbooks* sont fournis dans `deployment/ansible-vitam-exploitation` :

- fichier de *playbook* `stop_vitam_admin.yml`, permettent d'arrêter sélectivement les composants Consul, la chaîne de log (logstash / cluster elasticsearch log / kibana-log), cerebro et les docker mongo-express et elasticsearch-head
- fichier de *playbook* `start_vitam_admin.yml`, permettent de démarrer sélectivement les composants Consul, la chaîne de log (logstash / cluster elasticsearch log / kibana-log), cerebro et les docker mongo-express et elasticsearch-head

Avertissement : En passant le *playbook* d'arrêt, l'ensemble de la solution logicielle *VITAM* devient inutilisable.

5.5.4.4 Procédure de maintenance des *IHM*

Deux *playbooks* sont fournis dans `deployment/ansible-vitam-exploitation` :

- fichier de *playbook* `stop_vitam_ihm.yml`, permettent d'arrêter sélectivement les composants *VITAM IHM ihm-demo* et *ihm-recette*
- fichier de *playbook* `start_vitam_ihm.yml`, permettent de démarrer sélectivement les composants *VITAM IHM ihm-demo* et *ihm-recette*

5.5.4.5 Procédure de maintenance des Bases de données métier

Quatre *playbooks* sont fournis dans `deployment/ansible-vitam-exploitation` :

- fichier de *playbook* `start_elasticsearch_data.yml`, permettent de démarrer la totalité des elasticsearch-data
- fichier de *playbook* `start_mongodb.yml`, permettent de démarrer les composants mongodb
- fichier de *playbook* `stop_elasticsearch_data.yml`, permettent d'arrêter la totalité des composants elasticsearch-data
- fichier de *playbook* `stop_mongodb.yml`, permettent d'arrêter les composants mongodb

5.6 Sauvegarde / restauration

Avertissement : Cette méthode s'applique uniquement pour des déploiements de petite taille, et n'est pas recommandée pour un usage en production. La gestion de la sauvegarde des bases de métadonnées MongoDB et Elasticsearch, ainsi que de la restauration de leur contenu en cohérence avec les offres sous-jacentes, est déjà gérée par les mécanismes intrinsèques à la solution *VITAM*, cf. le *DAT*.

Les procédures sont issues des documentations officielles :

- mongoDB : <https://docs.mongodb.com/manual/tutorial/backup-and-restore-tools/>
- elasticsearch : <https://www.elastic.co/guide/en/elasticsearch/reference/current/modules-snapshots.html>

5.6.1 Sauvegarde

Note : Pour que cette sauvegarde soit fonctionnellement correcte, il faut que la solution logicielle *VITAM* soit dans un état stable et cohérent, sans possibilité d'ajouter des « ingests » et sans travail de fond (jobs de sécurisation, ...).

5.6.1.1 mongoDB

La commande suivante est à lancer depuis une machine hébergeant le composant `vitam-mongod` (pour data ET pour chaque offer) de *VITAM* ayant un espace suffisant dans le répertoire de sauvegarde (défini par `--out`)

```
mongodump --host mongodb.example.net --port 27017 --out /data/backup/ --username_vitamdb-admin --password "pass"
```

Note : Se reporter aux fichiers associés à l'ansiblerie de déploiement pour le mot de passe `vitamdb-admin`

Rapatrier sur un serveur approprié le produit généré dans la valeur de `--out`.

Pour rappel, il y a un cluster mongo de « data », ainsi qu'un cluster mongo « offer » associé à chaque offre. Pour un système cohérent, il faut effectuer la sauvegarde de chacun de ces *clusters*.

Note : Il est recommandé de procéder à une sauvegarde régulière des bases **identity**, **config** et **admin**.

5.6.1.2 Elasticsearch

La commande suivante est à lancer depuis une machine elasticsearch de *VITAM* ayant un espace suffisant dans le répertoire de sauvegarde

```
curl -X PUT http://elasticsearch-data.service.${consul_domain}:9200/_snapshot/vitam_
↳backup -d '{ "type": "fs", "settings": { "location": "${output_dir}" } }'
```

Cette étape va créer le *repository vitam_backup* de sauvegarde, l'arborescence étant définie par `${output_dir}`.

Pour vérifier l'état du *repository vitam_backup* sur les noeuds du cluster

```
curl -X POST http://elasticsearch-data.service.${consul_domain}:9200/_snapshot/vitam_
↳backup/_verify
```

Pour lancer un *snapshot* (dans l'exemple, appelé `snapshot_1`)

```
curl -X PUT http://elasticsearch-data.service.${consul_domain}:9200/_snapshot/vitam_
↳backup/snapshot_1?wait_for_completion=true
```

Note : la commande ne rendra la main qu'à la fin de la procédure de *snapshot*.

A l'issue de la sauvegarde, procéder à une copie de `${output_dir}` sur un serveur à part.

5.6.2 Restauration

Note : Comme pour la sauvegarde, la restauration ne peut s'effectuer que sur un environnement *VITAM* stable et cohérent, sans possibilité d'ajouter des « ingests » et sans travail de fond (jobs de sécurisation, ...). De plus, le contenu restauré doit être cohérent avec le contenu des offres de stockage sous-jacentes.

5.6.2.1 mongoDB

Il faut d'abord procéder au rapatriement dans `${output_dir}` de la sauvegarde à appliquer.

Avertissement : une sauvegarde ne peut se restaurer que sur un environnement dans la même version.

La commande suivante est à lancer depuis une machine mongo de *VITAM* possédant le répertoire de sauvegarde à restaurer :

```
mongorestore --host mongodbl.example.net --port 3017 --username vitamdb-admin --password "pass"
${output_dir}/${fichier}
```

Note : Se reporter aux fichiers associés à l'ansible de déploiement pour le mot de passe `vitamdb-admin`

5.6.2.2 Elasticsearch

Il faut d'abord procéder au rapatriement dans `${output_dir}` de la sauvegarde à appliquer.

Commande pour lister les *snapshots* de *vitam_backup* (repository)

```
curl -X GET http://elasticsearch-data.service.${consul_domain}:9200/_snapshot/vitam_
↳backup/
```

Pour lancer une restauration, placer le nom du *snapshot* à la place de **snapshot** dans l'URL suivante

```
curl -X POST http://elasticsearch-data.service.${consul_domain}:9200/_snapshot/vitam_
↳backup/*snapshot*/_restore
```

5.7 Batches et traitements

5.7.1 Curator

Il existe des jobs Curator de :

- fermeture d'index
- suppression d'index fermés

Ces jobs sont lancés via `crontab` toutes les nuits.

5.7.2 Timers systemd

Tous les *timers* systemd décrits ci-dessous sont paramétrables ; un comportement par défaut est appliqué. Se reporter à la procédure *Modifier la fréquence de lancement de certains timers systemd* (page 11) pour la bonne prise en compte.

5.7.2.1 Sécurisation des journaux d'opérations

Un *timer* systemd a été mis au point pour réaliser ces actions :

- *vitam-traceability-operations* (page 49)

Ce *timer* est installé avec le composant `logbook`.

5.7.2.2 Sécurisation des cycles de vie

Des *timers* systemd ont été mis au point pour réaliser ces actions :

- *vitam-traceability-lfc-unit* (page 49)
- *vitam-traceability-lfc-objectgroup* (page 50)

Ces *timers* sont installés avec le composant `logbook`.

5.7.2.3 Sécurisation des offres de stockages

Des *timers* systemd ont été mis au point pour réaliser ces actions :

- *vitam-storage-log-backup* (page 48)
- *vitam-storage-log-traceability* (page 49)

Ces *timers* sont installés avec le composant `storage`.

5.7.2.4 Autres *timers*

Les *timers* suivants sont apportés par le composant `functional_administration`

```
vitam-create-accession-register-symbolic
vitam-functional-administration-accession-register-reconstruction
vitam-rule-management-audit
vitam-functional-administration-reconstruction
```

Les *timers* suivants sont apportés par le composant `metadata`

```
vitam-metadata-store-graph
vitam-metadata-reconstruction
```

5.8 Sauvegarde des données graphe (Log shipping)

La sauvegarde des données graphe des métadonnées (UNIT/GOT) consiste à récupérer au fil de l'eau depuis la base de données (MongoDB) les données graphe par (UNIT/GOT) pour les stocker dans les offres de stockage.

Prudence : En cas de problème de sauvegarde des données *graphe*, on écrit dans le fichier log une [Consistency Error] qu'il conviendra de surveiller.

Avertissement : Si l'instance qui démarre le service de sauvegarde s'arrête, il faut lancer ce service de sauvegarde dans une autre instance.

5.8.1 Déclenchement de la sauvegarde

La sauvegarde des données *graphe* est lancée via un *timer* systemd (`vitam-metadata-store-graph` (page 52)), qui démarre le service systemd associé.

- Le timer se lance chaque 30 minutes (par défaut, modifiable selon le besoin)
- Le sauvegarde des données *graphe* se fait sur un intervalle de temps (depuis la dernière sauvegarde jusqu'à temps présent)
- Le fichier généré est un fichier au format zip, qui contient un ou plusieurs fichiers JSON. Ces fichiers JSON contiennent un tableau de données *graphe*.
- Le nom du fichier de la dernière sauvegarde contient les dates début et fin de sauvegarde. Ce nom est utilisé pour déterminer la dernière date de sauvegarde.
- La sauvegarde des UNIT est séparée de celle des GOT (Deux *containers* distincts dans chaque offre de stockage)

5.8.2 Reconstruction des données *graphe*

La reconstruction des données *graphe* se fait avec le même principe que la reconstruction des méta-données (UNIT/GOT) :

- Gérer l'offset de reconstruction des fichiers de sauvegarde des données *graphe*.
- Mettre à jour uniquement les données *graphe*. Si un document n'est pas trouvé, une création de ce document est faite et ne contiendra que les données du *graphe*.
- De même, la reconstruction des métadonnées ne modifie pas les données *graphe* potentiellement déjà existantes.

- Une purge est faite de tous les documents ayant uniquement les données graphe et qui sont vieux de (1 mois Configurable)
- Les documents ayant uniquement les données graphe ne sont pas indexés dans ElasticSearch.
- La reconstruction est séquentielle (D'abord les métadonnées UNIT/GOT ensuite leur graphe)

Voir aussi :

Se reporter à la procédure de *Reconstruction* (page 19) des métadonnées pour plus d'informations.

5.9 Recalcul des données graphe

Il est possible de recalculer les données du graphe en utilisant une requête *DSL*. En effet, dans le cadre de la procédure de *PRA*, il est nécessaire de pouvoir détecter les unités archivistiques ayant un graphe incohérent (construire le DSL requis) selon la procédure de déclenchement décrite ci-dessous.

Le recalcul de graphe permet de rétablir la cohérence des données *VITAM*.

Prudence : Cette procédure s'applique à partir de la version *VITAM* R8 (1.10.0).

Prudence : En cas de données de graphe incohérentes, le résultat des requêtes *DSL* sur les unités archivistiques pourra être incorrect et l'application des filtres de sécurité définis dans les contrats d'accès pourront ne pas être pris en compte.

5.9.1 Déclenchement

Le recalcul de *graphe* est déclenché par l'appel au point d'API porté par l'URL suivante sur le composant *metadata*

```
http://{{ ip_admin }}:{{ vitam.metadata.port_admin }}/metadata/v1/computegraph
```

Exemple d'appel à l'aide de curl :

```
curl -s -X POST -H "X-Tenant-Id: <tenant>" -H "Content-Type: application/json" --user  
↪ "${VITAM_ADMIN_AUTH}" --data @${CURRENT_DIR}/dslQuery.json ${URL}
```

Exemple de query DSL (dslQuery.json) :

```
{
  "$roots": [
    "aeaqaiaaaqhdytymabdeialenehzphiaaaeq",
    "aeaqaiaaaqhdytymabdeialenehzpbyaaajq"
  ],
  "...", "guid_n"
],
  "$query": [],
  "$projection": {}
}
```

La valeur utilisée pour la basic authentication sont issues du fichier *vault.yaml* et prennent la forme suivante

```
VITAM_ADMIN_AUTH={{ admin_basic_auth_user }}:{{ admin_basic_auth_password }}
```

- Le paramètre `adminUser` correspond à la valeur `admin_basic_auth_user` déclarée dans le fichier `vitam_security.yml`
- Le paramètre `adminPassword` correspond à la valeur `admin_basic_auth_password` déclarée dans le fichier `vault-vitam.yml`

Prudence : Si le *DSL* ne contient pas uniquement `$root`, alors la valeur du tenant positionnée dans le header `X-Tenant-Id` est essentielle car les index elasticsearch sont organisés par tenant.

5.10 Montée de version du fichier de signature de Siegfried

La solution logicielle *VITAM* utilise l'outil Siegfried pour la détection des formats des fichiers contenus dans les *SIP*. La nomenclature des formats est basée sur la norme « DROID » fournie par les *archives nationales anglaises*⁸.

Ce fichier est régulièrement mis à jour et il est recommandé de procéder à sa mise en place dans *VITAM*.

Pour ce faire, il faut récupérer le fichier *PRONOM*⁹ depuis les archives nationales anglaises et l'ajouter dans `deployment/environments`.

Il faut ensuite modifier la valeur de la directive `droid_filename` dans le fichier `deployment/environments/group_vars/all/vitam_vars.yml` (avec le nom du fichier récupéré précédemment).

Enfin, il faut lancer le *playbook* ansible suivant depuis `deployment`

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

ou bien, selon votre cas d'usage du *vault* ansible

```
``ansible-playbook -i <fichier d'inventaire> ansible-vitam/roy_build_signature.yml --  
↪vault-password-file vault_pass.txt``
```

5.11 Griffins

Note : nouveauté R9.

Afin de prendre en compte des considérations de réidentification et/ou préservation, la solution logicielle *VITAM* intègre désormais des *griffins* - greffons - pour réaliser les actions d'analyse, (ré)identification et préservation.

Comme décrit dans le *DIN*, le choix des *griffins* installés est défini dans le fichier `environments/group_vars/all/vitam_vars.yml` au niveau de la directive `vitam_griffins`.

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

⁸<http://www.nationalarchives.gov.uk/>

⁹<http://www.nationalarchives.gov.uk/information-management/manage-information/preserving-digital-records/droid/>

5.11.1 Ajout de nouveaux / mise à jour de *griffins*

Il est possible d'ajouter ou mettre à jour des *griffons* à une installation de la solution logicielle :term'VITAM'.

5.11.1.1 Ajout de *griffins*

Pour cela, il faut modifier le fichier `environments/group_vars/all/vitam-vars.yml` au niveau de la directive `vitam_griffins` par ajout du/des *griffon(s)* dans la liste.

Exemple d'ajout du greffon **vitam-un-nouveau-greffon-qui-est-necessaire**

```
vitam_griffins: ["vitam-imagemagick-griffin", "vitam-jhove-griffin", "vitam-un-nouveau-
↪greffon-qui-est-necessaire"]
```

5.11.1.2 Mise à jour des *griffins*

Dans le cadre d'une montée de version des composants *griffins*, le *playbook* se charge de déployer les composants les plus à jour ; il n'est pas nécessaire de modifier la directive `vitam_griffins`.

Note : Ne pas oublier, sur les partitions associées, de mettre à jour, si nécessaire, l'adresse du dépôt de binaires.

5.11.1.3 Préparation du système

Il faut également prévoir de mettre à disposition sur le(s) dépôt(s) de binaires les packages d'installation correspondants. Le nom indiqué dans la liste doit correspondre au nom du package (format rpm ou deb selon la plateforme). Les packages d'installation associés doivent se situer dans un dépôt accessible et connu par la machine sur laquelle ils vont être installés.

5.11.1.4 Prise en compte technique par VITAM

Enfin, il suffit de relancer le *playbook* d'installation de *VITAM* avec, en fin de ligne, cette directive

```
--tags griffins
```

5.12 Reconstruction

La reconstruction consiste à recréer le contenu des bases de données (MongoDB-data, Elasticsearch-data) en cas de perte de l'une ou l'autre à partir des informations présentes dans les offres de stockage. Elle part du principe que le contenu des offres n'a pas été altéré.

Prudence : Dans cette version de la solution logicielle *VITAM*, la reconstruction nécessite de couper le service aux utilisateurs.

Prudence : Une reconstruction complète à partir des offres de stockage peut être extrêmement longue, et ne doit être envisagée qu'en dernier recours.

5.12.1 Procédure multi-sites

5.12.1.1 Cas du site primaire

La reconstruction se réalise de la manière suivante :

1. Arrêt de *VITAM* sur le site à reconstruire

- Utiliser le playbook `ansible-vitam-exploitation/stop_vitam_timers.yml`
- Utiliser le playbook `ansible-vitam-exploitation/stop_vitam.yml`

Il est indispensable de valider que tous les services *VITAM* (y compris les *timers* `systemd`) sont bien arrêtés

2. Purge des données (le cas échéant) stockées dans MongoDB-data, excepté les bases **identity**, **config** et **admin**.
3. Purge des données (le cas échéant) stockées dans Elasticsearch-data,
4. Reconfiguration et démarrage en tant que site secondaire :
 - Paramétrer la variable `primary_site` à `false` dans le fichier `environmenrs/group_vars/all/vitam_vars.yml` puis utiliser le playbook `ansible-vitam/vitam.yml`
5. Dès lors, l'accès utilisateur reste coupé, et l'intégralité des données est reconstruite progressivement
 - Le suivi de la reconstruction se fait en observant l'évolution de l'offset de reconstruction stocké dans MongoDB-data
 - Pour la release 8, la procédure est décrite dans la section « Recalcul des données graphe »
6. La collection `Offset` de la base de données `metadata` est créée et permet de suivre l'avancement de la reconstruction.
7. Une fois la reconstruction terminée (plus de modification dans la collection `Offset`), il convient de reconfigurer en tant que site primaire, puis redémarrer :
 - Paramétrer la directive `primary_site` à `true` puis utiliser le playbook `ansible-vitam/vitam.yml`

5.12.1.2 Cas du site secondaire

La reconstruction se réalise de la manière suivante :

1. Arrêt de *VITAM* sur le site à reconstruire

- Utiliser le playbook `ansible-vitam-exploitation/stop_vitam_timers.yml`
- Utiliser le playbook `ansible-vitam-exploitation/stop_vitam.yml`

Il est indispensable de valider que tous les services *VITAM* (y compris les *timers* `systemd`) sont bien arrêtés.

2. Purge des données (le cas échéant) stockées dans MongoDB-data, excepté les bases **identity**, **config** et **admin**.
3. Purge des données (le cas échéant) stockées dans Elasticsearch-data,
4. Redémarrage du site secondaire Vitam
 - Utiliser le playbook `ansible-vitam-exploitation/start_vitam.yml`
 - La prochaine itération de reconstruction au fil de l'eau redémarrera la reconstruction à partir du début
 - Attendre la fin de la reconstruction au fil de l'eau sur le site secondaire
 - Le suivi de la reconstruction se fait en observant l'évolution de l'offset de reconstruction stocké dans MongoDB-data
 - Pour la release 7 (version 1.4.x) il faut lancer le service dédié `vitam-metadata-graph-builder.service` sur le composant `metadata` pour recalculer le graphe des unités archivistiques et des groupes d'objets techniques n'ayant pas encore reconstruit leurs données graphe

5.12.2 Procédure mono-site

La procédure à appliquer est la même que la procédure du site primaire pour une installation multi-sites.

5.13 Plan de Reprise d'Activité (PRA)

Le *PRA* consiste à passer un site *VITAM* secondaire en site primaire après incident majeur survenu sur le site primaire (cas de l'indisponibilité complète du site primaire).

Note : Les actions en cours sur le site primaire sont perdues (versements non terminés, batchs etc.). L'incohérence des données sera traitée dans une version ultérieure de *VITAM*..

Cette section décrit des actions qui ne peuvent donc s'effectuer que si une installation multi-sites a été effectuée au préalable.

Cette section s'appuie sur les procédures décrites dans les chapitres suivants :

- Resynchronisation d'une offre à partir d'une autre offre (*Resynchronisation d'une offre* (page 23))
- Reconstruction des bases de données (MongoDB-data, Elasticsearch-data) en cas de perte de l'une ou l'autre, à partir des informations présentes dans les offres (*Reconstruction* (page 19))

5.13.1 Déclenchement

Avant le déclenchement de la procédure de *PRA*, le système fonctionne en mode multi-sites (primaire/secondaire). Le service est indisponible à la suite de la perte du site primaire.

Le déclenchement du *PRA* s'effectue selon la procédure suivante :

1. Vérifier que le site primaire est bien complètement arrêté
 - Il est indispensable de valider que tous les services VITAM (y compris les *timers* systemd) sont bien arrêtés
2. Attendre la fin de la reconstruction au fil de l'eau sur le site secondaire
 - Le suivi de la reconstruction se fait en observant l'évolution de l'offset de reconstruction stocké dans MongoDB-data.
 - Pour la release 7 (version 1.4.x) il faut lancer le service dédié `vitam-metadata-graph-builder.service` sur le composant metadata pour recalculer les données graphe des unités archivistiques et des groupes d'objets techniques n'ayant pas encore reconstruit leurs données graphe
3. Reconfigurer le site secondaire en site primaire :
 - Si le site secondaire était partiellement déployé, ne pas oublier de rajouter tous les composants requis pour un fonctionnement en site primaire
 - Attention à adapter la stratégie de stockage en fonction du mode d'utilisation choisi pour le site de secours :
 - Mode « lecture/écriture » : la stratégie de stockage doit être modifiée afin de limiter les écritures aux seules offres encore disponibles sur le site de secours (*Activation/désactivation d'une offre* (page 31))
 - Mode « lecture seule » (recherche et consultation avec profil de droits dédié) : la stratégie de stockage ne change pas. Seule une reconfiguration du site primaire initial en mode secondaire permettra le retour à un fonctionnement nominal (cf. ci-dessous)
 - Paramétrer la variable `primary_site` à `true` puis jouer le *playbook* `ansible-vitam/vitam.yml`
4. En lien avec le processus de reconstruction (cf. *Reconstruction* (page 19)), en cas de bascule sur le site secondaire, il sera préférable de purger les documents des Unit et ObjectGroup reconstruits mais ne contenant potentiellement que des données de graphe (cas de l'élimination par exemple). Cette opération s'effectue avec la commande suivante :


```
curl -s -X DELETE -H "X-Tenant-Id: {{ vitam_tenant_admin }}" -H "Accept: application/
↪json" -H "Content-Type: application/json" --user "{{ admin_basic_auth_user }}:{{ _
↪admin_basic_auth_password }}" http://{{ ip_admin }}:{{ vitam.metadata.port_admin }}/
↪metadata/v1/purgeGraphOnlyDocuments/[UNIT | OBJECTGROUP | UNIT_OBJECTGROUP]
```

Après modification des accès pour les applications versantes (action infra. de type modification DNS, routage, conf etc.), le site secondaire peut alors être ouvert au service en tant que site primaire.

Le système fonctionne désormais en mode mono-site (primaire). Le service est de nouveau disponible sur le site de secours.

5.13.2 Retour en situation nominale

Le retour à la solution nominale s'effectue en deux étapes :

- Rétablissement du contenu du site primaire initial par reconfiguration temporaire en tant que site secondaire
- Retour à la configuration multi-sites initiale

Avertissement : Dans cette version, la resynchronisation partielle d'une offre de stockage n'étant pas supportée, le retour à la configuration multi-sites initiale nécessite de repartir d'offres vierges de toutes données sur le site à resynchroniser (on parle ici d'offre de remplacement)

5.13.2.1 Déclenchement

Avant déclenchement de la procédure de *PRA* inverse (retour en situation nominale), le système fonctionne en mode mono-site (primaire). Le service est disponible sur le site de secours.

Le déclenchement du *PRA* inverse s'effectue selon la procédure suivante :

- Vérifier que le site primaire initial est bien complètement arrêté
 - Il est indispensable de valider que tous les services VITAM (y compris les *timers* systemd) sont bien arrêtés
- Purger les données (le cas échéant) stockées dans MongoDB-data, excepté les bases **identity**, **config** et **admin**
- Purger les données (le cas échéant) stockées dans Elasticsearch-data
- Reconfigurer et démarrer le site primaire initial en tant que site secondaire :
 - Paramétrer la variable `primary_site` à `false` puis utiliser le playbook `ansible-vitam/vitam.yml`
 - Le mécanisme de reconstruction du contenu des bases de données (MongoDB-data, Elasticsearch-data) à partir des informations présentes dans les offres de stockage est actif (aucune donnée à resynchroniser à cette étape)
- Resynchroniser les offres de stockage à partir des offres du site de secours en se référant à la procédure suivante *Resynchronisation d'une offre* (page 23)
 - En fonction du mode d'utilisation choisi pour le site de secours :
 - Mode lecture/écriture : la stratégie de stockage du site de secours doit auparavant être modifiée afin de référencer de nouveau les offres du site primaire initial
 - Mode lecture seule : la stratégie de stockage ne change pas. Les offres du site primaire initial sont toujours connues du site de secours
- Le mécanisme de reconstruction au fil de l'eau reconstruit progressivement le contenu des bases de données
 - Le suivi de la reconstruction se fait en observant l'évolution de l'offset de reconstruction stocké dans MongoDB data

- Pour la release 7 (version 1.4.x) il faut lancer le service dédié `vitam-metadata-graph-builder.service` sur le composant metadata pour recalculer les données graphe des unités archivistiques et des groupes d'objets techniques n'ayant pas encore reconstruit leurs données graphe
- Une fois la reconstruction terminée, reconfiguration en tant que site primaire et démarrage :
 - Paramétrer la variable `primary_site` à `true` puis utiliser le playbook `ansible-vitam/vitam.yml`
- Reconfiguration et démarrage en tant que site secondaire du site de secours :

Avertissement : Cette opération provoque une indisponibilité temporaire des principaux services *VITAM* (versement, gestion, recherche et consultation)

- Paramétrer la variable `primary_site` à `false` puis utiliser le playbook `ansible-vitam/vitam.yml`

Après modification des accès pour les applications versantes (action infra. de type modification DNS, routage, conf etc.), le site primaire initial peut alors être de nouveau ouvert au service en tant que site primaire.

Le système fonctionne désormais de nouveau en mode multi-sites (primaire/secondaire). Le service est de nouveau disponible sur le site primaire initial.

5.14 Resynchronisation d'une offre

Une offre de stockage peut être désynchronisée par rapport à une autre à la suite d'une indisponibilité plus ou moins longue voire totale de l'offre (*crash* majeur du système, panne matérielle etc.) ou bien encore à la suite d'une mise en maintenance programmée.

Le mécanisme de resynchronisation d'une offre par rapport à une autre nécessite une intervention d'exploitation manuelle permettant de remédier à la perte de données dans le système.

Note : En cas d'indisponibilité d'une offre, le processus d'entrée d'un *SIP* n'étant réussi que si et seulement si toutes les offres de stockage définies dans la stratégie sont accessibles, et que tous les fichiers sont bien copiés sur la totalité de ces offres, il sera nécessaire de désactiver l'offre (cf. chapitre *Activation/désactivation d'une offre* (page 31)) afin de permettre à nouveau les entrées de *SIP* (ingest/versement).

5.14.1 Cas de l'ajout d'une nouvelle offre

Avertissement : Lors de l'ajout d'une nouvelle offre (portant un nouvel identifiant d'offre), les métadonnées des AU / GOT existants ne seront pas mis à jour avec l'information sur la nouvelle stratégie de stockage utilisée. L'ajout d'un mécanisme de mise à jour / propagation des métadonnées est prévu dans une version ultérieure. Lors de l'ajout d'une offre en remplacement d'une précédente offre, l'intégrité des métadonnées sera garantie en conservant le même identifiant d'offre.

L'ajout d'une nouvelle offre de stockage requiert le déploiement des applicatifs *VITAM* associés selon la procédure suivante :

- Editer le fichier de configuration de l'inventaire de déploiement (généralement, fichier `hosts`) afin d'ajouter les nouveaux serveurs portant les composants à déployer (fonction de la topologie de déploiement retenue) :

```
[hosts-storage-offer-default]
hostname-new-storage-offer      offer_conf=<offer-z>

[hosts-mongos-offer]
hostname-new-mongos-offer       offer_conf=<offer-z>

[hosts-mongoc-offer]
hostname-new-mongoc-offer       offer_conf=<offer-z>

[hosts-mongod-offer]
hostname-new-mongod-offer       offer_conf=<offer-z>
```

- Editer le fichier de configuration de la strategie de stockage `offer_opts.yml` afin d'ajouter la nouvelle offre :

```
vitam_strategy:
- name: <offer-x>
  referent: true
# <offer-z> is the new offer
- name: <offer-z>
  referent: false

vitam_offers:
  <offer-x>:
    provider: filesystem
  # <offer-z> is the new offer
  <offer-z>:
    provider: filesystem
```

- Editer le fichier de déclaration des secrets généraux `vault-vitam.yml` afin d'y ajouter les secrets associés à la nouvelle offre :

```
mongodb:
  <offer-z>:
    passphrase: <passphrase>
    admin:
      user: <admin-user>
      password: <admin-password>
    localadmin:
      user: <localadmin-user>
      password: <localadmin-password>
    offer:
      user: <offer-user>
      password: <offer-password>
```

- Exécuter la commande suivante afin de déployer les nouveaux composants `storage-offer`, `mongos-offer`, `mongoc-offer`, `mongod-offer` :

Note : On considère que les étapes de génération des *hostvars*, de génération des magasins de certificats et de mise en place des repositories *VITAM* ont été réalisées au préalable pour les serveurs concernées (se référer aux sections du *DIN* correspondantes).

```
ansible-playbook -i environments/<hosts> -l "hostname-new-storage-offer,hostname-new-
↳ mongos-offer,hostname-new-mongoc-offer,hostname-new-mongod-offer" ansible-vitam/
↳ vitam.yml --ask-vault-pass
```

La nouvelle offre doit ensuite être déclarée dans la stratégie de stockage par reconfiguration du moteur de stockage selon la procédure suivante :

Avertissement : Cette opération provoque une indisponibilité temporaire des principaux services *VITAM* (versement, gestion, recherche et consultation)

- Exécuter la commande suivante afin de reconfigurer le composant storage-engine :

```
ansible-playbook -i environments/<hosts> -l hosts-storage-engine ansible-
↪vitam/vitam.yml --ask-vault-pass --tags update_vitam_configuration
```

5.14.2 Cas de la resynchronisation d'une offre temporairement indisponible

La resynchronisation d'une offre à partir du contenu d'une autre offre s'effectue en suivant la procédure suivante :

Note : Cette procédure n'impacte pas les services *VITAM*. Le mécanisme de reconstruction du contenu des bases de données (MongoDB-data, Elasticsearch-data) à partir des informations présentes dans les offres de stockage fonctionne de manière concurrente au mécanisme de resynchronisation.

- Exécuter la commande suivante afin de resynchroniser la nouvelle offre vis-à-vis de l'offre (des offres) source(s) :

```
curl -v -X POST -u adminUser:adminPassword http://<offer-x.hosts-storage-
↪offer-default>:29102/storage/v1/offerSync < query
{
  "sourceOffer": "<offer-x>.service.consul",
  "targetOffer": "<offer-z>.service.consul",
  "container": <container>,
  "tenantId": <tenantId>
}
```

- Le paramètre `adminUser` correspond à la valeur `admin_basic_auth_user` déclarée dans le fichier `vitam_security.yml`
- Le paramètre `adminPassword` correspond à la valeur `admin_basic_auth_password` déclarée dans le fichier `vault-vitam.yml`
- Le paramètre `sourceOffer` correspond à l'id de l'offre source utilisée pour la resynchronisation de la nouvelle offre
- Le paramètre `targetOffer` correspond à l'id de l'offre à resynchroniser
- Suivre les journaux de la resynchronisation dans les logs du composant storage offer avec la commande suivante :

```
tail -F /vitam/log/storage/storage_offer_sync.\*.log
```

- Vérifier l'état d'exécution de la synchronisation via la commande (peut être scriptée) :

```
curl -v -X HEAD -i -u adminUser:adminPassword http://<offer-x.hosts-
↪storage-offer-default>:29102/storage/v1/offerSync
```

L'entête `Running` indique l'état d'exécution de processus de synchronisation.

- Vérifier le détail d'exécution de la synchronisation via la commande :

```
curl -v -X GET -u adminUser:adminPassword http://<offer-x.hosts-storage-
↪offer-default>:29102/storage/v1/offerSync
```

- En cas de resynchronisation partielle d'une offre, il est possible d'exécuter le processus de resynchronisation à partir d'un `offset` :

```
curl -v -X POST -u adminUser:adminPassword http://<hosts-storage-offer-
↪default>:29102/storage/v1/offerSync < query
{
  "sourceOffer": "<offer-x>.<consul_domain>",
  "targetOffer": "<offer-z>.<consul_domain>",
  "offset": <offset>,
  "container": <container>,
  "tenantId": <tenantId>
}
```

- Le paramètre `offset` correspond à la valeur du dernier `offset` observé dans les logs du composant `storage offer` (cas d'une reprise suite à interruption ou échec de la procédure de resynchronisation). Le paramètre `offset` peut également être déterminé via les enregistrements de la collection `OfferLog` (database `offer`) depuis la base MongoDB associée à l'offre à resynchroniser (cas d'une panne ou d'une mise en maintenance programmée à une date précise).

5.15 Procédure d'exploitation suite à la création ou la modification d'une ontologie

Au préalable à la création ou à la modification d'une ontologie, les index Elasticsearch correspondant aux ontologies doivent être créés ou mis à jour.

5.15.1 Création d'une ontologie

Suite à la création d'une nouvelle ontologie, les index Elasticsearch doivent être mis à jour selon la procédure suivante :

- Dans le cas d'une création, il suffit de créer un nouveau mapping dans les index concernés.

Ex : Ajout d'une propriété `Licence` dans tous les index `unit` (`unit*` signifiant tous les index `unit_0`, `unit_1` etc ...)

Commandes à lancer sur une des partitions hébergeant le cluster `elasticsearch` « `data` » :

```
curl -XPUT "http://localhost:9200/unit*/_mapping/typeunique?update_all_types" -d'
{
  "properties": {
    "Licence": {
      "type": "text"
    }
  }
}'
```

Pour vérifier sur un ou tous les index `unit` :

```
curl -XGET "http://localhost:9200/unit_0/_mapping/?pretty=true"
```

```
curl -XGET "http://localhost:9200/unit*/_mapping/?pretty=true"
```

5.15.2 Changement de type d'une ontologie existante

Dans ce cas, le changement de type dans elasticsearch n'est pas possible. Il faut donc créer un nouvel index Elastic-Search avec un nouveau mapping, puis reindexer l'ancien index dans ce dernier.

On récupère d'abord l'ancien index

```
curl -XGET 'localhost:9200/unit_1/_mapping?pretty=true'
```

On crée un fichier json et on y copie les données obtenues (ne conserver que la balise « mappings » : { ... } et son contenu). On modifie le mapping en changeant le type des propriétés choisies. On crée un nouvel index on lui passant en paramètre le fichier du nouveau mapping .

```
curl -XPUT "http://localhost:9200/new_unit_1" -H 'Content-Type: application/json' -d @newmapping.json
```

Vérifier l'index :

```
curl -XGET 'localhost:9200/new_unit_1/_mapping/'
```

On reindexe unit_1 vers le nouvel index new_unit_1

```
curl -XPOST 'localhost:9200/_reindex' -H 'Content-Type:application/json' -d '{
  "source" : {
    "index" : "unit_1"
  },
  "dest" : {
    "index" : "new_unit_1",
    "version_type": "external"
  }
}'
```

On efface l'alias de l'ancien index unit_1

```
curl -XDELETE 'localhost:9200/unit_1/_alias/unit_1'
```

et on l'affecte au nouvel index new_unit_1

```
curl -XPUT 'localhost:9200/new_unit_1/_alias/unit_1'
```

Avertissement : les index elasticsearch de *VITAM* sont créés par tenant. Il faudra refaire l'opération ci-dessus pour chaque tenant.

Avertissement : en cas de reindexation des index elasticsearch par le service *REST* de *VITAM*, les données sont réindexées suivant le mapping initial. Les nouveaux mappings ne seront donc pas pris en compte. Ce comportement sera modifié dans le futur.

5.16 Procédure d'exploitation pour la mise en pause forcée d'une opération

Pour permettre le traitement non-concurrent de certaines opérations (ingest et reclassement en particulier), il est possible de pouvoir forcer la mise en pause à la réception d'opérations (toutes ou seulement d'un type donné, sur tous les tenants ou un tenant donné en particulier).

Concrètement, elle permet de forcer le mode « pas à pas » pour toutes ou un type donné seulement d'opérations, sur l'ensemble des tenants ou sur un tenant donné seulement.

5.16.1 Mise en pause forcée

La mise en pause forcée est déclenchée par l'appel au point d'API porté par le composant access-external à l'URL suivante : `http://{{ ip_service }}:{{ vitam.accessexternal.port_service }}/admin-external/v1/forcepause`

Exemple d'appel à l'aide de curl :

```
curl -X POST -k
--key vitam-vitam_1.key
--cert vitam-vitam_1.pem
`https://{{ ip_service }}:{{ vitam.accessexternal.port_service }}/admin-external/v1/
↪forcepause`
-H 'X-Tenant-Id: 0'
-H 'X-Access-Contract-Id: ContratTNR'
-H 'Content-Type: application/json;charset=UTF-8'
-H 'Accept: application/json'
--data-binary '{"type" : "INGEST", "tenant" : "0"}'
--compressed
```

Exemple de Json pour mettre une pause sur le processus d'ingest pour le tenant 0 :

```
'{"type" : "INGEST", "tenant" : "0"}'
```

Exemple de Json pour mettre une pause sur tous les processus pour le tenant 0 :

```
'{"tenant" : "0"}'
```

Exemple de Json pour mettre une pause sur tous les processus pour tous les tenants :

```
'{"pauseAll":true}'
```

5.16.2 Sortie de la mise en pause forcée

La sortie de mise en pause forcée est déclenchée par l'appel au point d'API porté par le composant access-external à l'URL suivante : `http://{{ ip_service }}:{{ vitam.accessexternal.port_service }}/admin-external/v1/removeforcepause`

Exemple d'appel à l'aide de curl :

```
curl -X POST -k
--key vitam-vitam_1.key
--cert vitam-vitam_1.pem
`https://{{ ip_service }}:{{ vitam.accessexternal.port_service }}/admin-external/v1/
↪removeforcepause`
```

(suite sur la page suivante)

(suite de la page précédente)

```
-H 'X-Tenant-Id: 0'
-H 'X-Access-Contract-Id: ContratTNR'
-H 'Content-Type: application/json;charset=UTF-8'
-H 'Accept: application/json'
--data-binary '{"type" : "INGEST", "tenant" : "0"}'
--compressed
```

Exemple de Json pour sortir de la mise en pause sur le processus d'ingest pour le tenant 0 :

```
'{"type" : "INGEST", "tenant" : "0"}'
```

Exemple de Json pour sortir de la mise en pause sur tous les processus pour le tenant 0 :

```
'{"tenant" : "0"}'
```

Exemple de Json pour sortir de la mise en pause sur tous les processus pour tous les tenants :

```
'{"pauseAll":false}'
```

Avvertissement : Les états de mise en pause ne sont pas sauvegardés. En cas de redémarrage des applications (en particulier le composant access-external), ces états sont perdus.

5.17 Réindexation

Cette procédure consiste à réindexer le contenu des bases de données Elasticsearch-data (cluster d'indexation dédié aux données métier) en cas de perte ou d'inconsistance de données, à partir des informations présentes dans les bases de données MongoDB-data (replicaset MongoDB stockant les données métier de Vitam). Elle part du principe que le contenu des collections MongoDB-data n'a pas été altéré et que les différents index Elasticsearch-data sont toujours existants.

5.17.1 Déclenchement

La réindexation se déclenche de la manière suivante :

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

Ce playbook s'assure que le composant vitam-functional-administration est démarré, puis procède à la réindexation et au *re-aliasing* (bascule sur le nouvel index) des collections suivantes :

- unit
- logbookoperation
- objectgroup
- securityprofile
- context
- ontology
- ingestcontract
- agencies
- accessionregisterdetail

- archiveunitprofile
- accessionregistersummary
- accesscontract
- fileformat
- filerules
- profile
- griffin
- preservationscenario

Prudence : La réindexation de la collection griffin n'est pas utilisable dans cette version (bug 5762).

Prudence : La purge des anciens index n'est pas réalisée par cette procédure scriptée et est laissée à la charge de l'exploitant.

5.18 Procédure d'exploitation pour la révocation des certificats SIA et Personae

Cette section fait référence au chapitre *Intégration d'une application externe dans Vitam* (page 168).

La version 1.10.0 (« R8 ») introduit une nouvelle fonctionnalité permettant la révocation des certificats *SIA* et *Personae* afin d'empêcher des accès non autorisés aux *API* de la solution logicielle *VITAM* (vérification dans la couche https des *CRL*).

Le fonctionnement de la validation des certificats de la solution logicielle *VITAM SIA* et *Personae* par *CRL* est le suivant :

- L'administrateur transmet à la solution logicielle *VITAM* le *CRL* d'un *CA* qui a émis le certificat présent dans la solution logicielle *VITAM*, via le point d'API suivant

```
http://{{hosts-security-internal}}:{{vitam.security_internal.port_admin}}/v1/api/
→crl
```

Prudence : La CRL fournie doit être obligatoirement au format DER (cf. <http://www.ietf.org/rfc/rfc3280.txt> »>RFC 3280 : Internet X.509 Public Key Infrastructure Certificate and CRL Profile)

Exemple :

```
curl -v -X POST -u {{ admin_basic_auth_user }}:{{ admin_basic_auth_password }} http://
→/{{hosts-security-internal}}:{{vitam.security_internal.port_admin}}/v1/api/crl -H
→'Content-Type: application/octet-stream' --data-binary @/path/to/crl/my.crl
```

Le paramètre `adminUser` correspond à la valeur `admin_basic_auth_user` déclarée dans le fichier `vitam_security.yml`

Le paramètre `adminPassword` correspond à la valeur `admin_basic_auth_password` déclarée dans le fichier `vault-vitam.yml`

- Le système va contrôler tous les certificats (collections `identity.Certificate` et `identity.PersonalCertificate`) émis par le *IssuerDN* correspondant à la *CRL*, en vérifiant si ces derniers sont révoqués ou non. Si c'est le cas, alors la solution logicielle *VITAM* positionne le statut du certificat révoqué à **REVOKED**. Cela a pour conséquence le rejet de tout accès aux *API VITAM* avec utilisation du certificat révoqué (les filtres de sécurité émettront des exceptions dans les journaux de *log*).
- Une alerte de sécurité est émise dans les journaux en cas de révocation.

5.19 Activation/désactivation d'une offre

Prudence : Ne pas oublier, en cas de désactivation d'une offre considérée référente par *VITAM*, de déclarer une autre offre (contenant les données) comme nouvelle référente (modifications à apporter dans `deployment/environments/group_vars/all/offers_opts.yml` par la directive `referent: true`).

Lors de la détection de la perte d'une offre de stockage ou lors d'une maintenance programmée sur celle-ci, afin de permettre à nouveau les versements, l'offre pourra être désactivée.

La désactivation d'une offre s'effectue selon la procédure suivante :

- Exécuter la commande suivante afin de désactiver l'offre :

```
ansible-playbook -i environments/<hosts> ansible-vitam-exploitation/
↳ activate_vitam_offer.yml --ask-vault-pass
```

Rentrer (entre quotes) le nom de l'offre (parmi les offres listées par la question) à activer/désactiver en réponse à la question suivante :

```
Which offer do you want to disable ?
  nom-offre-1
  nom-offre-2
  ...
  nom-offre-n
: "nom-offre-x"
```

Choisir l'action à réaliser en répondant à la question suivante :

```
Do you want to enable (yes) or to disable (no) the chosen offer ?
Answer with ('yes'|'no') : no
```

Signification de la réponse :

- **yes** : active l'offre choisie
- **no** : désactive l'offre choisie

Avertissement : Cette opération provoque une indisponibilité complète de *VITAM*

- Exécuter la commande suivante afin de reconfigurer le composant storage-engine :

```
ansible-playbook -i environments/<hosts> -l hosts-storage-engine ansible-
↳ vitam/vitam.yml --ask-vault-pass --tags update_vitam_configuration
```

Avertissement : Cette opération provoque une indisponibilité temporaire des principaux services *VITAM* (versement, gestion, recherche et consultation)

5.20 Nettoyage d'un environnement

Un *playbook* ansible de nettoyage d'un environnement est fourni et permet de purger un environnement afin de le réinitialiser à son état presque initial.

Le nettoyage d'un environnement s'effectue selon la procédure suivante :

- Exécuter la commande suivante afin de nettoyer l'environnement (offres de stockage, workspace et bases de données) :

```
ansible-playbook -i environments/<hosts> ansible-vitam-exploitation/
  ↪cleaning.yml --ask-vault-pass
```

En détails, le nettoyage d'un environnement va exécuter la liste des actions suivantes :

1. Arrêt des modules externes, afin que Vitam ne puisse plus accepter de demandes provenant de l'extérieur.
2. Purge des collections MongoDB : LogbookLifeCycleObjectGroup, LogbookLifeCycleObjectGroupInProgress, LogbookLifeCycleUnit, LogbookLifeCycleUnitInProgress, LogbookOperation, AccessionRegisterDetail, AccessionRegisterSummary, AccessionRegisterSymbolic, ObjectGroup, Unit, EliminationActionObjectGroup, EliminationActionUnit, PreservationReport.
3. Réindexation des collections à indexer et purgées en phase 2 : le but étant d'obtenir en fin de traitement des indexes vides (ne contenant aucun document).
 - LogbookOperation, ObjectGroup & Unit : une réindexation sera effectuée pour chaque tenant configuré (ex : unit_0_20190130_104530, unit_1_20190130_104540, etc...).
 - AccessionRegisterDetail, AccessionRegisterSummary, AccessionRegisterSymbolic : 3 nouveaux indexes seront créés au total pour les 3 collections (ex : accessionregistersymbolic_20190121_133507, accessionregistersummary_20190121_133503 & accessionregisterdetail_20190121_133505)
4. Nettoyage des offres de stockage. Selon la configuration de l'environnement, le script est en charge de nettoyer chaque offre de stockage configurée :
 - Pour chaque tenant configuré dans Vitam, le script va supprimer tous les sous-containers ainsi que leurs contenus exceptés : « backup » (contenant les référentiels intégrés dans Vitam) et « rules ».
 - Pour l'offre File-System, chaque sous-containers supprimé et vidés de son contenu, sera recréé à vide (ex : int_0_accessionregisterdetail)
5. Nettoyage du *workspace* : le contenu des objets contenus dans le workspace sera purgé également.

Avertissement : Pour le moment, seules les offres de stockage S3 et File-System sont prises en compte dans la purge des offres de stockage. Les offres Swift ne font encore pas partie du scope de *VITAM*.

Avertissement : Cette opération provoque une indisponibilité complète de *VITAM*

5.20.1 Etat des lieux après purge

Après le passage du script, l'environnement est purgé :

- Les référentiels sont toujours présents (Contrats d'accès, contrats d'entrée, règles de gestion, etc. . .).
- L'environnement est disponible et utilisable : les modules externes sont accessibles (IngestExternal et AccessExternal).
- Les offres de stockage sont vidées, à l'exception des backups des référentiels.
- La cohérence entre MongoDB et ElasticSearch est assurée. La plupart des collections sont vidées, et les indexes E/S associés ne contiennent aucun document.
- Le workspace est purgé, aucune opération n'est en cours et ne peut être relancée.

5.20.2 Limitations

Le fait de purger les journaux et non les référentiels provoquera une incohérence de la plate-forme vis-à-vis de la norme NFZ-42020 (suppression des logs d'imports de référentiels, mais présence de ceux-ci).

Pour le moment, les offres de stockage de type *Swift* ne sont prises en compte dans le nettoyage d'un environnement. Si le script est lancé dans un environnement configuré avec une (ou plusieurs) offre de stockage de type *Swift*, alors l'offre ne sera pas nettoyée.

6.1 Veille et patchs sécurité

Les éléments d'infrastructure suivants sont particulièrement sensibles pour la sécurité de la solution logicielle *VITAM* et nécessitent d'être intégrés à la veille sécurité du système :

- Runtime Java (OpenJDK 8)

6.2 Métriques

La solution logicielle *VITAM* intègre une solution de *monitoring* des applications à l'aide de métriques. L'exploitant peut, s'il le souhaite, changer la configuration des remontées de métriques, ou bien utiliser celle par défaut proposée dans *VITAM*.

6.2.1 Configuration

6.2.1.1 Activation/désactivation

Il est possible de définir les composants sur lesquels activer/désactiver ces métriques. Pour cela, il faut éditer le fichier `deployment/environments/group_vars/all/vitam_vars.yml` et définir, pour chaque composant, la valeur de la directive `metrics_enabled` (`true/false`).

A l'issue, lancer la commande de déploiement (se reporter au *DIN*).

6.2.1.2 Registres

Par défaut, 3 registres de métriques sont créés pour toutes les applications *VITAM* :

- les métriques de Jersey
- les métriques de la JVM (Java Virtual Machine)

- les métriques « métier »

JERSEY : Les métriques Jersey correspondent à 3 métriques, des *Timers*, des *Meters*, et des *ExceptionMeters* qui vont être enregistrées pour chaque URI des API Rest de VITAM.

- Les *Meters* font office de compteurs. Ils sont incrémentés de 1 chaque fois qu'une URI est requêtée.
- Les *Timers* font office de chronomètres. Ils chronomètrent le temps de réponse d'une URI chaque fois que celle-ci est requêtée.
- Les *ExceptionMeters* font office de compteurs. Ils sont incrémentés de 1 chaque fois qu'une URI soulève une Exception dans le code.

JVM : Les métriques JVM correspondent à des *Gauges* qui enregistrent des valeurs de ressources système utilisées par la Java Virtual Machine pour chaque application VITAM.

BUSINESS : Les métriques métiers correspondent à des métriques de n'importe quel type qui peuvent remonter toute donnée considérée utile dans une application VITAM.

6.2.1.3 Reporters

Par défaut, 2 reporters de métriques sont disponibles pour les applications VITAM. Les reporters de métriques sont en charge de collecter les valeurs des métriques à des intervalles réguliers.

LogBack : le reporter LogBack affiche les valeurs des métriques dans LogBack.

ELASTICSEARCH : le reporter Elasticsearch sauvegarde les valeurs des métriques dans une base de données Elasticsearch qui peut être configurée dans le fichier de configuration.

6.2.1.4 Fichier de configuration

Le fichier de configuration des métriques est situé dans `/vitam/conf/<service_id>/vitam.metrics.conf`. Ce fichier contient la documentation nécessaire pour configurer correctement les métriques. Une description des clés YAML y est disponible.

6.2.2 Métier

Aucune métrique n'a encore été définie à ce stade du projet.

6.2.3 Métriques techniques

6.2.3.1 Métriques système critiques

Aucune métrique n'a encore été définie à ce stade du projet.

6.2.3.2 Indicateurs de SLA

Aucun indicateur n'a encore été défini à ce stade du projet.

6.2.3.3 Indicateurs de performance

Aucun indicateur n'a encore été défini à ce stade du projet.

6.2.4 Visualisation

Si un reporter de type **ElasticSearch** est configuré, alors les métriques peuvent être visualisées via l'application web Kibana ¹⁰.

L'application Kibana comporte 4 sections qui seront développées :

- **Discover**
- **Visualize**
- **Dashboards**
- **Settings**

Néanmoins si vous souhaitez travailler avec Kibana, il est judicieux de consulter la documentation officielle. Celle-ci n'ayant pour but qu'une présentation sommaire de l'outil.

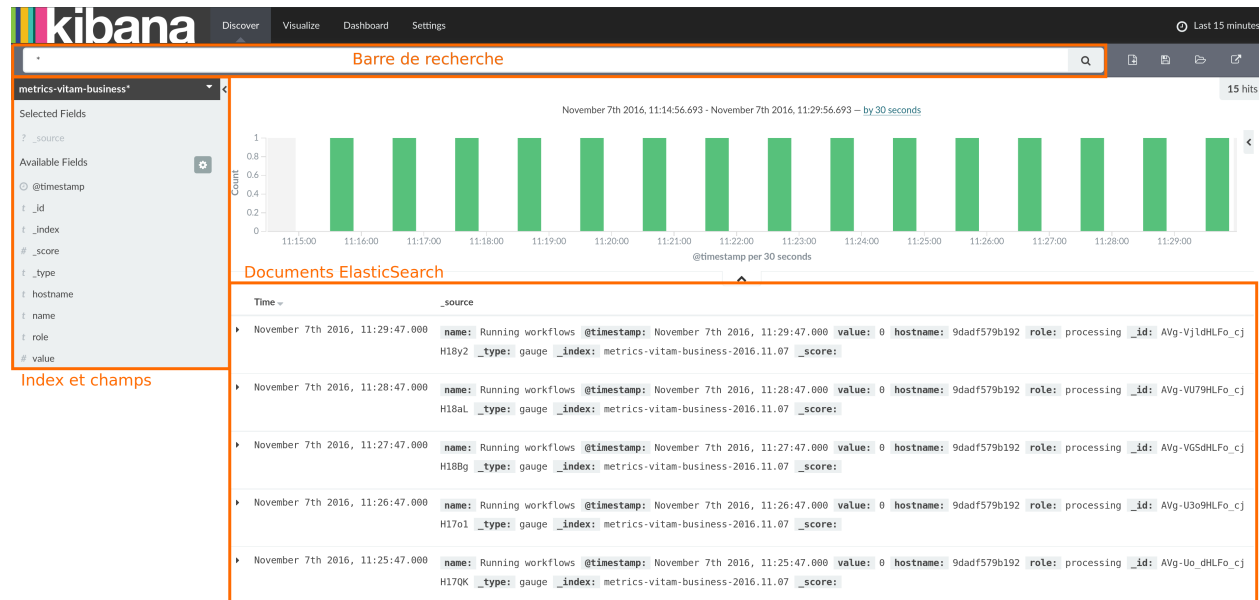
Voir aussi :

Documentation officielle de Kibana ¹¹

6.2.4.1 Discover

La section **Discover** permet de consulter rapidement les données présentes dans un index d'ElasticSearch. Pour cela, il suffit de sélectionner un index dans la barre latérale gauche, de choisir les champs que l'on souhaite consulter (optionnel) et les données apparaissent triées par ordre chronologique décroissant.

Il est possible d'effectuer des recherches poussées sur les documents, comme des expressions régulières, grâce à la barre de recherche en haut de la page. Une fois la recherche exécutée, il peut être utile de la sauvegarder afin de la réutiliser pour des visualisations.



6.2.4.2 Visualize

La section **Visualize** permet de consulter les données présentes dans ElasticSearch à travers différents graphiques statistiques. Les graphiques disponibles sont :

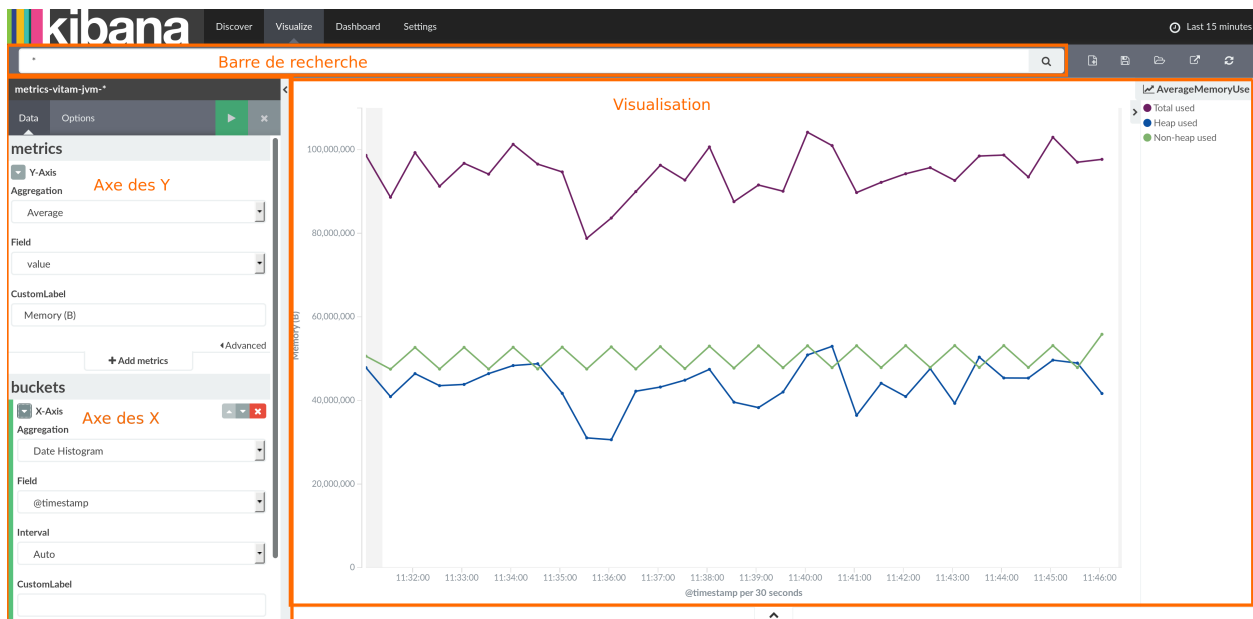
<https://www.elastic.co/fr/products/kibana>

<https://www.elastic.co/guide/en/kibana/current/index.html>

- **Area chart** : utile pour un regroupement de séries chronologiques dans lequel le total des séries est plus important que la différence entre plusieurs séries.
- **Data table** : un tableau de données classique.
- **Line chart** : graphique pour des séries temporelles. Très utile pour comparer deux séries entre elles.
- **Markdown widget** : utile pour insérer informations sur un dashboard Kibana.
- **Metric** : représentation d'une aggrégation de données sous la forme d'un seul nombre.
- **Pie chart** : un diagramme circulaire classique.
- **Tile map** : représentation de coordonnées géographiques sur une carte.
- **Vertical bar chart** : un histogramme classique.

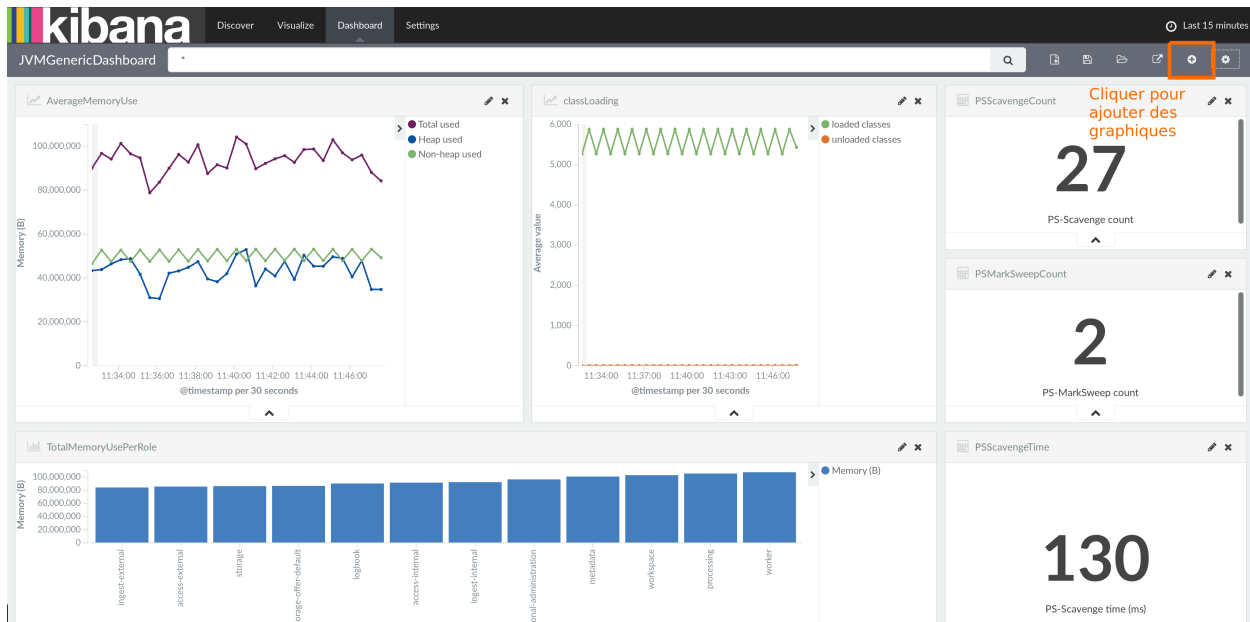
La barre latérale gauche du panneau de visualisation permet de configurer la donnée à représenter. Pour l'axe des Y, il est impératif d'utiliser un aggrégation (moyenne, minimum/maximum, écart type. . .) sur une valeur pour la représenter. En fonction du graphique sélectionné, il est possible de configurer l'axe des X, toujours au moyen d'aggrégations (dates, date range, terme. . .).

En haut se situe la même barre de recherche que sur la partie **Discover**, qui permet d'affiner son graphique en effectuant des tris sur sa donnée.



6.2.4.3 Dashboards

La section **Dashboard** permet de regrouper plusieurs graphiques pour constituer un dashboard. Pour ce faire il suffit d'importer des graphiques avec le bouton « + » en haut à droite.



6.3 API de de supervision

Chaque composant *VITAM* peut dialoguer, selon le paramétrage, via 2 réseaux :

- patte d'administration
- patte de service

Si les partitions ne possèdent qu'une seule interface, les deux « pattes » passent par cette unique interface.

6.3.1 Patte d'administration

La solution logicielle *VITAM* expose en interne de la plate-forme les *API REST* suivantes sur ses composants :

- `/admin/v1/status` : statut simple, renvoyant un statut de fonctionnement incluant des informations techniques sur l'état actuel du composant. Un exemple d'utilisation typique est l'intégration à un outil de supervision ou à un élément actif tiers (ex : load-balancer, ...). L'appel doit être peu coûteux.
- `/admin/v1/version` : informations de version, build, commit git ayant servi à builder les différents jar.
- `/admin/v1/autotest` : autotest du composant, lançant un test de présence des différentes ressources requises par le composant et renvoyant un statut d'état de ces ressources.

6.3.1.1 /admin/v1/status

L'API de status renvoie un fichier JSON contenant les informations suivantes :

```
{
  "serverIdentity": {
    "Name": "vitam-iaas-app-01",
    "Role": "logbook",
    "PlatformId": 425367
  },
  "status": true,
  "detail": {
```

(suite sur la page suivante)

(suite de la page précédente)

```

    },
    "componentsVersions": {
      "e2eb99d93a74409b3ebc5224e596953e9b8a178f": 18
    }
  }
}

```

Signification des champs :

- **serverIdentity**
 - Name : hostname du serveur hébergeant le composant (type : texte)
 - Role : Nom du composant (type : texte)
 - PlatformId : ID de l'environnement (type : entier)
- status : Statut du composant (OK/KO) (type : booléen)
- detail : vide dans cette version, sera défini ultérieurement
- **componentsVersions**
 - hash de commit git : nombre de jars avec buildés depuis ce hash

6.3.1.2 /admin/v1/version

L'API de version renvoie les informations suivantes :

```

[
  {
    "Scm-tags": "",
    "Scm-commit-id": "e2eb99d93a74409b3ebc5224e596953e9b8a178f",
    "Scm-commit-id-abbrev": "e2eb99d",
    "Maven-version": "0.13.0-SNAPSHOT",
    "Scm-dirty": "false",
    "Scm-commit-time": "2017-01-11T16:38:14+01",
    "Maven-build-timestamp": "2017-01-11T16:06:09Z",
    "Scm-branch": "origin/master_iteration_13",
    "Build-Jdk": "1.8.0_111",
    "Maven-artefactId": "logbook-rest",
    "Maven-groupId": "fr.gouv.vitam"
  },
  {
    "Scm-tags": "",
    "Scm-commit-id": "e2eb99d93a74409b3ebc5224e596953e9b8a178f",
    "Scm-commit-id-abbrev": "e2eb99d",
    "Maven-version": "0.13.0-SNAPSHOT",
    "Scm-dirty": "false",
    "Scm-commit-time": "2017-01-11T16:38:14+01",
    "Maven-build-timestamp": "2017-01-11T16:06:09Z",
    "Scm-branch": "origin/master_iteration_13",
    "Build-Jdk": "1.8.0_111",
    "Maven-artefactId": "logbook-administration",
    "Maven-groupId": "fr.gouv.vitam"
  },
  ...
  ...
  ...
]

```

Signification des champs :

- Scm-tags : en cours de définition
- Scm-commit-id : hash de commit git à partir duquel le composant a été buildé
- Scm-commit-id-abbrev : hash de commit abrégé
- Maven-version : Version indiquée à maven dans le fichier pom.xml
- Scm-dirty : Etat du repo git au moment du build (si présence de fichiers unstaged => dirty)
- Scm-commit-time : Date du commit git
- Maven-build-timestamp : Date du build par maven
- Scm-branch : Nom de la branche git à partir de laquelle le composant a été buildé
- Build-Jdk : Version de la jdk ayant servi à builder le composant
- Maven-artefactId : Nom du composant
- Maven-groupId : namespace du composant

6.3.1.3 /admin/v1/autotest

L'API d'autotest renvoie les informations suivantes :

```
{
  "httpCode":200,
  "code":"000000",
  "context":"logbook",
  "state":"OK",
  "message":"All services are available",
  "description":"All services are available",
  "errors": [
    {
      "httpCode":200,
      "code":"1",
      "context":"LogbookMongoDbAccessImpl",
      "state":"OK",
      "message":"Sub service is available",
      "description":"LogbookMongoDbAccessImpl service is available"
    },
    {
      "httpCode":200,
      "code":"2",
      "context":"logbook",
      "state":"OK",
      "message":"Internal service is available",
      "description":"vitam-iaas-app-01 service is available"
    }
  ]
}
```

Signification des champs :

- **httpCode** : code de retour http
- **code** : en cours de définition ; futur code retour interne VITAM
- **context** : Nom du composant
- **state** : Etat du composant (OK/KO)
- **message** : Message de statut
- **description** : Message de description
- **errors** :

- `httpCode` : code de retour http
- `code` : code de retour
- `context` : nom du composant
- `state` : Etat du composant
- `message` : Message sur l'état du composant
- `description` : Description sur l'état du composant

6.3.2 Patte de service

- `/<composant>/v1/status` : statut simple, renvoyant un statut de fonctionnement incluant des informations techniques sur l'état actuel du composant. Un exemple d'utilisation typique est l'intégration à un outil de supervision ou à un élément actif tiers (ex : load-balancer, ...). L'appel doit être peu coûteux. Le statut normal HTTP renvoyé est 204.

Avertissement : Les composants **vitam-elastic-kibana-interceptor**, **security-internal**, **library** et les *IHM* ne possèdent pas ce statut.

Note : Pour le composant **security-internal**, le point d'*API* est `/v1/status`.

6.4 Logs

La solution logicielle *VITAM* propose une solution ouverte, au choix de l'exploitant. Ce dernier peut, à l'installation, comme à la mise à jour de la solution logicielle *VITAM*, choisir d'utiliser sa propre solution de « regroupement » des logs ou la solution embarquée dans la solution logicielle *VITAM*.

Dans le cas de la solution embarquée, celle-ci se décompose en :

- `rsyslog` ou `syslog-ng` (choix à l'installation, se référer au *DIN*) déployé sur les machines « applicatives » *VITAM* et les envois applicatifs syslog vers un serveur de centralisation de logs (via `facility local0`)
- un serveur de centralisation de logs, comprenant :
 - un mono-noeud (au minimum, ou multi-noeuds) Elasticsearch
 - un moteur logstash, parsant les messages VITAM
 - un afficheur de rendu/aggrégation de données Kibana dédié

Voir aussi :

Les principes & implémentation du système de gestion de logs inclus dans VITAM sont décrits plus en détail dans le DAT.

6.4.1 Changement des règles de log

- Pour les logs fichiers :
 - Définition : fichier `/vitam/conf/<composant>/logback.xml`
 - Format des logs (`encoder`) : ne doit pas être changé ;
 - La sévérité peut être changée ;

- Roulement : le roulement des fichiers défini par défaut dépend du temps, avec une taille globale maximale ; il est défini par la politique `TimeBasedRollingPolicy` de l'appendeur `RollingFileAppender`¹², avec les paramètres suivants :
 - Nombre total de fichiers conservés : 30 (paramètre `maxHistory`);
 - Taille totale des fichiers de logs : 5 Go (paramètre `totalSizeCap`);
 - Pattern des fichiers : dans le répertoire de logs de l'application : `<service_id>.%d.log` (%d étant remplacé par `yyyy-MM-dd`) (paramètre `fileNamePattern`).
- Pour les logs syslog :
 - Format des logs (`suffixPattern`) : ne doit pas être changé;
 - La sévérité peut être changée;
 - Les stacktraces sont exclues de l'envoi à la centralisation des logs (paramètre `throwableExcluded` placé à `false`); ce paramètre ne doit pas être changé.
- Pour les logs du garbage collector :
 - Niveau de détail : activation des détails et des timestamps (paramètres `JVM -XX:+PrintGCDetails -XX:+PrintGCApplicationStoppedTime`)
 - Roulement : le roulement des fichiers dépend de la taille des fichiers, avec un nombre de fichiers maximal ; il est défini comme suit :
 - Activation du roulement : (paramètre `JVM -XX:+UseGCLogFileRotation`)
 - Nombre total de fichiers conservés : 10 (paramètre `JVM -XX:NumberOfGCLogFiles=10`)
 - Taille unitaire maximale d'un" fichiers de logs : 10 Mo (paramètre `JVM -XX:GCLogFileSize=10M`)
 - Pattern des fichiers : dans le répertoire de logs de l'application (paramètre `-Xloggc:$LOG_FOLDER/gc.log`) pour le fichier courant; après roulement, les fichiers sont nommés `gc.log.<n>``` (avec ```<n>` le numéro du fichier, sur base 0).
- Pour les logs accès :
 - Définition : fichier `/vitam/conf/<service_id>/logback-access.xml`
 - Format des logs (`encoder`) : ne doit pas être changé;
 - Roulement : le roulement des fichiers défini par défaut dépend du temps, avec une taille globale maximale ; il est défini par la politique `TimeBasedRollingPolicy` de l'appendeur `RollingFileAppender`¹³, avec les paramètres suivants :
 - Nombre total de fichiers conservés : 7 (paramètre `maxHistory`);
 - Taille totale des fichiers de logs : 14 Go (paramètre `totalSizeCap`);
 - Pattern des fichiers : dans le répertoire de logs de l'application : `accesslog-<service_id>.%d.log` (%d étant remplacé par `yyyy-MM-dd`) (paramètre `fileNamePattern`).

Prudence : La configuration de la durée de rétention des logs accès et/ou leur externalisation devra être ajustée pour respecter les contraintes légales en vigueur pour le système déployé.

6.4.2 Rétention des index sous elasticsearch-log

Curator est l'outil défini dans *VITAM* pour nettoyer les index du *cluster* elasticsearch de log. Curator a été paramétré avec les informations contenues, durant l'installation, dans le fichier `cots_vars.yml`.

Pour les différents index dans le *cluster* Elasticsearch de log, deux paramètres sont définis pour Curator :

<http://logback.qos.ch/manual/appenders.html#RollingFileAppender>
<http://logback.qos.ch/manual/appenders.html#RollingFileAppender>

- durée de fermeture : nombre de jours avant clôture de l'index
- durée de suppression : nombre de jours avant suppression de l'index.

Note : concernant les index « logstash-* », il est recommandé de laisser une durée de rétention de 1 an.

Il est possible de modifier le comportement de curator. Pour ce faire, il faut :

1. modifier le fichier `cots_vars.yml`
2. rejouer le playbook de déploiement, en ajoutant en fin de commande `--tags curator_logs`.

6.5 Audit

Divers audits mis à disposition des utilisateurs et administrateurs par le biais de l'*IHM* de démonstration sont décrits dans le Manuel Utilisateurs.

6.6 Gestion de la capacité

La gestion de la scalabilité du système dépend de ses usages métier ; le lien entre les usages et les composants *VITAM* sollicités est indiqué dans le *DAT*, avec des dimensionnements de plateforme standard pour différents usages.

Le suivi de la charge sur chaque serveur se fait par les outils standard de l'exploitant.

6.7 Suivi de l'état de sécurité

Une étude est actuellement en cours pour réaliser ce type de suivi.

6.8 Alerting

6.8.1 Système

Le suivi des alertes système est à charge de l'exploitant.

6.8.2 Applicatif

Les logs applicatifs de la solution *VITAM* permettent à l'exploitant de mettre en place un *alerting* adapté à l'usage de son équipe métier et technique. Par défaut, et en guise d'exemple, des dashboards Kibana sont disponibles avec un rassemblement des événements courants de sécurité / erreur (ex. : incohérence règles de gestion, désynchronisation MongoDB / Elasticsearch...).

6.9 Suivi des Workflows

La solution logicielle *VITAM* intègre une solution de suivi et de gestion des *Workflows*. Elle permet entre autres de :

- Relancer un Workflow arrêté
- Mettre en pause un Workflow démarré

- Rejouer une étape d'un Workflow
- Annuler un workflow

6.9.1 Suivi

Le suivi peut être réalisé via *IHM* ou par des appels *REST*.

6.9.1.1 IHM

Il existe une page dans l'*IHM* de démonstration, permettant d'influer sur les processus en cours. Tous les processus mis en pause, automatiquement (lors d'un FATAL) ou bien manuellement (Mode pas à pas) apparaissent sur cette *IHM*. Il est également possible, à partir de cette *IHM*, de relancer le processus ou bien de rejouer une étape, après action d'exploitation.

6.9.1.2 Appels REST

Il est tout aussi possible d'exécuter ces différentes actions sur l'*API* en direct, via des appels `curl` par exemple sur le composant access external :

- PUT sur le endpoint /operations/GUID avec comme header X-Action :RESUME par exemple.

Pour plus d'information, consulter la documentation des *API* externes.

6.9.2 Cas des *workflows* en FATAL

Un workflow se met en pause dès qu'il se retrouve en statut **FATAL**. Plusieurs causes peuvent expliquer un tel état.

6.9.2.1 Plugins et Handlers

Plusieurs problèmes peuvent expliquer qu'un *Handler* ou un *plugin* retourne une erreur « FATAL » et donc provoque la mise en pause du *Workflow*.

Si le composant workspace est défectueux ou ne répond plus, alors un FATAL pourra être obtenu pour tous les *Handlers* et *plugins*.

Si le composant logbook est défectueux ou ne répond plus, alors un FATAL pourra être obtenu pour les *handlers* suivants :

- CommitLifecycleActionHandler
- CommitLifecycleObjectGroupActionHandler
- CommitLifecycleUnitActionHandler
- ListLifecycleTraceabilityActionHandler
- FinalizeLifecycleTraceabilityActionHandler
- RollBackActionHandler

Si le composant functional-administration est défectueux ou ne répond plus, alors un FATAL pourra être obtenu pour les *Handlers* suivants :

- CheckArchiveProfileRelationActionHandler
- CheckArchiveProfileActionHandler
- GenerateAuditReportActionHandler
- PrepareAuditActionHandler

Si le composant metadata est défectueux ou ne répond plus, alors un FATAL pourra être obtenu pour les Handlers suivants :

- AccessionRegisterActionHandler
- ListArchiveUnitsActionHandler
- PrepareAuditActionHandler
- ArchiveUnitRulesUpdateActionPlugin
- AuditCheckObjectPlugin
- IndexObjectGroupActionPlugin
- IndexUnitActionPlugin
- RunningIngestsUpdateActionPlugin

Si le composant storage est défectueux ou ne répond plus, alors un FATAL pourra être obtenu pour les Handlers suivants :

- CheckStorageAvailabilityActionHandler
- FinalizeLifecycleTraceabilityActionHandler
- GenerateAuditReportActionHandler
- PrepareTraceabilityCheckProcessActionHandler
- PutBinaryOnWorkspace
- CheckIntegrityObjectPlugin
- CheckExistenceObjectPlugin
- StoreMetaDataSetObjectGroupActionPlugin
- StoreMetaDataSetUnitActionPlugin
- StoreObjectActionHandler
- StoreObjectGroupActionPlugin

Si le composant processing est défectueux ou ne répond plus, alors un FATAL pourra être obtenu pour les Handlers suivants :

- ListRunningIngestsActionHandler

Si le composant FormatIdentifier est défectueux et ne répond plus, alors un FATAL pourra être obtenu pour le Handler suivant :

- FormatIdentificationActionPlugin

6.9.2.2 Distributor

Plusieurs cas peuvent provoquer un FATAL au niveau du processing :

- si metadata ou workspace est injoignable
- si un *handler* (ou plugin) inexistant est appelé.
- si le distributeur tente d'appeler une famille de worker inexistante

6.9.2.3 Processing - State Machine

Dans le cas où le Processing ne parvient pas à enregistrer l'état du workflow sur le workspace, un FATAL est provoqué. Il en va de même si le composant logbook est défectueux.

6.9.3 Redémarrer un processus en cas de pause

6.9.3.1 Trouver la cause

De manière générale, il convient d'identifier le composant (ou les composants) posant problème. Il s'agira majoritairement de mtadata, de logbook, du etorage ou encore du workspace.

A partir du Guid de l'opération mise en pause, il est facilement possible de voir, dans les logs du processing ou des workers quels sont les composants incriminés.

6.9.3.2 Relancer le Workflow

A partir du Guid de l'opération mise en pause et une fois le composant redémarré, il est possible de relancer le workflow.

6.9.3.2.1 Vérifier les inputs

S'assurer à partir du GUID de l'opération que l'on nommera X la présence :

- d'un fichier X.json dans /vitam/data/workspace/process/distributorIndex/
- d'un répertoire X dans /vitam/data/workspace/ contenant à minima une liste de sous-répertoires (et notamment le *SIP* décompressé dans le sous répertoire SIP).

6.9.3.2.2 Rejouer une étape

Depuis l'*IHM*, relancer l'étape précédente en cliquant sur l'icône « Replay ». Via les *API*, il suffit de lancer un appel `curl` sur le composant access external : PUT sur le endpoint /operations/GUID avec comme header X-Action :REPLAY.

Cette action aura pour résultat d'exécuter une deuxième fois l'étape qui a échoué. En sortie de ce replay, le statut du workflow doit passer à OK et l'état à PAUSE.

6.9.3.2.3 Prochaine étape

Depuis l'*IHM*, exécuter l'étape suivante en cliquant sur l'icône « Next ». Via les *API*, il suffit de lancer un appel `curl` sur le composant « access-external » : PUT sur le endpoint /operations/GUID avec comme header X-Action :NEXT.

Cette action aura pour résultat d'exécuter l'étape suivante. En sortie de ce replay, le statut du workflow doit passer à OK et l'état à PAUSE.

6.9.3.2.4 Finaliser le workflow

Il est possible de poursuivre le workflow jusqu'à son terme.

Depuis l'*IHM*, finaliser le workflow en cliquant sur l'icône « Fast Forward ».

Via les *API*, il suffit de lancer un appel `curl` sur le composant access-external : PUT sur le endpoint /operations/GUID avec comme header X-Action :RESUME.

6.10 Cohérence des journaux

Il existe un outil d'administration utilisable par l'exploitant afin de réaliser un test de cohérence des journaux. Cet outil permet de vérifier que les données enregistrées dans la collection LogbookOperations sont bien en cohérence avec les informations sauvegardées dans les collections LFC.

Actuellement, seuls les *TNR* utilisent le point d'API.

A l'avenir, il sera possible de préciser les modalités dans un fichier json associé, et il sera possible d'utiliser le contrôle de cohérence indépendamment.

6.10.1 Lancement

Pour lancer l'outil de cohérence, il suffit de lancer une requête (*curl*, par exemple) sur le serveur logbook interne (sur la « patte » d'administration) :

- POST sur le endpoint /checklogbook

6.10.2 Résultat

L'outil de cohérence renvoie un code OK, si l'opération s'est bien déroulée. En cas d'erreur interne, un code HTTP 500 sera renvoyé.

Dans le cadre d'un OK, un rapport au format Json sera généré, et sera enregistré sur les offres de stockage.

Le rapport contiendra les informations suivantes :

- checkedEvents : la liste des événements vérifiés.
- checkErrors : la liste des erreurs constatées.

6.11 Liste des *timers* systemd

Note : Dans les sections suivantes, les éléments de type `<curator.log.metrics.close>` correspondent à des variables de l'inventaire ansible utilisé.

6.11.1 Timers de maintenance des index elasticsearch-log

Ces *timers* gèrent la maintenance des index elasticsearch du cluster elasticsearch-log.

Ces *timers* sont activés sur tous les sites d'un déploiement multi-sites.

6.11.1.1 vitam-curator-metrics-indexes

Maintenance des indexes `metrics-vitam-*` (sur elasticsearch-log) (qui contiennent les métriques remontées par les composants VITAM) :

- Ferme les indexes de plus de `<curator.log.metrics.close>` jours ;
- Supprime les indexes de plus de `<curator.log.metrics.delete>` jours.

Units systemd :

- vitam-curator-metrics-indexes.service

- vitam-curator-metrics-indexes.timer

Exécution :

- Localisation : groupe ansible [hosts-elasticsearch-log] (sur toutes les instances du groupe)
- Périodicité : Lancé chaque jour à 00 :30.

6.11.1.2 vitam-curator-close-old-indexes

Fermeture des anciens indexes `logstash-*` (sur `elasticsearch-log`) de plus de `<curator.log.logstash.close>` jours (ces indexes contiennent les logs remontés par les composants et COTS VITAM).

Units systemd :

- vitam-curator-close-old-indexes.service
- vitam-curator-close-old-indexes.timer

Exécution :

- Localisation : groupe ansible [hosts-elasticsearch-log] (sur toutes les instances du groupe)
- Périodicité : Lancé chaque jour à 00 :10.

6.11.1.3 vitam-curator-delete-old-indexes

Suppression des indexes `logstash-*` (sur `elasticsearch-log`) de plus de `<curator.log.logstash.delete>` jours (ces indexes contiennent les logs remontés par les composants et COTS VITAM).

Units systemd :

- vitam-curator-delete-old-indexes.service
- vitam-curator-delete-old-indexes.timer

Exécution :

- Localisation : groupe ansible [hosts-elasticsearch-log] (sur toutes les instances du groupe)
- Périodicité : Lancé chaque jour à 00 :20.

6.11.2 Timers de gestion des journaux (preuve systématique)

Ces *timers* gèrent la sécurisation des journaux métier *VITAM*.

Ces *timers* sont activés uniquement sur le site primaire d'un déploiement multi-sites.

6.11.2.1 vitam-storage-log-backup

Backup des journaux d'écriture de storage dans les offres de stockage.

Units systemd :

- vitam-storage-log-backup.service
- vitam-storage-log-backup.timer

Exécution :

- Localisation : groupe ansible [hosts-storage-engine] (sur toutes les instances du groupe)
- Périodicité : Lancé toutes les heures à 0 minutes 0 secondes (donc : 0h00, 1h00, ...)

6.11.2.2 vitam-storage-accesslog-backup

Backup des journaux d'accès de storage dans les offres de stockage.

Units systemd :

- vitam-storage-accesslog-backup.service
- vitam-storage-accesslog-backup.timer

Exécution :

- Localisation : groupe ansible [hosts-storage-engine] (sur toutes les instances du groupe)
- Périodicité : Lancé toutes les heures à 0 minutes 0 secondes (donc : 0h00, 1h00, ...)

6.11.2.3 vitam-storage-log-traceability

Sécurisation des journaux d'écriture de storage.

Units systemd :

- vitam-storage-log-traceability.service
- vitam-storage-log-traceability.timer

Exécution :

- Localisation : groupe ansible [hosts-storage-engine] (sur la dernière instance du groupe uniquement)
- Périodicité : Lancé toutes les heures à 10 minutes 0 secondes (donc : 0h10, 1h10, ...)

6.11.2.4 vitam-traceability-operations

Sécurisation du journal des opérations.

Units systemd :

- vitam-traceability-operations.service
- vitam-traceability-operations.timer

Exécution :

- Localisation : groupe ansible [hosts-logbook] (sur la dernière instance du groupe uniquement)
- Périodicité : lancé à chaque changement d'heure.

6.11.2.5 vitam-traceability-lfc-unit

Sécurisation du journal du cycle de vie des unités archivistiques.

Units systemd :

- vitam-traceability-lfc-unit.service
- vitam-traceability-lfc-unit.timer

Exécution :

- Localisation : groupe ansible [hosts-logbook] (sur la dernière instance du groupe uniquement)
- Périodicité : lancé à chaque changement d'heure.

6.11.2.6 vitam-traceability-lfc-objectgroup

Sécurisation du journal du cycle de vie des groupes d'objets.

Units systemd :

- vitam-traceability-lfc-objectgroup.service
- vitam-traceability-lfc-objectgroup.timer

Exécution :

- Localisation : groupe ansible [hosts-logbook] (sur la dernière instance du groupe uniquement)
- Périodicité : lancé à chaque changement d'heure.

6.11.3 Timers d'audit interne VITAM

Ces *timers* gèrent le déclenchement périodique des tâches d'audit interne *VITAM*.

Ces *timers* sont activés uniquement sur le site primaire d'un déploiement multi-sites.

6.11.3.1 vitam-traceability-audit

Contrôle de la validité de la sécurisation des journaux.

Units systemd :

- vitam-traceability-audit.service
- vitam-traceability-audit.timer

Exécution :

- Localisation : groupe ansible [hosts-logbook] (sur la dernière instance du groupe uniquement)
- Périodicité : lancé chaque jour à 0 :00.

6.11.3.2 vitam-rule-management-audit

Validation de la cohérence des règles de gestion entre les offres de stockage et les bases de données.

Units systemd :

- vitam-rule-management-audit.service
- vitam-rule-management-audit.timer

Exécution :

- Localisation : groupe ansible [hosts-functional-administration] (sur la dernière instance du groupe uniquement)
- Périodicité : lancé à chaque changement d'heure.

6.11.4 Timer relatif aux liens symboliques de *accession register*

6.11.4.1 vitam-create-accession-register-symbolic

Déclenche une commande qui va calculer le registre des fonds symbolique et les ajoute dans les bases de données.

Units systemd :

- vitam-create-accession-register-symbolic.service (activé sur site primaire uniquement)

- vitam-create-accession-register-symbolic.timer (activé sur site primaire uniquement)

Exécution :

- Localisation : groupe ansible [hosts-functional-administration] (sur la dernière instance du groupe uniquement)
- Périodicité : chaque jour à minuit.

6.11.5 Timers de reconstruction VITAM

Ces timers gèrent la reconstruction des bases de données VITAM à partir des informations persistées dans les offres de stockage.

Ces timers sont activés uniquement sur le site secondaire d'un déploiement multi-sites.

6.11.5.1 vitam-functional-administration-reconstruction

Reconstruction des données portées par le composant functional-administration.

Units systemd :

- vitam-functional-administration-reconstruction.service
- vitam-functional-administration-reconstruction.timer
- vitam-functional-administration-accession-register-reconstruction.service (activé sur site secondaire seulement)
- vitam-functional-administration-accession-register-reconstruction.timer (activé sur site secondaire seulement)

Exécution :

- Localisation : groupe ansible [hosts-functional-administration] (sur la dernière instance du groupe uniquement)
- Périodicité : lancé tous les cinq minutes.

6.11.5.2 vitam-logbook-reconstruction

Reconstruction des données portées par le composant logbook.

Units systemd :

- vitam-logbook-reconstruction.service
- vitam-logbook-reconstruction.timer

Exécution :

- Localisation : groupe ansible [hosts-logbook] (sur la dernière instance du groupe uniquement)
- Périodicité : lancé tous les 5 minutes.

6.11.5.3 vitam-metadata-reconstruction

Reconstruction des données portées par le composant metadata.

Units systemd :

- vitam-metadata-reconstruction.timer
- vitam-metadata-reconstruction.service

Exécution :

- Localisation : groupe ansible [hosts-metadata] (sur la dernière instance du groupe uniquement)
- Périodicité : lancé toutes les 5 minutes.

6.11.5.4 vitam-metadata-store-graph

Log shipping des données graphes portées par le composant metadata.

Units systemd :

- vitam-metadata-store-graph.timer
- vitam-metadata-store-graph.service

Exécution :

- Localisation : groupe ansible [hosts-metadata] (sur la dernière instance du groupe uniquement)
- Périodicité : lancé toutes les 30 minutes.

Exploitation des COTS de la solution logicielle VITAM

7.1 Généralités

Les composants de la solution logicielle *VITAM* sont déployés par un *playbook* ansible qui :

1. déploie, selon l'inventaire employé, les *packages* nécessaires
2. applique la configuration de chaque composant selon son contexte défini dans l'inventaire

Les composants *VITAM* sont décrits ci-après.

Avertissement : En cas de modification de la configuration, redémarrer le service associé.

7.2 COTS

7.2.1 Cerebro

7.2.1.1 Présentation

Cerebro est un utilitaire de supervision de l'état d'un cluster ElasticSearch.

7.2.1.2 Configuration / fichiers utiles

7.2.1.2.1 Fichier `/vitam/conf/cerebro/application.conf`

```
http.port = {{ cerebro.port }}
http.address = {{ ip_admin }}
# Secret will be used to sign session cookies, CSRF tokens and for other encryption_
↳utilities.
```

(suite sur la page suivante)

(suite de la page précédente)

```

# It is highly recommended to change this value before running cerebro in production.
secret = "{{ cerebro.secret_key }}"

# Application base path
basePath = "{{ cerebro.baseuri }}"

# Defaults to RUNNING_PID at the root directory of the app.
# To avoid creating a PID file set this value to /dev/null
pidfile.path = "/dev/null"

# Rest request history max size per user
rest.history.size = 50 // defaults to 50 if not specified

# Path of local database file
data.path = "{{ vitam_defaults.folder.root_path }}/data/cerebro/cerebro.db"

# Authentication
auth = {
    # Example of LDAP authentication
    #type: ldap
    #settings: {
        #url = "ldap://host:port"
        #base-dn = "ou=active,ou=Employee"
        #method = "simple"
        #user-domain = "domain.com"
    #}
{% if cerebro.basicauth is defined %}
    # Simple username/password authentication
    type: basic
    settings: {
        username = "{{ cerebro.basicauth.username }}"
        password = "{{ cerebro.basicauth.password }}"
    }
{% else %}
    # Example of simple username/password authentication
    #type: basic
    #settings: {
        #username = "admin"
        #password = "1234"
    #}
{% endif %}
}

# A list of known hosts
hosts = [
{% if groups['hosts-elasticsearch-log']|length > 0 %}
    {
        host = "http://{{ elasticsearch.log.host }}:{{ elasticsearch.log.port_http }}"
        name = "{{ elasticsearch.log.cluster_name }}"
    },
{% endif %}
{% if groups['hosts-elasticsearch-data']|length > 0 %}
    {
        host = "http://{{ elasticsearch.data.host }}:{{ elasticsearch.data.port_http }}"
        name = "{{ elasticsearch.data.cluster_name }}"
    },
{% endif %}
]

```

(suite sur la page suivante)

(suite de la page précédente)

```
# {
#   host = "http://localhost:9200"
#   name = "Some Cluster"
# },
# Example of host with authentication
# {
#   host = "http://some-authenticated-host:9200"
#   name = "Secured Cluster"
#   auth = {
#     username = "username"
#     password = "secret-password"
#   }
# }
]
```

7.2.1.3 Opérations

- Démarrage du service

Les commandes suivantes sont à passer sur les différentes machines hébergeant le composant vitam-elasticsearch-cerebro.

En tant qu'utilisateur root : `systemctl start vitam-elasticsearch-cerebro`

- Arrêt du service

Les commandes suivantes sont à passer sur les différentes machines constituant le composant vitam-elasticsearch-cerebro.

En tant qu'utilisateur root : `systemctl stop vitam-elasticsearch-cerebro`

- Sauvegarde du service

N/A

- Supervision du service

Contrôler le retour HTTP 200 sur l'URL <protocole web https ou https>://<host>:9000/cerebro

- Exports

N/A

- gestion de la capacité

N/A

- actions récurrentes

N/A

- cas des batches

N/A

7.2.2 consul

7.2.2.1 Présentation

Consul est un DNS applicatif.

7.2.2.1.1 Cas serveur

Le serveur Consul fédère les agents dans leurs requêtes « DNS-like » et permet de rebondir sur un DNS externe, s'il ne permet pas de lui-même, de faire la résolution.

7.2.2.1.2 Cas agent

L'agent Consul annonce aux serveurs les services qu'il permet de porter et *checke* régulièrement l'état de ces services.

7.2.2.2 Configuration / fichiers utiles

Les fichiers de configuration sont gérés par les procédures d'installation ou de mise à niveau de l'environnement *VITAM*. Se référer au *DIN*.

7.2.2.2.1 Cas des applicatifs monitorés par Consul

Pour chaque composant *VITAM* nécessitant une supervision de la part de Consul, un fichier est installé sur l'agent de la machine sous `vitam/conf/consul` et est basé sur ce squelette :

7.2.2.2.1.1 Fichier `/vitam/conf/consul/service-<composant>.json`

```

1 {
2   "service": {
3     {% if vitam_struct.vitam_component == vitam.storageofferdefault.vitam_component %}
4       "name": "{{ offer_conf }}",
5     {% else %}
6       "name": "{{ vitam_struct.vitam_component }}",
7     {% endif %}
8     "address": "{{ ip_service }}",
9     {% if ip_wan is defined %}
10      "advertise_addr_wan": "{{ ip_wan }}",
11    {% endif %}
12    "port": {{ vitam_struct.port_service }},
13    "enable_tag_override": false,
14    "tags": ["vitam", "{{ vitam_struct.vitam_component }}"],
15    "checks": [
16      {
17        "name": "{{ vitam_struct.vitam_component }}: business service check",
18      {% if vitam_struct.https_enabled==true %}
19        "notes": "HTTPS port opened",
20        "tcp": "{{ ip_service }}:{{ vitam_struct.port_service }}",
21      {% else %}
22        "notes": "HTTP port opened",
23        "tcp": "{{ ip_service }}:{{ vitam_struct.port_service }}",
24      {% endif %}
25      "interval": "1s"
26    },
27    {
28      "name": "{{ vitam_struct.vitam_component }} : admin service check",
29      "notes": "Status admin : /admin/v1/status",
30      "http": "http://{{ ip_admin }}:{{ vitam_struct.port_admin }}/admin/v1/status",

```

(suite sur la page suivante)

(suite de la page précédente)

```

31     "interval": "1s"
32   }
33   {% if (vitam_struct.https_enabled != true) and (vitam_struct.vitam_component !=
↳ vitam.elastickibanainterceptor.vitam_component) and (vitam_struct.vitam_component !=
↳ vitam.security_internal.vitam_component) and (vitam_struct.vitam_component !=
↳ vitam.ihm_demo.vitam_component) and (vitam_struct.vitam_component != vitam.ihm_
↳ recette.vitam_component) and (vitam_struct.vitam_component != vitam.library.vitam_
↳ component) %}
34     , {
35       "name": "{{ vitam_struct.vitam_component }} : http business service check",
36       "notes": "Status business : /{{ vitam_struct.baseuri }}/v1/status",
37       "http": "http://{{ ip_service }}:{{ vitam_struct.port_service }}/{{ vitam_
↳ struct.baseuri }}/v1/status",
38       "interval": "1s"
39     }
40   {% endif %}
41   {% if (vitam_struct.vitam_component == vitam.security_internal.vitam_
↳ component) %}
42     , {
43       "name": "{{ vitam_struct.vitam_component }} : http business service check",
44       "notes": "Status business : /status",
45       "http": "http://{{ ip_service }}:{{ vitam_struct.port_service }}/status",
46       "interval": "1s"
47     }
48   {% endif %}
49   {% if (vitam_struct.vitam_component == vitam.worker.vitam_component) or (vitam_
↳ struct.vitam_component == vitam.ingestexternal.vitam_component) %}
50     , {
51       "name": "Siegfried check",
52       "notes": "Is siegfried running ?",
53       "tcp": "localhost:{{ siegfried.port }}",
54       "interval": "1s"
55     }
56   {% endif %}
57   {% if vitam_struct.antivirus is defined %}
58     , {
59       "name": "Antivirus check",
60       "notes": "Is {{ vitam_struct.antivirus }} running ?",
61       "args": ["{{ vitam_folder_conf }}/scan-{{ vitam_struct.antivirus }}.sh", "{{
↳ vitam_folder_conf }}/scan-{{ vitam_struct.antivirus }}.sh"],
62       "interval": "30s",
63       "timeout": "5s"
64     }
65   {% endif %}
66   ]
67 }
68 }

```

7.2.2.3 Opérations

- Démarrage du service

En tant qu'utilisateur root : `systemctl start vitam-consul`

- Arrêt du service

En tant qu'utilisateur root : `systemctl stop vitam-consul`

Avertissement : en cas de redémarrage du cluster serveur consul, il faut procéder à un arrêt/relance par serveur avant de passer au suivant.

- Sauvegarde du service

Ce service ne nécessite pas de sauvegarde particulière.

- Logs

Les logs applicatifs sont envoyés par rsyslog à la solution de centralisation des logs ; il est néanmoins possible d'en virsionner une représentation par la commande :

```
journalctl --unit vitam-consul
```

- Supervision du service

Consul possède une IHM permettant de superviser l'ensemble des services qu'il couvre.

http(s) ://<adresse> :<port>/ui

- Exports

N/A

- gestion de la capacité

N/A

- actions récurrentes

- cas des batches

N/A

7.2.3 Kibana interceptor

7.2.3.1 Présentation

Le composant est une interface d'accès entre kibana « métier » et le *cluster* Elasticsearch de données métier.

Prudence : Ce composant **N'EST PAS** à installer en environnement de production.

7.2.3.2 Configuration / fichiers utiles

Les fichiers de configuration sont définis sous /vitam/conf/elastic-kibana-interceptor.

7.2.3.2.1 Fichier elastic-kibana-interceptor.conf

```
jettyConfig: jetty-config.xml

clusterName: {{ vitam_struct.cluster_name }}
elasticsearchNodes:
{% for server in groups['hosts-elasticsearch-data'] %}
- hostName: {{ hostvars[server]['ip_service'] }}
  tcpPort: {{ elasticsearch.data.port_http }}
{% endfor %}
whitelist : ["tenant", "all", "mgt", "min", "max", "nbc", "og", "ops", "opi", "sp",
↪ "sps", "uds", "up", "us", "storage", "unitType", "v", "qualifiers"]
```

7.2.3.3 Opérations

- Démarrage du service

Les commandes suivantes sont à passer sur les différentes machines constituant le *cluster* Elasticsearch de données.

En tant qu'utilisateur root : `systemctl start vitam-elastic-kibana-interceptor`

- Arrêt du service

Les commandes suivantes sont à passer sur les différentes machines constituant le *cluster* Elasticsearch de données.

En tant qu'utilisateur root : `systemctl stop vitam-elastic-kibana-interceptor`

- Sauvegarde du service

N/A

- Supervision du service

Contrôler le retour HTTP 200 sur l'URL <protocole web https ou https>://<host>:<port>/

- Exports

N/A

- gestion de la capacité

N/A

- actions récurrentes

- cas des batches

N/A

- Modification de la liste blanche

Modifier dans le fichier `/vitam/conf/elastic-kibana-interceptor/elastic-kibana-interceptor.conf` le contenu de la directive `whitelist`.

A l'issue, redémarrer le composant.

7.2.4 elasticsearch chaîne de log

7.2.4.1 Présentation

Le composant `vitam-elasticsearch-log` est une instance de la base d'indexation `elasticsearch` stockant les informations suivantes :

- les logs des applications VITAM ;
- les logs des applications du sous-système de centralisation des logs ;
- les métriques applicatives.

7.2.4.2 Configuration / fichiers utiles

Se reporter au *DIN*, qui configure le *cluster* ElasticSearch de la chaîne de log.

Les fichiers de configuration sont définis sous `/vitam/conf/elasticsearch-log`.

7.2.4.2.1 Fichier `/vitam/conf/elasticsearch-log/log4j2.properties`

```

status = error

# log action execution errors for easier debugging
logger.action.name = org.elasticsearch.action
logger.action.level = debug

appender.console.type = Console
appender.console.name = console
appender.console.layout.type = PatternLayout
appender.console.layout.pattern = [%d{ISO8601}][%-5p][%-25c{1.}] %marker%m%n

appender.syslog.type = Syslog
appender.syslog.name = syslog
appender.syslog.appName = {{ composant.cluster_name }}
appender.syslog.facility = {{ vitam_defaults.syslog_facility }}
appender.syslog.host = {{ inventory_hostname }}
appender.syslog.protocol = UDP
appender.syslog.port = 514
appender.syslog.layout.type = PatternLayout
# Note: rsyslog only parse RFC3195-formatted syslog messages by default ; AND, to_
↳make it work with log4j2, we need to start the layout by the app-name.
# IF we were in 5424, we wouldn't have to do this.
appender.syslog.layout.pattern = {{ composant.cluster_name }}: [%d{ISO8601}][%-5p][%-
↳25c{1.}] %marker%m%n
# appender.syslog.format = RFC5424
# appender.syslog.mdcId = esdata

appender.rolling.type = RollingFile
appender.rolling.name = rolling
appender.rolling.fileName = ${sys:es.logs.base_path}${sys:file.separator}${sys:es.
↳logs.cluster_name}.log
appender.rolling.layout.type = PatternLayout
appender.rolling.layout.pattern = [%d{ISO8601}][%-5p][%-25c{1.}] %marker%. -10000m%n
appender.rolling.filePattern = ${sys:es.logs.base_path}${sys:file.separator}${sys:es.
↳logs.cluster_name}-%d{yyyy-MM-dd}.log
appender.rolling.policies.type = Policies
appender.rolling.policies.time.type = TimeBasedTriggeringPolicy
appender.rolling.policies.time.interval = 1
appender.rolling.policies.time.modulate = true

rootLogger.level = info
rootLogger.appenderRef.console.ref = console
rootLogger.appenderRef.rolling.ref = rolling
rootLogger.appenderRef.syslog.ref = syslog

appender.deprecation_rolling.type = RollingFile
appender.deprecation_rolling.name = deprecation_rolling
appender.deprecation_rolling.fileName = ${sys:es.logs.base_path}${sys:file.separator}${
↳sys:es.logs.cluster_name}_deprecation.log
appender.deprecation_rolling.layout.type = PatternLayout
appender.deprecation_rolling.layout.pattern = [%d{ISO8601}][%-5p][%-25c{1.}] %marker%.
↳-10000m%n
appender.deprecation_rolling.filePattern = ${sys:es.logs.base_path}${sys:file.
↳separator}${sys:es.logs.cluster_name}_deprecation-%i.log.gz
appender.deprecation_rolling.policies.type = Policies
appender.deprecation_rolling.policies.size.type = SizeBasedTriggeringPolicy
appender.deprecation_rolling.policies.size.size = 1GB

```

(suite sur la page suivante)

(suite de la page précédente)

```

appender.deprecation_rolling.strategy.type = DefaultRolloverStrategy
appender.deprecation_rolling.strategy.max = 4

logger.deprecation.name = org.elasticsearch.deprecation
logger.deprecation.level = warn
logger.deprecation.appenderRef.deprecation_rolling.ref = deprecation_rolling
logger.deprecation.additivity = false

appender.index_search_slowlog_rolling.type = RollingFile
appender.index_search_slowlog_rolling.name = index_search_slowlog_rolling
appender.index_search_slowlog_rolling.fileName = ${sys:es.logs.base_path}${sys:file.
↳separator}${sys:es.logs.cluster_name}_index_search_slowlog.log
appender.index_search_slowlog_rolling.layout.type = PatternLayout
appender.index_search_slowlog_rolling.layout.pattern = [%d{ISO8601}][%-5p][%-25c]
↳%marker%. -10000m%n
appender.index_search_slowlog_rolling.filePattern = ${sys:es.logs.base_path}${
↳{sys:file.separator}${sys:es.logs.cluster_name}_index_search_slowlog-%d{yyyy-MM-dd}.
↳log
appender.index_search_slowlog_rolling.policies.type = Policies
appender.index_search_slowlog_rolling.policies.time.type = TimeBasedTriggeringPolicy
appender.index_search_slowlog_rolling.policies.time.interval = 1
appender.index_search_slowlog_rolling.policies.time.modulate = true

logger.index_search_slowlog_rolling.name = index.search.slowlog
logger.index_search_slowlog_rolling.level = {{ composant.index_search_slowlog_rolling_
↳level }}
logger.index_search_slowlog_rolling.appenderRef.index_search_slowlog_rolling.ref = _
↳index_search_slowlog_rolling
logger.index_search_slowlog_rolling.additivity = false

appender.index_indexing_slowlog_rolling.type = RollingFile
appender.index_indexing_slowlog_rolling.name = index_indexing_slowlog_rolling
appender.index_indexing_slowlog_rolling.fileName = ${sys:es.logs.base_path}${sys:file.
↳separator}${sys:es.logs.cluster_name}_index_indexing_slowlog.log
appender.index_indexing_slowlog_rolling.layout.type = PatternLayout
appender.index_indexing_slowlog_rolling.layout.pattern = [%d{ISO8601}][%-5p][%-25c]
↳%marker%. -10000m%n
appender.index_indexing_slowlog_rolling.filePattern = ${sys:es.logs.base_path}${
↳{sys:file.separator}${sys:es.logs.cluster_name}_index_indexing_slowlog-%d{yyyy-MM-
↳dd}.log
appender.index_indexing_slowlog_rolling.policies.type = Policies
appender.index_indexing_slowlog_rolling.policies.time.type = TimeBasedTriggeringPolicy
appender.index_indexing_slowlog_rolling.policies.time.interval = 1
appender.index_indexing_slowlog_rolling.policies.time.modulate = true

logger.index_indexing_slowlog.name = index.indexing.slowlog.index
logger.index_indexing_slowlog.level = {{ composant.index_indexing_slowlog_level }}
logger.index_indexing_slowlog.appenderRef.index_indexing_slowlog_rolling.ref = index_
↳indexing_slowlog_rolling
logger.index_indexing_slowlog.additivity = false

```

7.2.4.2.2 Fichier /vitam/conf/elasticsearch-log/jvm.options

```
## JVM configuration
```

(suite sur la page suivante)

(suite de la page précédente)

```
#####
## IMPORTANT: JVM heap size
#####
##
## You should always set the min and max JVM heap
## size to the same value. For example, to set
## the heap to 4 GB, set:
##
## -Xms4g
## -Xmx4g
##
## See https://www.elastic.co/guide/en/elasticsearch/reference/current/heap-size.html
## for more information
##
#####

# Xms represents the initial size of total heap space
# Xmx represents the maximum size of total heap space

-Xms{{ elasticsearch_memory }}
-Xmx{{ elasticsearch_memory }}

#####
## Expert settings
#####
##
## All settings below this section are considered
## expert settings. Don't tamper with them unless
## you understand what you are doing
##
#####

## GC configuration
-XX:+UseConcMarkSweepGC
-XX:CMSInitiatingOccupancyFraction=75
-XX:+UseCMSInitiatingOccupancyOnly

## optimizations

# pre-touch memory pages used by the JVM during initialization
-XX:+AlwaysPreTouch

## basic

# force the server VM (remove on 32-bit client JVMs)
-server

# explicitly set the stack size (reduce to 320k on 32-bit client JVMs)
-Xss1m

# set to headless, just in case
-Djava.awt.headless=true

# ensure UTF-8 encoding by default (e.g. filenames)
-Dfile.encoding=UTF-8
```

(suite sur la page suivante)

(suite de la page précédente)

```

# use our provided JNA always versus the system one
-Djna.nosys=true

# use old-style file permissions on JDK9
-Djdk.io.permissionsUseCanonicalPath=true

# flags to configure Netty
-Dio.netty.noUnsafe=true
-Dio.netty.noKeySetOptimization=true
-Dio.netty.recycler.maxCapacityPerThread=0

# log4j 2
-Dlog4j.shutdownHookEnabled=false
-Dlog4j2.disable.jmx=true
-Dlog4j.skipJansi=true

## heap dumps

# generate a heap dump when an allocation from the Java heap fails
# heap dumps are created in the working directory of the JVM
-XX:+HeapDumpOnOutOfMemoryError

# specify an alternative path for heap dumps
# ensure the directory exists and has sufficient space
-XX:HeapDumpPath={{ elasticsearch_log_dir }}

## GC logging

-XX:+UseGCLogFileRotation
-XX:NumberOfGCLogFiles=10
-XX:GCLogFileSize=10M
-XX:+PrintGCDetails
-XX:+PrintGCApplicationStoppedTime

#-XX:+PrintGCDetails
#-XX:+PrintGCTimeStamps
#-XX:+PrintGCDateStamps
#-XX:+PrintClassHistogram
#-XX:+PrintTenuringDistribution
#-XX:+PrintGCApplicationStoppedTime

# log GC status to a file with time stamps
# ensure the directory exists
-Xloggc:${loggc}

# By default, the GC log file will not rotate.
# By uncommenting the lines below, the GC log file
# will be rotated every 128MB at most 32 times.
#-XX:+UseGCLogFileRotation
#-XX:NumberOfGCLogFiles=32
#-XX:GCLogFileSize=128M

# Elasticsearch 5.0.0 will throw an exception on unquoted field names in JSON.
# If documents were already indexed with unquoted fields in a previous version
# of Elasticsearch, some operations may throw errors.
#
# WARNING: This option will be removed in Elasticsearch 6.0.0 and is provided

```

(suite sur la page suivante)

(suite de la page précédente)

```
# only for migration purposes.
#-Delasticsearch.json.allow_unquoted_field_names=true
```

7.2.4.2.3 Fichier /vitam/conf/elasticsearch-log/elasticsearch.yml

```
# ===== Elasticsearch Configuration =====
#
# NOTE: Elasticsearch comes with reasonable defaults for most settings.
#       Before you set out to tweak and tune the configuration, make sure you
#       understand what are you trying to accomplish and the consequences.
#
# The primary way of configuring a node is via this file. This template lists
# the most important settings you may want to configure for a production cluster.
#
# Please see the documentation for further information on configuration options:
# <http://www.elastic.co/guide/en/elasticsearch/reference/current/setup-configuration.
↪html>
#
# ----- Cluster -----
#
# Use a descriptive name for your cluster:
#
cluster.name: {{ composant.cluster_name }}
#
# ----- Node -----
#
# Use a descriptive name for the node:
#
node.name: {{ inventory_hostname }}
# TODO: Better handling of this as we have to modify wich nodes are requested by ↪
↪logstash / kibana
node.master: {{ is_master|default('true') }}
node.data: {{ is_data|default('true') }}
#
# Add custom attributes to the node:
#
# node.rack: r1
#
# ----- Paths -----
#
# Path to directory where to store the data (separate multiple locations by comma):
#
path.data: {{ elasticsearch_data_dir }}
#
# Path to log files:
#
path.logs: {{ elasticsearch_log_dir }}
#
# ----- Memory -----
#
# Lock the memory on startup:
# = Disable swapping
bootstrap.memory_lock: true
#
```

(suite sur la page suivante)

(suite de la page précédente)

```

# Make sure that the `ES_HEAP_SIZE` environment variable is set to about half the_
↪memory
# available on the system and that the owner of the process is allowed to use this_
↪limit.
#
# Elasticsearch performs poorly when the system is swapping the memory.
#
# ----- Network -----
#
# Set the bind address to a specific IP (IPv4 or IPv6):
#
# Note : if installing to localhost, notably a docker container, we need to bind_
↪larger than localhost
{% if inventory_hostname == "localhost" %}
network.host: 0.0.0.0
http.cors.enabled: true
http.cors.allow-origin: "*"
{% else %}
# KWA TODO: Check it again (ansible_hostname VS inventory_hostname VS ip_service)
network.host: {{ ip_admin }}
{% endif %}
# Set a custom port for HTTP:
#
http.port: {{ composant.port_http }}
#network.port: {{ composant.port_tcp }}
transport.tcp.port: {{ composant.port_tcp }}

# For more information, see the documentation at:
# <http://www.elastic.co/guide/en/elasticsearch/reference/current/modules-network.
↪html>
#
# ----- Discovery -----
#
# Pass an initial list of hosts to perform discovery when new node is started:
# The default list of hosts is ["127.0.0.1", "[::1]"]
#
discovery.zen.ping.unicast.hosts: [ {% for host in groups['hosts-elasticsearch-log']
↪%} "{{ hostvars[host]['ip_admin'] }}" {% if not loop.last %},{% endif %} {% endfor %} ]
#
# Prevent the "split brain" by configuring the majority of nodes (total number of_
↪nodes / 2 + 1):
#
# discovery.zen.minimum_master_nodes: 3
#
# For more information, see the documentation at:
# <http://www.elastic.co/guide/en/elasticsearch/reference/current/modules-discovery.
↪html>
#
# ----- Gateway -----
#
# Block initial recovery after a full cluster restart until N nodes are started:
#
# gateway.recover_after_nodes: 3
#
# For more information, see the documentation at:
# <http://www.elastic.co/guide/en/elasticsearch/reference/current/modules-gateway.
↪html>

```

(suite sur la page suivante)

(suite de la page précédente)

```
#
# ----- Various -----
#
# Disable starting multiple nodes on a single system:
#
# node.max_local_storage_nodes: 1
#
# Require explicit names when deleting indices:
#
action.destructive_requires_name: true
```

7.2.4.2.4 Fichier /vitam/conf/elasticsearch-log/sysconfig/elasticsearch

```
#####
# Elasticsearch
#####

# Elasticsearch home directory
#ES_HOME=/usr/share/elasticsearch

# Elasticsearch configuration directory
ES_PATH_CONF={{ vitam_defaults.folder.root_path }}/conf/{{ composant.cluster_name }}

# Elasticsearch data directory
#DATA_DIR={{ vitam_defaults.folder.root_path }}/data/{{ composant.cluster_name }}

# Elasticsearch logs directory
#LOG_DIR={{ vitam_defaults.folder.root_path }}/log/{{ composant.cluster_name }}

# Elasticsearch PID directory
#PID_DIR=/var/run/{{ composant.cluster_name }}

# Heap size defaults to 256m min, 1g max
# Set ES_HEAP_SIZE to 50% of available RAM, but no more than 31g
#ES_JAVA_OPTS=

#####
# Elasticsearch service
#####

# SysV init.d
#
# The number of seconds to wait before checking if Elasticsearch started successfully,
→as a daemon process
ES_STARTUP_SLEEP_TIME=5

# Heap new generation
#ES_HEAP_NEWSIZE=

# Maximum direct memory
#ES_DIRECT_SIZE=

# Additional Java OPTS
```

(suite sur la page suivante)

(suite de la page précédente)

```

ES_JAVA_OPTS="-XX:+UseGCLogFileRotation -XX:NumberOfGCLogFiles=10 -
↳XX:GCLogFileSize=10M -XX:+PrintGCDetails -XX:+PrintGCApplicationStoppedTime"

# Configure restart on package upgrade (true, every other setting will lead to not_
↳restarting)
#RESTART_ON_UPGRADE=true

# Path to the GC log file
#ES_GC_LOG_FILE={{ vitam_defaults.folder.root_path }}/log/{{ composant.cluster_name }}
↳/gc.log

#####
# Elasticsearch service
#####

# SysV init.d
#
# When executing the init script, this user will be used to run the elasticsearch_
↳service.
# The default value is 'elasticsearch' and is declared in the init.d file.
# Note that this setting is only used by the init script. If changed, make sure that
# the configured user can read and write into the data, work, plugins and log_
↳directories.
# For systemd service, the user is usually configured in file /usr/lib/systemd/system/
↳elasticsearch.service

# Note: useless for VITAM, as the startup is managed by systemd
ES_USER={{ vitam_defaults.users.vitamdb }}
ES_GROUP={{ vitam_defaults.users.group }}

# The number of seconds to wait before checking if Elasticsearch started successfully_
↳as a daemon process
ES_STARTUP_SLEEP_TIME=5

#####
# System properties
#####

# Specifies the maximum file descriptor number that can be opened by this process
# When using Systemd, this setting is ignored and the LimitNOFILE defined in
# /usr/lib/systemd/system/elasticsearch.service takes precedence
#MAX_OPEN_FILES=65536

# The maximum number of bytes of memory that may be locked into RAM
# Set to "unlimited" if you use the 'bootstrap.memory_lock: true' option
# in elasticsearch.yml (ES_HEAP_SIZE must also be set).
# When using Systemd, the LimitMEMLOCK property must be set
# in /usr/lib/systemd/system/elasticsearch.service
#MAX_LOCKED_MEMORY=unlimited

# Maximum number of VMA (Virtual Memory Areas) a process can own
# When using Systemd, this setting is ignored and the 'vm.max_map_count'
# property is set at boot time in /usr/lib/sysctl.d/elasticsearch.conf
#MAX_MAP_COUNT=262144

```

7.2.4.2.5 Fichier `/usr/lib/tmpfiles.d/elasticsearch-log.conf`

```
d    /var/run/{{ composant.cluster_name }}    0755 {{ vitam_defaults.users.vitamdb }} {  
↪ { vitam_defaults.users.group }} - -
```

7.2.4.3 Opérations

- Démarrage du service

Les commandes suivantes sont à passer sur les différentes machines constituant le cluster Elasticsearch.

En tant qu'utilisateur root : `systemctl start vitam-elasticsearch-log`

- Arrêt du service

Les commandes suivantes sont à passer sur les différentes machines constituant le cluster Elasticsearch.

En tant qu'utilisateur root : `systemctl stop vitam-elasticsearch-log`

- Sauvegarde du service

Dans cette version du système, seule une sauvegarde à froid du service est supportée (par la sauvegarde des fichiers de données présents dans `/vitam/data`)

- Supervision du service

Contrôler le retour HTTP 200 sur l'URL `<protocole web https ou https>://<host>:<port>/`

- Exports

N/A

- gestion de la capacité

N/A

- Réouverture d'un index fermé

Les index sont fermés par action récurrente de Curator ; il est néanmoins possible de rouvrir un index fermé par la commande suivante :

```
curl -XPOST '<adresseIP>:<port>/<index_fermé>/_open'
```

Référence ¹⁴

- actions récurrentes
- cas des batches

N/A

7.2.5 elasticsearch Vitam

7.2.5.1 Présentation

Le composant `vitam-elasticsearch-data` est une instance de la base d'indexation elasticsearch stockant les informations relatives aux archives hébergées dans *VITAM*. Elle participe dans ce sens à l'indexation et la recherche des données contenues dans MongoDB.

<https://www.elastic.co/guide/en/elasticsearch/reference/2.4/indices-open-close.html>

7.2.5.2 Configuration / fichiers utiles

Se reporter au *DIN*, qui configure le *cluster* Elasticsearch de données.

Les fichiers de configuration sont définis sous `/vitam/conf/elasticsearch-data`.

7.2.5.2.1 Fichier `log4j2.properties`

```
status = error

# log action execution errors for easier debugging
logger.action.name = org.elasticsearch.action
logger.action.level = debug

appender.console.type = Console
appender.console.name = console
appender.console.layout.type = PatternLayout
appender.console.layout.pattern = [%d{ISO8601}][%-5p][%-25c{1.}] %marker%m%n

appender.syslog.type = Syslog
appender.syslog.name = syslog
appender.syslog.appName = {{ composant.cluster_name }}
appender.syslog.facility = {{ vitam_defaults.syslog_facility }}
appender.syslog.host = {{ inventory_hostname }}
appender.syslog.protocol = UDP
appender.syslog.port = 514
appender.syslog.layout.type = PatternLayout
# Note: rsyslog only parse RFC3195-formatted syslog messages by default ; AND, to
↳ make it work with log4j2, we need to start the layout by the app-name.
# IF we were in 5424, we wouldn't have to do this.
appender.syslog.layout.pattern = {{ composant.cluster_name }}: [%d{ISO8601}][%-5p][%-
↳ 25c{1.}] %marker%m%n
# appender.syslog.format = RFC5424
# appender.syslog.mdcId = esdata

appender.rolling.type = RollingFile
appender.rolling.name = rolling
appender.rolling.fileName = ${sys:es.logs.base_path}${sys:file.separator}${sys:es.
↳ logs.cluster_name}.log
appender.rolling.layout.type = PatternLayout
appender.rolling.layout.pattern = [%d{ISO8601}][%-5p][%-25c{1.}] %marker%. -10000m%n
appender.rolling.filePattern = ${sys:es.logs.base_path}${sys:file.separator}${sys:es.
↳ logs.cluster_name}-%d{yyyy-MM-dd}.log
appender.rolling.policies.type = Policies
appender.rolling.policies.time.type = TimeBasedTriggeringPolicy
appender.rolling.policies.time.interval = 1
appender.rolling.policies.time.modulate = true

rootLogger.level = info
rootLogger.appenderRef.console.ref = console
rootLogger.appenderRef.rolling.ref = rolling
rootLogger.appenderRef.syslog.ref = syslog

appender.deprecation_rolling.type = RollingFile
appender.deprecation_rolling.name = deprecation_rolling
appender.deprecation_rolling.fileName = ${sys:es.logs.base_path}${sys:file.separator}${
↳ sys:es.logs.cluster_name}_deprecation.log
```

(suite sur la page suivante)

(suite de la page précédente)

```

appender.deprecation_rolling.layout.type = PatternLayout
appender.deprecation_rolling.layout.pattern = [%d{ISO8601}][%-5p][%-25c{1.}] %marker%.
↳-10000m%n
appender.deprecation_rolling.filePattern = ${sys:es.logs.base_path}${sys:file.
↳separator}${sys:es.logs.cluster_name}_deprecation-%i.log.gz
appender.deprecation_rolling.policies.type = Policies
appender.deprecation_rolling.policies.size.type = SizeBasedTriggeringPolicy
appender.deprecation_rolling.policies.size.size = 1GB
appender.deprecation_rolling.strategy.type = DefaultRolloverStrategy
appender.deprecation_rolling.strategy.max = 4

logger.deprecation.name = org.elasticsearch.deprecation
logger.deprecation.level = warn
logger.deprecation.appenderRef.deprecation_rolling.ref = deprecation_rolling
logger.deprecation.additivity = false

appender.index_search_slowlog_rolling.type = RollingFile
appender.index_search_slowlog_rolling.name = index_search_slowlog_rolling
appender.index_search_slowlog_rolling.fileName = ${sys:es.logs.base_path}${sys:file.
↳separator}${sys:es.logs.cluster_name}_index_search_slowlog.log
appender.index_search_slowlog_rolling.layout.type = PatternLayout
appender.index_search_slowlog_rolling.layout.pattern = [%d{ISO8601}][%-5p][%-25c]
↳%marker%. -10000m%n
appender.index_search_slowlog_rolling.filePattern = ${sys:es.logs.base_path}${
↳{sys:file.separator}${sys:es.logs.cluster_name}_index_search_slowlog-%d{yyyy-MM-dd}.
↳log
appender.index_search_slowlog_rolling.policies.type = Policies
appender.index_search_slowlog_rolling.policies.time.type = TimeBasedTriggeringPolicy
appender.index_search_slowlog_rolling.policies.time.interval = 1
appender.index_search_slowlog_rolling.policies.time.modulate = true

logger.index_search_slowlog_rolling.name = index.search.slowlog
logger.index_search_slowlog_rolling.level = {{ composant.index_search_slowlog_rolling_
↳level }}
logger.index_search_slowlog_rolling.appenderRef.index_search_slowlog_rolling.ref =
↳index_search_slowlog_rolling
logger.index_search_slowlog_rolling.additivity = false

appender.index_indexing_slowlog_rolling.type = RollingFile
appender.index_indexing_slowlog_rolling.name = index_indexing_slowlog_rolling
appender.index_indexing_slowlog_rolling.fileName = ${sys:es.logs.base_path}${sys:file.
↳separator}${sys:es.logs.cluster_name}_index_indexing_slowlog.log
appender.index_indexing_slowlog_rolling.layout.type = PatternLayout
appender.index_indexing_slowlog_rolling.layout.pattern = [%d{ISO8601}][%-5p][%-25c]
↳%marker%. -10000m%n
appender.index_indexing_slowlog_rolling.filePattern = ${sys:es.logs.base_path}${
↳{sys:file.separator}${sys:es.logs.cluster_name}_index_indexing_slowlog-%d{yyyy-MM-
↳dd}.log
appender.index_indexing_slowlog_rolling.policies.type = Policies
appender.index_indexing_slowlog_rolling.policies.time.type = TimeBasedTriggeringPolicy
appender.index_indexing_slowlog_rolling.policies.time.interval = 1
appender.index_indexing_slowlog_rolling.policies.time.modulate = true

logger.index_indexing_slowlog.name = index.indexing.slowlog.index
logger.index_indexing_slowlog.level = {{ composant.index_indexing_slowlog_level }}
logger.index_indexing_slowlog.appenderRef.index_indexing_slowlog_rolling.ref = index_
↳indexing_slowlog_rolling

```

(suite sur la page suivante)

(suite de la page précédente)

```
logger.index_indexing_slowlog.additivity = false
```

7.2.5.2.2 Fichier jvm.options

```
## JVM configuration

#####
## IMPORTANT: JVM heap size
#####
##
## You should always set the min and max JVM heap
## size to the same value. For example, to set
## the heap to 4 GB, set:
##
## -Xms4g
## -Xmx4g
##
## See https://www.elastic.co/guide/en/elasticsearch/reference/current/heap-size.html
## for more information
##
#####

# Xms represents the initial size of total heap space
# Xmx represents the maximum size of total heap space

-Xms{{ elasticsearch_memory }}
-Xmx{{ elasticsearch_memory }}

#####
## Expert settings
#####
##
## All settings below this section are considered
## expert settings. Don't tamper with them unless
## you understand what you are doing
##
#####

## GC configuration
-XX:+UseConcMarkSweepGC
-XX:CMSInitiatingOccupancyFraction=75
-XX:+UseCMSInitiatingOccupancyOnly

## optimizations

# pre-touch memory pages used by the JVM during initialization
-XX:+AlwaysPreTouch

## basic

# force the server VM (remove on 32-bit client JVMs)
-server

# explicitly set the stack size (reduce to 320k on 32-bit client JVMs)
```

(suite sur la page suivante)

(suite de la page précédente)

```
-Xsslm

# set to headless, just in case
-Djava.awt.headless=true

# ensure UTF-8 encoding by default (e.g. filenames)
-Dfile.encoding=UTF-8

# use our provided JNA always versus the system one
-Djna.nosys=true

# use old-style file permissions on JDK9
-Djdk.io.permissionsUseCanonicalPath=true

# flags to configure Netty
-Dio.netty.noUnsafe=true
-Dio.netty.noKeySetOptimization=true
-Dio.netty.recycler.maxCapacityPerThread=0

# log4j 2
-Dlog4j.shutdownHookEnabled=false
-Dlog4j2.disable.jmx=true
-Dlog4j.skipJansi=true

## heap dumps

# generate a heap dump when an allocation from the Java heap fails
# heap dumps are created in the working directory of the JVM
-XX:+HeapDumpOnOutOfMemoryError

# specify an alternative path for heap dumps
# ensure the directory exists and has sufficient space
-XX:HeapDumpPath={{ elasticsearch_log_dir }}

## GC logging

-XX:+UseGCLogFileRotation
-XX:NumberOfGCLogFiles=10
-XX:GCLogFileSize=10M
-XX:+PrintGCDetails
-XX:+PrintGCApplicationStoppedTime

#-XX:+PrintGCDetails
#-XX:+PrintGCTimeStamps
#-XX:+PrintGCDateStamps
#-XX:+PrintClassHistogram
#-XX:+PrintTenuringDistribution
#-XX:+PrintGCApplicationStoppedTime

# log GC status to a file with time stamps
# ensure the directory exists
#-Xloggc:${loggc}

# By default, the GC log file will not rotate.
# By uncommenting the lines below, the GC log file
# will be rotated every 128MB at most 32 times.
#-XX:+UseGCLogFileRotation
```

(suite sur la page suivante)

(suite de la page précédente)

```
#-XX:NumberOfGCLogFiles=32
#-XX:GCLogFileSize=128M

# Elasticsearch 5.0.0 will throw an exception on unquoted field names in JSON.
# If documents were already indexed with unquoted fields in a previous version
# of Elasticsearch, some operations may throw errors.
#
# WARNING: This option will be removed in Elasticsearch 6.0.0 and is provided
# only for migration purposes.
#-Delasticsearch.json.allow_unquoted_field_names=true
```

7.2.5.2.3 Fichier elasticsearch.yml

```
# ===== Elasticsearch Configuration =====
#
# NOTE: Elasticsearch comes with reasonable defaults for most settings.
#       Before you set out to tweak and tune the configuration, make sure you
#       understand what are you trying to accomplish and the consequences.
#
# The primary way of configuring a node is via this file. This template lists
# the most important settings you may want to configure for a production cluster.
#
# Please see the documentation for further information on configuration options:
# <http://www.elastic.co/guide/en/elasticsearch/reference/current/setup-configuration.
↪html>
#
# ----- Cluster -----
#
# Use a descriptive name for your cluster:
#
cluster.name: {{ composant.cluster_name }}
#
# ----- Node -----
#
# Use a descriptive name for the node:
#
node.name: {{ inventory_hostname }}
# TODO: Better handling of this as we have to modify wich nodes are requested by VITAM
node.master: {{ is_master|default('true') }}
node.data: {{ is_data|default('true') }}
#
# Add custom attributes to the node:
#
# node.rack: r1
#
# ----- Paths -----
#
# Path to directory where to store the data (separate multiple locations by comma):
#
path.data: {{ elasticsearch_data_dir }}
#
# Path to log files:
#
path.logs: {{ elasticsearch_log_dir }}
```

(suite sur la page suivante)

(suite de la page précédente)

```

#
# ----- Memory -----
#
# Lock the memory on startup:
# = Disable swapping
bootstrap.memory_lock: true
#
# Make sure that the 'ES_HEAP_SIZE' environment variable is set to about half the_
↳memory
# available on the system and that the owner of the process is allowed to use this_
↳limit.
#
# Elasticsearch performs poorly when the system is swapping the memory.
#
# ----- Network -----
#
# Set the bind address to a specific IP (IPv4 or IPv6):
#
# Note : if installing to localhost, notably a docker container, we need to bind_
↳larger than localhost
{% if inventory_hostname == "localhost" %}
network.host: 0.0.0.0
http.cors.enabled: true
http.cors.allow-origin: "*"
{% else %}
network.host: {{ ip_service }}
{% endif %}
#
# Set a custom port for HTTP:
#
http.port: {{ composant.port_http }}
transport.tcp.port: {{ composant.port_tcp }}
#
# For more information, see the documentation at:
# <http://www.elastic.co/guide/en/elasticsearch/reference/current/modules-network.
↳html>
#
# ----- Discovery -----
#
# Pass an initial list of hosts to perform discovery when new node is started:
# The default list of hosts is ["127.0.0.1", "[::1]"]
#
discovery.zen.ping.unicast.hosts: [ {% for host in groups['hosts-elasticsearch-data']
↳ } }"{{ hostvars[host]['ip_service'] }}" {% if not loop.last %},{% endif %} {% endfor %}
↳ ]
#
# Prevent the "split brain" by configuring the majority of nodes (total number of_
↳nodes / 2 + 1):
#
discovery.zen.minimum_master_nodes: { { (groups['hosts-elasticsearch-data']|length /_
↳2)+1) | round (0, 'floor') | int } }
#
# For more information, see the documentation at:
# <http://www.elastic.co/guide/en/elasticsearch/reference/current/modules-discovery.
↳html>
#
# ----- Gateway -----

```

(suite sur la page suivante)

(suite de la page précédente)

```

#
# Block initial recovery after a full cluster restart until N nodes are started:
#
gateway.expected_nodes: {{ (groups['hosts-elasticsearch-data'] | length) }}
gateway.recover_after_nodes: {{ ((groups['hosts-elasticsearch-data']|length / 2)+1)|
↳round (0, 'floor')| int }}
#
# For more information, see the documentation at:
# <http://www.elastic.co/guide/en/elasticsearch/reference/current/modules-gateway.
↳html>
#
# ----- Various -----
#
# Disable starting multiple nodes on a single system:
#
# node.max_local_storage_nodes: 1
#
# Require explicit names when deleting indices:
#
action.destructive_requires_name: true

# For Vitam multiquery
indices.query.bool.max_clause_count: 10000

{% if composant.index_buffer_size_ratio is defined %}
# some performamnce tuning ; see https://www.elastic.co/guide/en/elasticsearch/
↳reference/6.4/tune-for-indexing-speed.html
# 0.1 may be enough, cots_vars declares {{ composant.index_buffer_size_ratio }} as
↳ratio on total memory {{ elasticsearch_memory }}
indices.memory.index_buffer_size: {{ ((elasticsearch_memory_value|int)*(composant.
↳index_buffer_size_ratio|float))|round (0, 'floor')| int }}{{ elasticsearch_memory_
↳unit }}
{% endif %}

# thread_pool configuration
thread_pool:
  index:
    size: {{ (ansible_processor_cores * ansible_processor_threads_per_core) |
↳round (0, 'floor') | int }}
    queue_size: 5000
  get:
    size: {{ (ansible_processor_cores * ansible_processor_threads_per_core) |
↳round (0, 'floor') | int }}
    queue_size: 5000
  search:
    size: {{ ((ansible_processor_cores * ansible_processor_threads_per_core * 3 /
↳2) + 1) | round (0, 'floor') | int }}
    min_queue_size: 1000
    queue_size: 5000
  write:
    size: {{ (ansible_processor_cores * ansible_processor_threads_per_core + 1) |
↳round (0, 'floor') | int }}
    queue_size: 5000
  warmer:
    core: 1
    max: {{ ((ansible_processor_cores * ansible_processor_threads_per_core / 2) +
↳0.5) | round (0, 'floor') | int }}

```

(suite sur la page suivante)

(suite de la page précédente)

```

keep_alive: 2m

# Note : the 0.5 in the previous expression is for there is only 1 CPU (else the
↳thread pool size would be zero) ! ; Note bis : max 10 threads #
# Note : in ES5 and further : the thread pool "refresh" is of type scaling with a
↳keep-alive of 5m and a max of min(10, (# of available processors)/2)

```

7.2.5.2.4 Fichier sysconfig/elasticsearch

```

#####
# Elasticsearch
#####

# Elasticsearch home directory
#ES_HOME=/usr/share/elasticsearch

# Elasticsearch configuration directory
ES_PATH_CONF={{ vitam_defaults.folder.root_path }}/conf/{{ composant.cluster_name }}

# Elasticsearch data directory
#DATA_DIR={{ vitam_defaults.folder.root_path }}/data/{{ composant.cluster_name }}

# Elasticsearch logs directory
#LOG_DIR={{ vitam_defaults.folder.root_path }}/log/{{ composant.cluster_name }}

# Elasticsearch PID directory
#PID_DIR=/var/run/{{ composant.cluster_name }}

# Heap size defaults to 256m min, 1g max
# Set ES_HEAP_SIZE to 50% of available RAM, but no more than 31g
#ES_JAVA_OPTS=

#####
# Elasticsearch service
#####

# SysV init.d
#
# The number of seconds to wait before checking if Elasticsearch started successfully
↳as a daemon process
ES_STARTUP_SLEEP_TIME=5

# Heap new generation
#ES_HEAP_NEWSIZE=

# Maximum direct memory
#ES_DIRECT_SIZE=

# Additional Java OPTS
ES_JAVA_OPTS="-XX:+UseGCLogFileRotation -XX:NumberOfGCLogFiles=10 -
↳XX:GCLogFileSize=10M -XX:+PrintGCDetails -XX:+PrintGCApplicationStoppedTime"

# Configure restart on package upgrade (true, every other setting will lead to not
↳restarting)

```

(suite sur la page suivante)

(suite de la page précédente)

```
#RESTART_ON_UPGRADE=true

# Path to the GC log file
#ES_GC_LOG_FILE={{ vitam_defaults.folder.root_path }}/log/{{ composant.cluster_name }}
↪/gc.log

#####
# Elasticsearch service
#####

# SysV init.d
#
# When executing the init script, this user will be used to run the elasticsearch_
↪service.
# The default value is 'elasticsearch' and is declared in the init.d file.
# Note that this setting is only used by the init script. If changed, make sure that
# the configured user can read and write into the data, work, plugins and log_
↪directories.
# For systemd service, the user is usually configured in file /usr/lib/systemd/system/
↪elasticsearch.service

# Note: useless for VITAM, as the startup is managed by systemd
ES_USER={{ vitam_defaults.users.vitamdb }}
ES_GROUP={{ vitam_defaults.users.group }}

# The number of seconds to wait before checking if Elasticsearch started successfully_
↪as a daemon process
ES_STARTUP_SLEEP_TIME=5

#####
# System properties
#####

# Specifies the maximum file descriptor number that can be opened by this process
# When using Systemd, this setting is ignored and the LimitNOFILE defined in
# /usr/lib/systemd/system/elasticsearch.service takes precedence
#MAX_OPEN_FILES=65536

# The maximum number of bytes of memory that may be locked into RAM
# Set to "unlimited" if you use the 'bootstrap.memory_lock: true' option
# in elasticsearch.yml (ES_HEAP_SIZE must also be set).
# When using Systemd, the LimitMEMLOCK property must be set
# in /usr/lib/systemd/system/elasticsearch.service
#MAX_LOCKED_MEMORY=unlimited

# Maximum number of VMA (Virtual Memory Areas) a process can own
# When using Systemd, this setting is ignored and the 'vm.max_map_count'
# property is set at boot time in /usr/lib/sysctl.d/elasticsearch.conf
#MAX_MAP_COUNT=262144
```

7.2.5.2.5 Fichier /usr/lib/tmpfiles.d/elasticsearch-data.conf

```
d    /var/run/{{ composant.cluster_name }}    0755 {{ vitam_defaults.users.vitamdb }} {
↪{{ vitam_defaults.users.group }} - -
```


7.2.5.3 Opérations

- Démarrage du service

Les commandes suivantes sont à passer sur les différentes machines constituant le cluster Elasticsearch.

En tant qu'utilisateur root : `systemctl start vitam-elasticsearch-data`

- Arrêt du service

Les commandes suivantes sont à passer sur les différentes machines constituant le cluster Elasticsearch.

En tant qu'utilisateur root : `systemctl stop vitam-elasticsearch-data`

- Sauvegarde du service

Dans cette version du système, seule une sauvegarde à froid du service est supportée (par la sauvegarde des fichiers de données présents dans `/vitam/data`)

- Supervision du service

Contrôler le retour HTTP 200 sur l'URL `<protocole web https ou https>://<host>:<port>/`

- Exports

N/A

- gestion de la capacité

N/A

- actions récurrentes

- cas des batches

N/A

7.2.6 Kibana

7.2.6.1 Présentation

Kibana est l'outil permettant de représenter de façon agrégée des données stockées dans ElasticSearch.

Afin de forcer la bonne version de Kibana, *VITAM* déploie un installeur-chapeau `vitam-kibana`.

Prudence : le composant kibana ne peut se connecter qu'à un cluster ElasticSearch ; pour superviser les clusters Elasticsearch de données et de log, il convient de définir des machines différentes (pour chaque kibana) durant l'installation.

VITAM injecte des *dashboards* durant l'installation.

7.2.6.2 Configuration / fichiers utiles

Les fichiers de configuration sont gérés par les procédures d'installation ou de mise à niveau de l'environnement *VITAM*. Se référer au *DIN*.

Le *playbook* d'installation effectue des actions de modification du fichier de configuration standard `/etc/kibana/kibana.yml`.

7.2.6.3 Opérations

- Démarrage du service

En tant qu'utilisateur root : `systemctl start kibana`

- Arrêt du service

En tant qu'utilisateur root : `systemctl stop kibana`

- Sauvegarde du service

Ce service ne nécessite pas de sauvegarde particulière.

- Logs

Les logs applicatifs sont envoyés par rsyslog à la solution de centralisation des logs ; il est néanmoins possible d'en versionner une représentation par la commande :

```
journalctl --unit kibana
```

- Supervision du service

Kibana possède une IHM accessible via la « patte » d'administration :

`http(s) ://<adresse> :5601/`

- Exports

N/A

- gestion de la capacité

N/A

- actions récurrentes

- cas des batches

N/A

7.2.7 log server

7.2.7.1 Présentation

Ce composant représente en réalité l'ensemble des 3 composants suivants :

- Kibana, pour la présentation des dashboards de logs et de métriques ;
- Logstash, pour l'analyse et la centralisation des logs ;
- Curator, pour la maintenance des index elasticsearch de log.

Le présent chapitre ne s'intéressera qu'à logstash.

7.2.7.2 Configuration / fichiers utiles

L'ansible se charge du paramétrage de ces composants.

7.2.7.3 Opérations

- Démarrage du service

En tant qu'utilisateur root :

Pré-requis : le cluster elasticsearch associé est déjà démarré.

```
systemctl start logstash
```

- Arrêt du service

En tant qu'utilisateur root :

```
systemctl stop logstash
```

Post-requis : le cluster elasticsearch-log associé est arrêté.

- Sauvegarde du service

Ce service ne nécessite pas de sauvegarde particulière.

- Supervision du service

N/A

- Exports

N/A

- gestion de la capacité

N/A

- actions récurrentes

batch Curator, actuellement purgeant les données de plus de XX jours (selon ce qui a été défini dans l'inventaire de ansible) dans Elasticsearch de logs.

- cas des batches

Curator

7.2.8 mongoC

7.2.8.1 Présentation

Replicaset mongoDB servant à stocker la configuration MongoDB (clés de sharding, shards, ...) lors de l'utilisation de MondoDB en mode sharding.

7.2.8.2 Configuration / fichiers utiles

7.2.8.2.1 Fichier `mongoc.conf`

```
# mongod.conf

# for documentation of all options, see:
#   http://docs.mongodb.org/manual/reference/configuration-options/

# where to write logging data.
systemLog:
  destination: syslog
  syslogFacility: local0
  logAppend: true

# Where and how to store data.
```

(suite sur la page suivante)

(suite de la page précédente)

```

storage:
  dbPath: {{ mongo_db_path }}
  directoryPerDB: true

# network interfaces
net:
  port: {{ mongodb.mongoc_port }}
  bindIp: {{ ip_service }}{% if groups['hosts-dev-tools'] | length > 0 and ip_service_
↳ != ip_admin %},{% ip_admin %}{% endif %}

# operationProfiling:
replication:
  replSetName: configsvr # name of the replica set
  enableMajorityReadConcern: true

sharding:
  clusterRole: configsvr # role du shard

# ansible managed security conf

```

7.2.8.2 Fichier keyfile

```
{{ mongodb[mongo_cluster_name].passphrase }}
```

7.2.8.3 Opérations

- Démarrage du service

En tant qu'utilisateur root : `systemctl start vitam-mongoc`

- Arrêt du service

En tant qu'utilisateur root : `systemctl stop vitam-mongoc`

- Sauvegarde du service

Il est recommandé d'effectuer des sauvegardes régulières des données.

Pour cela, la procédure à suivre est :

1. Arrêt du service
2. Lancement d'un backup (à définir)
3. Démarrage du service

- Supervision du service
- Exports
- gestion de la capacité

N/A

- actions récurrentes
- cas des batches

Cas de l'export tous les soirs/matins ?

7.2.9 mongoD

7.2.9.1 Présentation

Replicaset MongoDB stockant les données métier de Vitam.

7.2.9.2 Configuration / fichiers utiles

7.2.9.2.1 Fichier `mongod.conf`

```
# mongod.conf

# for documentation of all options, see:
#   http://docs.mongodb.org/manual/reference/configuration-options/

# where to write logging data.
systemLog:
  destination: syslog
  syslogFacility: local0
  logAppend: true

# Where and how to store data.
storage:
  dbPath: {{ mongo_db_path }}
  directoryPerDB: true

{% if mongod_memory is defined and mongod_memory != '' %}
  wiredTiger:
    engineConfig:
      cacheSizeGB: {{ mongod_memory }}
{% endif %}

# network interfaces
net:
  port: {{ mongodb.mongod_port }}
  bindIp: {{ ip_service }}{% if groups['hosts-dev-tools'] | length > 0 and ip_service_
↪ != ip_admin %},{{ ip_admin }}{% endif %}

# operationProfiling:
replication:
  replSetName: shard{{ mongo_shard_id }} # name of the replica set
  enableMajorityReadConcern: true

sharding:
  clusterRole: shardsvr # role du shard

# ansible managed security conf
```

7.2.9.2.2 Fichier keyfile

```
{ { mongodb[mongo_cluster_name].passphrase } }
```

7.2.9.3 Opérations

- Démarrage du service

En tant qu'utilisateur root : `systemctl start vitam-mongod`

- Arrêt du service

En tant qu'utilisateur root : `systemctl stop vitam-mongod`

- Sauvegarde du service

Il est recommandé d'effectuer des sauvegardes régulières des données.

Pour cela, la procédure à suivre est :

1. Arrêt du service
2. Lancement d'un backup (à définir)
3. Démarrage du service

- Supervision du service

Via mongo-express ?

- Exports
- gestion de la capacité

N/A

- actions récurrentes
- cas des batches

Cas de l'export tous les soirs/matins ?

7.2.10 mongoS

7.2.10.1 Présentation

Point d'accès frontal à la base de données MongoDB de Vitam. Redirige sur le bon shard en fonction de la clé de sharding positionnée sur la collection.

7.2.10.2 Configuration / fichiers utiles

7.2.10.2.1 Fichier mongos.conf

```
# mongod.conf

# for documentation of all options, see:
#   http://docs.mongodb.org/manual/reference/configuration-options/

# where to write logging data.
systemLog:
  destination: syslog
  syslogFacility: local0
  logAppend: true

# network interfaces
net:
  port: {{ mongodb.mongos_port }}
  bindIp: {{ ip_service }}{% if groups['hosts-dev-tools'] | length > 0 and ip_service_
↳ != ip_admin %},{{ ip_admin }}{% endif %}

sharding:
  configDB: configsvr/{% for item in mongoc_list %}{{ hostvars[item]['ip_service'] }}:
↳ {{ mongodb.mongoc_port }}{% if not loop.last %},{% endif %}{% endfor %}

# ansible managed security conf
```

7.2.10.2.2 Fichier keyfile

```
{{ mongodb[mongo_cluster_name].passphrase }}
```

7.2.10.3 Opérations

- Démarrage du service

En tant qu'utilisateur root : `systemctl start vitam-mongos`

- Arrêt du service

En tant qu'utilisateur root : `systemctl stop vitam-mongos`

- Sauvegarde du service

Il est recommandé d'effectuer des sauvegardes régulières des données.

Pour cela, la procédure à suivre est :

1. Arrêt du service
2. Lancement d'un backup (à définir)
3. Démarrage du service

- Supervision du service

Via mongo-express ?

- Exports
- gestion de la capacité

N/A

- actions récurrentes
- cas des batches

Cas de l'export tous les soirs/matins ?

7.2.11 siegfried

7.2.11.1 Présentation

Siegfried est un outil permettant la détection de format d'un fichier.

7.2.11.2 Configuration / fichiers utiles

Les fichiers de configuration sont gérés par les procédures d'installation ou de mise à niveau de l'environnement *VITAM*. Se référer au *DIN*.

7.2.11.3 Opérations

- Démarrage du service

En tant qu'utilisateur root : `systemctl start vitam-siegfried`

- Arrêt du service

En tant qu'utilisateur root : `systemctl stop vitam-siegfried`

Avertissement : ne pas oublier que cela peut perturber le comportement de certains composants Vitam (ingest-external et worker).

- Sauvegarde du service

Ce service ne nécessite pas de sauvegarde particulière.

- Logs

Les logs applicatifs sont envoyés par rsyslog à la solution de centralisation des logs ; il est néanmoins possible d'en visionner une représentation par la commande :

```
journalctl --unit vitam-siegfried
```

- Supervision du service

N/A

- Exports

N/A

- gestion de la capacité

N/A

- actions récurrentes
- cas des batches

N/A

- Montée de version du fichier de signatures

Se reporter à *Montée de version du fichier de signature de Siegfried* (page 18)

Exploitation des composants de la solution logicielle VITAM

Les sections qui suivent donnent une description plus fine pour l'exploitation des services VITAM.

8.1 Généralités

Les composants de la solution logicielle *VITAM* sont déployés par un *playbook* ansible qui :

1. déploie, selon l'inventaire employé, les *packages* nécessaires
2. applique la configuration de chaque composant selon son contexte défini dans l'inventaire

Les composants *VITAM* sont décrits ci-après.

Avertissement : En cas de modification de la configuration, redémarrer le service associé.

8.2 Composants

8.2.1 Fichiers communs

Les composants de la solution logicielle *VITAM* utilisent un socle de fichiers communs.

8.2.1.1 Fichier `/vitam/conf/<composant>/sysconfig/java_opts`

Ce fichier définit les JVMARGS.

```
1 # *****
2 # Copyright French Prime minister Office/SGMAP/DINSIC/Vitam Program (2015-2019)
3 #
4 # contact.vitam@culture.gouv.fr
```

(suite sur la page suivante)

(suite de la page précédente)

```

5 #
6 # This software is a computer program whose purpose is to implement a digital
  ↳ archiving back-office system managing
7 # high volumetry securely and efficiently.
8 #
9 # This software is governed by the CeCILL 2.1 license under French law and abiding by
  ↳ the rules of distribution of free
10 # software. You can use, modify and/ or redistribute the software under the terms of
  ↳ the CeCILL 2.1 license as
11 # circulated by CEA, CNRS and INRIA at the following URL "http://www.cecill.info".
12 #
13 # As a counterpart to the access to the source code and rights to copy, modify and
  ↳ redistribute granted by the license,
14 # users are provided only with a limited warranty and the software's author, the
  ↳ holder of the economic rights, and the
15 # successive licensors have only limited liability.
16 #
17 # In this respect, the user's attention is drawn to the risks associated with loading,
  ↳ using, modifying and/or
18 # developing or reproducing the software by the user in light of its specific status
  ↳ of free software, that may mean
19 # that it is complicated to manipulate, and that also therefore means that it is
  ↳ reserved for developers and
20 # experienced professionals having in-depth computer knowledge. Users are therefore
  ↳ encouraged to load and test the
21 # software's suitability as regards their requirements in conditions enabling the
  ↳ security of their systems and/or data
22 # to be ensured and, more generally, to use and operate it in the same conditions as
  ↳ regards security.
23 #
24 # The fact that you are presently reading this means that you have had knowledge of
  ↳ the CeCILL 2.1 license and that you
25 # accept its terms.
26 #*****
27 JAVA_OPTS="{{ vitam_struct.jvm_opts.gc | default(gc_opts) }}" {{ vitam_struct.jvm_opts.
  ↳ memory | default(memory_opts) }} {{ vitam_struct.jvm_opts.java | default(java_opts)
  ↳ }} -Dorg.owasp.esapi.resources={{ vitam_folder_conf }} -Dlogback.configurationFile={{
  ↳ {{ vitam_folder_conf }}/logback.xml -Dvitam.config.folder={{ vitam_folder_conf }} -
  ↳ Dvitam.data.folder={{ vitam_folder_data }} -Dvitam.tmp.folder={{ vitam_folder_tmp }}
  ↳ -Dvitam.log.folder={{ vitam_folder_log }} -Djava.security.properties={{ vitam_
  ↳ folder_conf }}/java.security -XX:+HeapDumpOnOutOfMemoryError -XX:HeapDumpPath={{
  ↳ vitam_folder_log }}{% if vitam_struct.jvm_log %} -XX:+UnlockDiagnosticVMOptions -
  ↳ XX:+LogVMOutput -XX:LogFile={{ vitam_folder_log }}/jvm.log{% endif %}"
28 JAVA_ARGS="{{ vitam_folder_conf }}/{{ vitam_struct.vitam_component }}.conf"

```

8.2.1.2 Fichier /vitam/conf/<composant>/logback.xml

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <configuration>
3 {% if vitam_struct.logback_rolling_policy|lower == "true" %}
4
5     <appender name="FILE" class="ch.qos.logback.core.rolling.RollingFileAppender">
6         <rollingPolicy class="ch.qos.logback.core.rolling.
  ↳ TimeBasedRollingPolicy">
7             <fileNamePattern>{{ vitam_folder_log }}/accesslog-{{ vitam_
  ↳ struct.vitam_component }}.%d{yyyy-MM-dd}.log</fileNamePattern>

```

(suite sur la page suivante)

(suite de la page précédente)

```

8         <maxHistory>{{ days_to_delete_access_local }}</maxHistory>
9         <totalSizeCap>14GB</totalSizeCap>
10        </rollingPolicy>
11    {% else %}}
12
13        <appender name="FILE" class="ch.qos.logback.core.FileAppender">
14            <file>{{ vitam_folder_log }}/accesslog-{{ vitam_struct.vitam_
15    ↳component }}.log</file>
16            <append>true</append>
17        </appender>
18        <encoder>
19            <pattern>%h %l %u %t "%r" %s %b "%i{Referer}" "%i{User-agent}
20    ↳" %D %i{X-Request-Id} %i{X-Tenant-Id} %i{X-Application-Id}</pattern>
21            </encoder>
22        </appender>
23        <appender-ref ref="FILE" />
24    </configuration>

```

8.2.1.3 Fichier /vitam/conf/<composant>/logback-access.xml

```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <configuration>
3
4      <!-- Send debug messages to System.out -->
5      <appender name="STDOUT" class="ch.qos.logback.core.ConsoleAppender">
6          <!-- By default, encoders are assigned the type ch.qos.logback.
7    ↳classic.encoder.PatternLayoutEncoder -->
8          <encoder>
9              <pattern>%d{ISO8601} [%thread] [%X{X-Request-Id}] %-5level
10    ↳%logger - %replace(%caller{1..2}){'Caller\+1' at |\n',''} : %msg
11    ↳%rootException%n</pattern>
12          </encoder>
13      </appender>
14
15  {% if vitam_struct.logback_rolling_policy|lower == "true" %}
16      <appender name="FILE" class="ch.qos.logback.core.rolling.RollingFileAppender">
17          <rollingPolicy class="ch.qos.logback.core.rolling.
18    ↳SizeAndTimeBasedRollingPolicy">
19              <fileNamePattern>{{ vitam_folder_log }}/{{ vitam_struct.vitam_
20    ↳component }}.%d{yyyy-MM-dd}.%i.log</fileNamePattern>
21              <maxFileSize>{{ vitam_struct.logback_max_file_size }}</
22    ↳maxFileSize>
23              <maxHistory>{{ days_to_delete_logback_logfiles }}</maxHistory>
24              <totalSizeCap>{{ vitam_struct.logback_total_size_cap }}</
25    ↳totalSizeCap>
26          </rollingPolicy>
27  {% else %}
28      <appender name="FILE" class="ch.qos.logback.core.FileAppender">
29          <file>{{ vitam_folder_log }}/{{ vitam_struct.vitam_component }}.log</
30    ↳file>
31          <append>true</append>
32      </appender>
33      <encoder>
34          <pattern>%d{ISO8601} [%thread] [%X{X-Request-Id}] %-5level
35    ↳%logger - %replace(%caller{1..2}){'Caller\+1' at |\n',''} : %msg %rootException%n
36    ↳</pattern>

```

(suite sur la page suivante)

(suite de la page précédente)

```

27         </encoder>
28     </appender>
29
30     {% if vitam_struct.logback_rolling_policy|lower == "true" %}
31         <appender name="SECURITY" class="ch.qos.logback.core.rolling.
↪RollingFileAppender">
32             <rollingPolicy class="ch.qos.logback.core.rolling.
↪SizeAndTimeBasedRollingPolicy">
33                 <fileNamePattern>{{ vitam_folder_log }}/{{ vitam_struct.vitam_
↪component }}_security.%d{yyyy-MM-dd}.%i.log</fileNamePattern>
34                 <maxFileSize>{{ vitam_struct.logback_max_file_size }}</
↪maxFileSize>
35                 <maxHistory>{{ days_to_delete_logback_logfiles }}</maxHistory>
36                 <totalSizeCap>{{ vitam_struct.logback_total_size_cap }}</
↪totalSizeCap>
37             </rollingPolicy>
38         {% else %}
39             <appender name="SECURITY" class="ch.qos.logback.core.FileAppender">
40                 <file>{{ vitam_folder_log }}/{{ vitam_struct.vitam_component }}_
↪security.log</file>
41                 <append>true</append>
42         {% endif %}
43         <encoder>
44             <pattern>%d{ISO8601} [[%thread]] [%X{X-Request-Id}] %-5level
↪%logger - %replace(%caller{1..2}){'Caller\+1' at |\n',''} : %msg %rootException%n
45             </pattern>
46         </encoder>
47     </appender>
48
49     {% if vitam_struct.vitam_component == 'storage' %}
50     {% if vitam_struct.logback_rolling_policy|lower == "true" %}
51         <appender name="OFFERSYNC" class="ch.qos.logback.core.rolling.
↪RollingFileAppender">
52             <rollingPolicy class="ch.qos.logback.core.rolling.
↪SizeAndTimeBasedRollingPolicy">
53                 <fileNamePattern>{{ vitam_folder_log }}/{{ vitam_struct.vitam_
↪component }}_offer_sync.%d{yyyy-MM-dd}.%i.log</fileNamePattern>
54                 <maxFileSize>{{ vitam_struct.logback_max_file_size }}</
↪maxFileSize>
55                 <maxHistory>{{ days_to_delete_logback_logfiles }}</maxHistory>
56                 <totalSizeCap>{{ vitam_struct.logback_total_size_cap }}</
↪totalSizeCap>
57             </rollingPolicy>
58         {% else %}
59             <appender name="OFFERSYNC" class="ch.qos.logback.core.FileAppender">
60                 <file>{{ vitam_folder_log }}/{{ vitam_struct.vitam_component }}_offer_
↪sync.log</file>
61                 <append>true</append>
62         {% endif %}
63         <encoder>
64             <pattern>%d{ISO8601} [[%thread]] [%X{X-Request-Id}] %-5level
↪%logger - %replace(%caller{1..2}){'Caller\+1' at |\n',''} : %msg %rootException%n
65             </pattern>
66         </encoder>
67     </appender>
68     {% endif %}
69

```

(suite sur la page suivante)

(suite de la page précédente)

```

70 {% if vitam_struct.vitam_component == 'offer' %}
71   {% if vitam_struct.logback_rolling_policy|lower == "true" %}
72     <appender name="OFFER_TAPE" class="ch.qos.logback.core.rolling.
↪RollingFileAppender">
73       <rollingPolicy class="ch.qos.logback.core.rolling.
↪SizeAndTimeBasedRollingPolicy">
74         <fileNamePattern>{{ vitam_folder_log }}/{{ vitam_struct.vitam_
↪component }}_offer_tape.%d{yyyy-MM-dd}%.i.log</fileNamePattern>
75         <maxFileSize>{{ vitam_struct.logback_max_file_size }}</
↪maxFileSize>
76         <maxHistory>{{ days_to_delete_logback_logfiles }}</maxHistory>
77         <totalSizeCap>{{ vitam_struct.logback_total_size_cap }}</
↪totalSizeCap>
78       </rollingPolicy>
79     {% else %}
80       <appender name="OFFER_TAPE" class="ch.qos.logback.core.FileAppender">
81         <file>{{ vitam_folder_log }}/{{ vitam_struct.vitam_component }}_offer_
↪tape.log</file>
82         <append>true</append>
83     {% endif %}
84     <encoder>
85       <pattern>%d{ISO8601} [%thread] [%X{X-Request-Id}] %-5level
↪%logger - %replace(%caller{1..2}){'Caller\+1      at |\n',''} : %msg %rootException%n
86     </pattern>
87   </encoder>
88 </appender>
89 {% endif %}
90
91   <appender name="SYSLOG" class="ch.qos.logback.classic.net.SyslogAppender">
92     <syslogHost>localhost</syslogHost>
93     <facility>{{ vitam_defaults.syslog_facility }}</facility>
94     <suffixPattern>vitam-{{ vitam_struct.vitam_component }}: %d{ISO8601}
↪[%thread] [%X{X-Request-Id}] %-5level %logger - %replace(%caller{1..2}){
↪'Caller\+1      at |\n',''} : %msg %rootException%n</suffixPattern>
95   </appender>
96   <!-- By default, the level of the root level is set to TRACE -->
97   <root level="{{ vitam_struct.log_level | default(vitam_defaults.services.log_
↪level) }}">
98     <!-- <appender-ref ref="STDOUT" /> -->
99     <appender-ref ref="FILE" />
100    <appender-ref ref="SYSLOG" />
101  </root>
102
103  <logger name="org.eclipse.jetty" level="WARN"/>
104  <logger name="fr.gouv.vitam.storage.engine.server.logbook.StorageLogbookMock" level=
↪"INFO"/>
105  <logger name="fr.gouv.vitam.metadata.core.graph.StoreGraphService" level="INFO"/>
106  <logger name="fr.gouv.vitam.metadata.core.graph.GraphComputeServiceImpl" level="INFO
↪"/>
107  <logger name="fr.gouv.vitam.common" level="WARN" />
108  {% if vitam_defaults.reconstruction.log_level is defined or vitam_struct.
↪reconstruction.log_level is defined %}
109    <logger name="fr.gouv.vitam.metadata.core.reconstruction.ReconstructionService"
↪
↪level="{{ vitam_struct.reconstruction.log_level |default(vitam_defaults.
↪reconstruction.log_level) }}" />
110    <logger name="fr.gouv.vitam.metadata.core.reconstruction.RestoreBackupService"
↪
↪level="{{ vitam_struct.reconstruction.log_level |default(vitam_defaults.
↪reconstruction.log_level) }}" />

```

(suite sur la page suivante)

(suite de la page précédente)

```

111 <logger name="fr.gouv.vitam.logbook.common.server.reconstruction.
    ↳ReconstructionService" level="{ { vitam_struct.reconstruction.log_level_
    ↳|default(vitam_defaults.reconstruction.log_level) } }"/>
112 <logger name="fr.gouv.vitam.logbook.common.server.reconstruction.
    ↳RestoreBackupService" level="{ { vitam_struct.reconstruction.log_level_
    ↳|default(vitam_defaults.reconstruction.log_level) } }"/>
113 <logger name="fr.gouv.vitam.functional.administration.common.impl.
    ↳ReconstructionServiceImpl" level="{ { vitam_struct.reconstruction.log_level_
    ↳|default(vitam_defaults.reconstruction.log_level) } }"/>
114 <logger name="fr.gouv.vitam.functional.administration.common.impl.
    ↳RestoreBackupServiceImpl" level="{ { vitam_struct.reconstruction.log_level_
    ↳|default(vitam_defaults.reconstruction.log_level) } }"/>
115 {% endif %}
116 {% if vitam_struct.performance_logger|lower == "true" %}
117 <logger name="fr.gouv.vitam.common.performance.PerformanceLogger" level="DEBUG"
    ↳additivity="false" >
118 <appender-ref ref="SYSLOG" />
119 </logger>
120 {% endif %}
121 <logger name="fr.gouv.vitam.common.alert.AlertServiceImpl" level="INFO">
122 <appender-ref ref="SECURITY" />
123 </logger>
124
125 {% if vitam_struct.vitam_component == 'storage' %}
126 <logger name="fr.gouv.vitam.storage.engine.server.offersynchronization" level="INFO
    ↳">
127 <appender-ref ref="OFFERSYNC" />
128 </logger>
129 {% endif %}
130
131 {% if vitam_struct.vitam_component == 'offer' %}
132 <logger name="fr.gouv.vitam.storage.offers.tape.process.ProcessExecutor" level="INFO
    ↳" additivity="false" >
133 <appender-ref ref="OFFER_TAPE" />
134 </logger>
135 {% endif %}
136
137
138 {% if vitam_struct.vitam_component == 'metadata' %}
139 <logger name="fr.gouv.vitam.metadata.core.migration" level="INFO"/>
140 {% endif %}
141
142 </configuration>

```

8.2.1.4 Fichier /vitam/conf/<composant>/jetty-config.xml

```

1 <?xml version="1.0"?>
2 <!DOCTYPE Configure PUBLIC "-//Jetty//Configure//EN" "http://www.eclipse.org/jetty/
    ↳configure_9_0.dtd">
3
4 <!-- ===== -->
5 <!-- Documentation of this file format can be found at: -->
6 <!-- http://wiki.eclipse.org/Jetty/Reference/jetty.xml_syntax -->
7 <!-- -->
8 <!-- Additional configuration files are available in $JETTY_HOME/etc -->

```

(suite sur la page suivante)

(suite de la page précédente)

```

9  <!-- and can be mixed in. See start.ini file for the default      -->
10 <!-- configuration files.                                         -->
11 <!--                                                             -->
12 <!-- For a description of the configuration mechanism, see the   -->
13 <!-- output of:                                                 -->
14 <!--   java -jar start.jar -?                                    -->
15 <!-- =====                                                  -->
16
17 <!-- =====                                                  -->
18 <!-- Configure a Jetty Server instance with an ID "Server"      -->
19 <!-- Other configuration files may also configure the "Server"  -->
20 <!-- ID, in which case they are adding configuration to the same -->
21 <!-- instance. If other configuration have a different ID, they -->
22 <!-- will create and configure another instance of Jetty.      -->
23 <!-- Consult the javadoc of o.e.j.server.Server for all        -->
24 <!-- configuration that may be set here.                        -->
25 <!-- =====                                                  -->
26 <Configure id="Server" class="org.eclipse.jetty.server.Server">
27
28
29     <!-- =====                                                  -->
30     <!-- Add shared Scheduler instance                          -->
31     <!-- =====                                                  -->
32     <Call name="addBean">
33         <Arg>
34             <New class="org.eclipse.jetty.util.thread.ScheduledExecutorScheduler"/>
35         </Arg>
36     </Call>
37
38     <!-- =====                                                  -->
39     <!-- Http Configuration.                                     -->
40     <!-- This is a common configuration instance used by all    -->
41     <!-- connectors that can carry HTTP semantics (HTTP, HTTPS, SPDY) -->
42     <!-- It configures the non wire protocol aspects of the HTTP -->
43     <!-- semantic.                                              -->
44     <!--                                                         -->
45     <!-- This configuration is only defined here and is used by -->
46     <!-- reference from the jetty-http.xml, jetty-https.xml and -->
47     <!-- jetty-spdy.xml configuration files which instantiate the -->
48     <!-- connectors.                                           -->
49     <!--                                                         -->
50     <!-- Consult the javadoc of o.e.j.server.HttpConfiguration -->
51     <!-- for all configuration that may be set here.          -->
52     <!-- =====                                                  -->
53     <New id="httpConfig" class="org.eclipse.jetty.server.HttpConfiguration">
54         <Set name="secureScheme">http</Set>
55         <Set name="securePort">8443</Set>
56         <Set name="outputBufferSize">32768</Set>
57         <Set name="requestHeaderSize">8192</Set>
58         <Set name="responseHeaderSize">8192</Set>
59         <Set name="sendServerVersion">false</Set>
60         <Set name="sendDateHeader">false</Set>
61         <Set name="headerCacheSize">512</Set>
62
63         <!-- Uncomment to enable handling of X-Forwarded- style headers
64         <Call name="addCustomizer">
65             <Arg><New class="org.eclipse.jetty.server.ForwardedRequestCustomizer"/></Arg>

```

↪Arg>

(suite sur la page suivante)

(suite de la page précédente)

```

66         </Call>
67         -->
68     </New>
69
70     <!-- ===== Original Connector ===== -
71     <!-- <Call name="addConnector">
72     <!--     <Arg>
73     <!--         <New class="org.eclipse.jetty.server.ServerConnector">
74     <!--             <Arg name="server"><Ref refid="Server" /></Arg>
75     <!--             <Arg name="factories">
76     <!--                 <Array type="org.eclipse.jetty.server.ConnectionFactory">
77     <!--                     <Item>
78     <!--                         <New class="org.eclipse.jetty.server.
79     <!--                         <HttpConnectionFactory">
80     <!--                             <Arg name="config"><Ref refid="httpConfig" /></
81     <!--                             <Arg>
82     <!--                             </New>
83     <!--                             </Item>
84     <!--                             </Array>
85     <!--                             </Arg>
86     <!--                             <Set name="port">{{ vitam_struct.port_service }}</Set>
87     <!--                             <Set name="idleTimeout">
88     <!--                             <Property name="http.timeout" default="{{ vitam_defaults.
89     <!--                             services.port_service_timeout }}"/>
90     <!--                             </Set>
91     <!--                             </New>
92     <!--                             </Arg>
93     <!-- </Call>
94
95     <!-- ===== -->
96     <!-- Set the default handler structure for the Server -->
97     <!-- A handler collection is used to pass received requests to -->
98     <!-- both the ContextHandlerCollection, which selects the next -->
99     <!-- handler by context path and virtual host, and the -->
100    <!-- DefaultHandler, which handles any requests not handled by -->
101    <!-- the context handlers. -->
    <!-- Other handlers may be added to the "Handlers" collection, -->

```

(suite sur la page suivante)

(suite de la page précédente)

```

102  <!-- for example the jetty-requestlog.xml file adds the      -->
103  <!-- RequestLogHandler after the default handler          -->
104  <!-- =====>
105  <Set name="handler">
106      <New id="Handlers" class="org.eclipse.jetty.server.handler.HandlerCollection">
107          <Set name="handlers">
108              <Array type="org.eclipse.jetty.server.Handler">
109                  <Item>
110                      <New id="Contexts" class="org.eclipse.jetty.server.handler.
↪ContextHandlerCollection"/>
111                  </Item>
112                  <Item>
113                      <New id="DefaultHandler" class="org.eclipse.jetty.server.
↪handler.DefaultHandler"/>
114                  </Item>
115              </Array>
116          </Set>
117      </New>
118  </Set>
119
120  <Set name="RequestLog">
121      <New id="RequestLogImpl" class="ch.qos.logback.access.jetty.RequestLogImpl
↪">
122          <Set name="fileName">{{vitam_folder_conf}}/logback-access.xml</Set>
123      </New>
124  </Set>
125  <Ref id="RequestLogImpl">
126      <Call name="start"/>
127  </Ref>
128
129  <!-- =====>
130  <!-- extra server options                                -->
131  <!-- =====>
132  <Set name="stopAtShutdown">true</Set>
133  <Set name="stopTimeout">5000</Set>
134  <Set name="dumpAfterStart">false</Set>
135  <Set name="dumpBeforeStop">false</Set>
136
137  {% if vitam_struct.https_enabled==true %}
138  <New id="httpsConfig" class="org.eclipse.jetty.server.HttpConfiguration">
139      <Set name="sendServerVersion">false</Set>
140      <Set name="sendDateHeader">false</Set>
141      <Call name="addCustomizer">
142          <Arg>
143              <New class="org.eclipse.jetty.server.SecureRequestCustomizer" />
144          </Arg>
145      </Call>
146  </New>
147  <New id="sslContextFactory" class="org.eclipse.jetty.util.ssl.
↪SslContextFactory$Server">
148      <Set name="KeyStorePath">{{ vitam_folder_conf }}/keystore_{{ vitam_struct.
↪vitam_component }}.jks</Set>
149      <Set name="KeyStorePassword">{{password_keystore}}</Set>
150      <Set name="KeyManagerPassword">{{password_manager_keystore}}</Set>
151      <Set name="TrustStorePath">{{ vitam_folder_conf }}/truststore_{{ vitam_struct.
↪vitam_component }}.jks</Set>
152      <Set name="TrustStorePassword">{{password_truststore}}</Set>

```

(suite sur la page suivante)

(suite de la page précédente)

```

153     <Set name="TrustStoreType">JKS</Set>
154     <Set name="NeedClientAuth">>false</Set>
155     <Set name="WantClientAuth">>true</Set>
156     <Set name="IncludeCipherSuites">
157         <Array type="String">
158             <Item>TLS_ECDHE.*</Item>
159             <Item>TLS_DHE_RSA.*</Item>
160         </Array>
161     </Set>
162     <Set name="IncludeProtocols">
163         <Array type="String">
164             <Item>TLSv1.2</Item>
165         </Array>
166     </Set>
167     <Set name="ExcludeCipherSuites">
168         <Array type="String">
169             <Item>.*NULL.*</Item>
170             <Item>.*RC4.*</Item>
171             <Item>.*MD5.*</Item>
172             <Item>.*DES.*</Item>
173             <Item>.*DSS.</Item>
174             <Item>TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA</Item>
175             <Item>TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA</Item>
176             <Item>TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA</Item>
177             <Item>TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA</Item>
178             <Item>TLS_DHE_RSA_WITH_AES_256_CBC_SHA</Item>
179             <Item>TLS_DHE_RSA_WITH_AES_128_CBC_SHA</Item>
180         </Array>
181     </Set>
182     <Set name="UseCipherSuitesOrder">>true</Set>
183     <Set name="RenegotiationAllowed">>true</Set>
184 </New>
185 <New id="sslConnectionFactory" class="org.eclipse.jetty.server.
→ SslConnectionFactory">
186     <Arg name="sslContextFactory">
187         <Ref refid="sslContextFactory" />
188     </Arg>
189     <Arg name="next">http/1.1</Arg>
190 </New>
191 <New id="businessConnector" class="org.eclipse.jetty.server.ServerConnector">
192     <Arg name="server">
193         <Ref refid="Server" />
194     </Arg>
195     <Arg name="factories">
196         <Array type="org.eclipse.jetty.server.ConnectionFactory">
197             <Item>
198                 <Ref refid="sslConnectionFactory" />
199             </Item>
200             <Item>
201                 <New class="org.eclipse.jetty.server.HttpConnectionFactory">
202                     <Arg name="config">
203                         <Ref refid="httpsConfig" />
204                     </Arg>
205                 </New>
206             </Item>
207         </Array>
208     </Arg>

```

(suite sur la page suivante)

(suite de la page précédente)

```

209     <Set name="host">{{ip_service}}</Set>
210     <Set name="port">
211         <SystemProperty name="jetty.port" default="{{ vitam_struct.port_service }}"
↪"/>
212     </Set>
213     <Set name="name">business</Set>
214 </New>
215
216 {% else %}
217
218     <!-- ===== -->
219     <!-- Connector for API business -->
220     <!-- Attach all ContextHanlder except Admin -->
221     <!-- ===== -->
222
223     <New id="businessConnector" class="org.eclipse.jetty.server.ServerConnector">
224         <Arg name="server"><Ref refid="Server" /></Arg>
225         <Arg name="factories">
226             <Array type="org.eclipse.jetty.server.ConnectionFactory">
227                 <Item>
228                     <New class="org.eclipse.jetty.server.HttpConnectionFactory">
229                         <Arg name="config"><Ref refid="httpConfig" /></Arg>
230                     </New>
231                 </Item>
232             </Array>
233         </Arg>
234         <Set name="host">{{ ip_service }}</Set>
235         <Set name="port">{{ vitam_struct.port_service }}</Set>
236         <Set name="name">business</Set>
237         <Set name="idleTimeout">
238             <Property name="http.timeout" default="{{ vitam_defaults.services.port_
↪service_timeout }}" />
239         </Set>
240     </New>
241
242 {% endif %}
243
244     <!-- ===== -->
245     <!-- Connector for API Admin -->
246     <!-- Attach all ContextHanlder -->
247     <!-- ===== -->
248
249     <New id="adminConnector" class="org.eclipse.jetty.server.ServerConnector">
250         <Arg name="server"><Ref refid="Server" /></Arg>
251         <Arg name="factories">
252             <Array type="org.eclipse.jetty.server.ConnectionFactory">
253                 <Item>
254                     <New class="org.eclipse.jetty.server.HttpConnectionFactory">
255                         <Arg name="config"><Ref refid="httpConfig" /></Arg>
256                     </New>
257                 </Item>
258             </Array>
259         </Arg>
260         <Set name="host">{{ ip_admin }}</Set>
261         <Set name="port">{{ vitam_struct.port_admin }}</Set>
262         <Set name="name">admin</Set>
263         <Set name="idleTimeout">

```

(suite sur la page suivante)

(suite de la page précédente)

```

264         <Property name="http.timeout" default="{ { vitam_defaults.services.port_
↪service_timeout } }"/>
265     </Set>
266 </New>
267
268
269
270
271     <Call name="setConnectors">
272         <Arg>
273             <Array type="org.eclipse.jetty.server.ServerConnector">
274                 <Item>
275                     <Ref refid="businessConnector" />
276                 </Item>
277                 <Item>
278                     <Ref refid="adminConnector" />
279                 </Item>
280             </Array>
281         </Arg>
282     </Call>
283
284 </Configure>

```

8.2.1.5 Fichier /vitam/conf/<composant>/logbook-client.conf

Ce fichier permet de configurer l'appel au composant logbook.

```

1 serverHost: { { vitam.logbook.host } }
2 serverPort: { { vitam.logbook.port_service } }

```

8.2.1.6 Fichier /vitam/conf/<composant>/server-identity.conf

```

1 identityName: { { ansible_nodename } }
2 identityRole: { { vitam_struct.vitam_composant } }
3 identitySiteId: { { vitam_site_id } }

```

8.2.1.7 Fichier /vitam/conf/<composant>/antisamy-esapi.xml

```

1 <?xml version="1.0" encoding="ISO-8859-1"?>
2
3 <!--
4 W3C rules retrieved from:
5 http://www.w3.org/TR/html401/struct/global.html
6 -->
7
8 <!--
9 Slashdot allowed tags taken from "Reply" page:
10 <b> <i> <p> <br> <a> <ol> <ul> <li> <dl> <dt> <dd> <em> <strong> <tt> <blockquote>
↪<div> <code> <quote>
11 -->
12

```

(suite sur la page suivante)

(suite de la page précédente)

```

13 <anti-samy-rules xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
14     xsi:noNamespaceSchemaLocation="antisamy.xsd">
15
16     <directives>
17         <directive name="omitXmlDeclaration" value="true"/>
18         <directive name="omitDoctypeDeclaration" value="true"/>
19         <directive name="maxInputSize" value="2000000"/>
20         <directive name="embedStyleSheets" value="false"/>
21     </directives>
22
23
24     <common-regexps>
25
26         <!--
27         From W3C:
28         This attribute assigns a class name or set of class names to an
29         element. Any number of elements may be assigned the same class
30         name or names. Multiple class names must be separated by white
31         space characters.
32         -->
33
34         <regexp name="htmlTitle" value="[a-zA-Z0-9\s_',';:\[\]!\./\\(\)]*" />
35         <!-- force non-empty with a '+' at the end instead of '*' -->
36         <regexp name="onsiteURL" value="([\w\\/\.\.?=&;\#~+|\\#(\w)+)"/>
37         <regexp name="offsiteURL" value="(\s)*((ht|f)tp(s?)://|mailto:)[A-Za-
38         <!-- z0-9]+[~a-zA-Z0-9-_\.\@#$$%&;:, \?=/\+!]*(\s)*"/>
39
40     </common-regexps>
41
42     <!--
43     Tag.name = a, b, div, body, etc.
44     Tag.action = filter: remove tags, but keep content, validate: keep content as
45     <!-- long as it passes rules, remove: remove tag and contents
46     Attribute.name = id, class, href, align, width, etc.
47     Attribute.onInvalid = what to do when the attribute is invalid, e.g., remove
48     <!-- the tag (removeTag), remove the attribute (removeAttribute), filter the tag
49     <!-- (filterTag)
50     Attribute.description = What rules in English you want to tell the users they
51     <!-- can have for this attribute. Include helpful things so they'll be able to tune
52     <!-- their HTML
53
54     -->
55
56     <!--
57     Some attributes are common to all (or most) HTML tags. There aren't many that
58     <!-- qualify for this. You have to make sure there's no
59     <!-- collisions between any of these attribute names with attribute names of other
60     <!-- tags that are for different purposes.
61
62     -->
63
64     <common-attributes>
65
66         <attribute name="lang" description="The 'lang' attribute tells the
67         <!-- browser what language the element's attribute values and content are written in">
68             <regexp-list>

```

(suite sur la page suivante)

(suite de la page précédente)

```

60         <regexp value="[a-zA-Z]{2,20}" />
61     </regexp-list>
62 </attribute>
63
64     <attribute name="title" description="The 'title' attribute provides
65 ↪ text that shows up in a 'tooltip' when a user hovers their mouse over the element">
66         <regexp-list>
67             <regexp name="htmlTitle" />
68         </regexp-list>
69     </attribute>
70
71     <attribute name="href" onInvalid="filterTag">
72         <regexp-list>
73             <regexp name="onsiteURL" />
74             <regexp name="offsiteURL" />
75         </regexp-list>
76     </attribute>
77
78     <attribute name="align" description="The 'align' attribute of an HTML
79 ↪ element is a direction word, like 'left', 'right' or 'center'">
80         <literal-list>
81             <literal value="center" />
82             <literal value="left" />
83             <literal value="right" />
84             <literal value="justify" />
85             <literal value="char" />
86         </literal-list>
87     </attribute>
88
89 </common-attributes>
90
91 <!--
92 ↪ This requires normal updates as browsers continue to diverge from the W3C and
93 ↪ each other. As long as the browser wars continue
94 ↪ this is going to continue. I'm not sure war is the right word for what's
95 ↪ going on. Doesn't somebody have to win a war after
96 ↪ a while?
97     -->
98
99 <global-tag-attributes>
100     <attribute name="title" />
101     <attribute name="lang" />
102 </global-tag-attributes>
103
104 <tag-rules>
105
106     <!-- Tags related to JavaScript -->
107
108     <tag name="script" action="remove" />
109     <tag name="noscript" action="remove" />
110
111     <!-- Frame & related tags -->
112
113     <tag name="iframe" action="remove" />
114     <tag name="frameset" action="remove" />

```

(suite sur la page suivante)

(suite de la page précédente)

```

113     <tag name="frame" action="remove"/>
114     <tag name="noframes" action="remove"/>
115
116
117     <!-- All reasonable formatting tags -->
118
119     <tag name="p" action="validate">
120         <attribute name="align"/>
121     </tag>
122
123     <tag name="div" action="validate"/>
124     <tag name="i" action="validate"/>
125     <tag name="b" action="validate"/>
126     <tag name="em" action="validate"/>
127     <tag name="blockquote" action="validate"/>
128     <tag name="tt" action="validate"/>
129
130     <tag name="br" action="truncate"/>
131
132     <!-- Custom Slashdot tags, though we're trimming the idea of having a
133     ↪possible mismatching end tag with the endtag="" attribute -->
134
135     <tag name="quote" action="validate"/>
136     <tag name="ecode" action="validate"/>
137
138
139     <!-- Anchor and anchor related tags -->
140
141     <tag name="a" action="validate">
142         <attribute name="href" onInvalid="filterTag"/>
143         <attribute name="nohref">
144             <literal-list>
145                 <literal value="nohref"/>
146                 <literal value=""/>
147             </literal-list>
148         </attribute>
149         <attribute name="rel">
150             <literal-list>
151                 <literal value="nofollow"/>
152             </literal-list>
153         </attribute>
154     </tag>
155
156     <!-- List tags -->
157
158     <tag name="ul" action="validate"/>
159     <tag name="ol" action="validate"/>
160     <tag name="li" action="validate"/>
161
162 </tag-rules>
163
164
165
166     <!-- No CSS on Slashdot posts -->
167
168 <css-rules>

```

(suite sur la page suivante)

(suite de la page précédente)

```

169     </css-rules>
170
171
172     <html-entities>
173         <entity name="amp" cdata="&amp;"/>
174         <entity name="nbsp" cdata="&amp;#160;"/>
175
176         <entity name="iexcl" cdata="&amp;#161;"/> <!--inverted exclamation_
177 ↪mark, U+00A1 ISONum -->
178         <entity name="cent" cdata="&amp;#162;"/> <!--cent sign, U+00A2 ISONum_
179 ↪-->
180         <entity name="pound" cdata="&amp;#163;"/> <!--pound sign, U+00A3_
181 ↪ISONum -->
182         <entity name="curren" cdata="&amp;#164;"/> <!--currency sign, U+00A4_
183 ↪ISONum -->
184         <entity name="yen" cdata="&amp;#165;"/> <!--yen sign = yuan sign, _
185 ↪U+00A5 ISONum -->
186         <entity name="brvbar" cdata="&amp;#166;"/> <!--broken bar = broken_
187 ↪vertical bar, U+00A6 ISONum -->
188         <entity name="sect" cdata="&amp;#167;"/> <!--section sign, U+00A7_
189 ↪ISONum -->
190         <entity name="uml" cdata="&amp;#168;"/> <!--diaeresis = spacing_
191 ↪diaeresis, U+00A8 ISODia -->
192         <entity name="copy" cdata="&amp;#169;"/> <!--copyright sign, U+00A9_
193 ↪ISONum -->
194         <entity name="ordf" cdata="&amp;#170;"/> <!--feminine ordinal_
195 ↪indicator, U+00AA ISONum -->
196         <entity name="laquo" cdata="&amp;#171;"/> <!--left-pointing double_
197 ↪angle quotation mark = left pointing guillemet, U+00AB ISONum -->
198         <entity name="not" cdata="&amp;#172;"/> <!--not sign, U+00AC ISONum --
199 ↪>
200         <entity name="shy" cdata="&amp;#173;"/> <!--soft hyphen = _
201 ↪discretionary hyphen, U+00AD ISONum -->
202         <entity name="reg" cdata="&amp;#174;"/> <!--registered sign = _
203 ↪registered trade mark sign, U+00AE ISONum -->
204         <entity name="macr" cdata="&amp;#175;"/> <!--macron = spacing macron_
205 ↪= overline = APL overbar, U+00AF ISODia -->
206         <entity name="deg" cdata="&amp;#176;"/> <!--degree sign, U+00B0_
207 ↪ISONum -->
208         <entity name="plusmn" cdata="&amp;#177;"/> <!--plus-minus sign = plus-
209 ↪or-minus sign, U+00B1 ISONum -->
210         <entity name="sup2" cdata="&amp;#178;"/> <!--superscript two = _
211 ↪superscript digit two = squared, U+00B2 ISONum -->
212         <entity name="sup3" cdata="&amp;#179;"/> <!--superscript three = _
213 ↪superscript digit three= cubed, U+00B3 ISONum -->
214         <entity name="acute" cdata="&amp;#180;"/> <!--acute accent = spacing_
215 ↪acute, U+00B4 ISODia -->
216         <entity name="micro" cdata="&amp;#181;"/> <!--micro sign, U+00B5_
217 ↪ISONum -->
218         <entity name="para" cdata="&amp;#182;"/> <!--pilcrow sign = paragraph_
219 ↪sign, U+00B6 ISONum -->
220         <entity name="middot" cdata="&amp;#183;"/> <!--middle dot = Georgian_
221 ↪comma = Greek middle dot, U+00B7 ISONum -->
222         <entity name="cedil" cdata="&amp;#184;"/> <!--cedilla = spacing_
223 ↪cedilla, U+00B8 ISODia -->
224         <entity name="sup1" cdata="&amp;#185;"/> <!--superscript one = _
225 ↪superscript digit one, U+00B9 ISONum -->

```

(suite sur la page suivante)

(suite de la page précédente)

```

201      <entity name="ordm" cdata="&#186;"/> <!--masculine ordinal_
↳indicator, U+00BA ISONum -->
202      <entity name="raquo" cdata="&#187;"/> <!--right-pointing double_
↳angle quotation mark = right pointing guillemet, U+00BB ISONum -->
203      <entity name="frac14" cdata="&#188;"/> <!--vulgar fraction one_
↳quarter = fraction one quarter, U+00BC ISONum -->
204      <entity name="frac12" cdata="&#189;"/> <!--vulgar fraction one_
↳half = fraction one half, U+00BD ISONum -->
205      <entity name="frac34" cdata="&#190;"/> <!--vulgar fraction three_
↳quarters = fraction three quarters, U+00BE ISONum -->
206      <entity name="quest" cdata="&#191;"/> <!--inverted question mark_
↳= turned question mark, U+00BF ISONum -->
207      <entity name="Agrave" cdata="&#192;"/> <!--latin capital letter A_
↳with grave = latin capital letter A grave, U+00C0 ISOLat1 -->
208      <entity name="Aacute" cdata="&#193;"/> <!--latin capital letter A_
↳with acute, U+00C1 ISOLat1 -->
209      <entity name="Acirc" cdata="&#194;"/> <!--latin capital letter A_
↳with circumflex, U+00C2 ISOLat1 -->
210      <entity name="Atilde" cdata="&#195;"/> <!--latin capital letter A_
↳with tilde, U+00C3 ISOLat1 -->
211      <entity name="Auml" cdata="&#196;"/> <!--latin capital letter A_
↳with diaeresis, U+00C4 ISOLat1 -->
212      <entity name="Aring" cdata="&#197;"/> <!--latin capital letter A_
↳with ring above = latin capital letter A ring, U+00C5 ISOLat1 -->
213      <entity name="Aelig" cdata="&#198;"/> <!--latin capital letter AE_
↳= latin capital ligature AE, U+00C6 ISOLat1 -->
214      <entity name="Ccedil" cdata="&#199;"/> <!--latin capital letter C_
↳with cedilla, U+00C7 ISOLat1 -->
215      <entity name="Egrave" cdata="&#200;"/> <!--latin capital letter E_
↳with grave, U+00C8 ISOLat1 -->
216      <entity name="Eacute" cdata="&#201;"/> <!--latin capital letter E_
↳with acute, U+00C9 ISOLat1 -->
217      <entity name="Ecirc" cdata="&#202;"/> <!--latin capital letter E_
↳with circumflex, U+00CA ISOLat1 -->
218      <entity name="Euml" cdata="&#203;"/> <!--latin capital letter E_
↳with diaeresis, U+00CB ISOLat1 -->
219      <entity name="Igrave" cdata="&#204;"/> <!--latin capital letter I_
↳with grave, U+00CC ISOLat1 -->
220      <entity name="Iacute" cdata="&#205;"/> <!--latin capital letter I_
↳with acute, U+00CD ISOLat1 -->
221      <entity name="Icirc" cdata="&#206;"/> <!--latin capital letter I_
↳with circumflex, U+00CE ISOLat1 -->
222      <entity name="Iuml" cdata="&#207;"/> <!--latin capital letter I_
↳with diaeresis, U+00CF ISOLat1 -->
223      <entity name="ETH" cdata="&#208;"/> <!--latin capital letter ETH, _
↳U+00D0 ISOLat1 -->
224      <entity name="Ntilde" cdata="&#209;"/> <!--latin capital letter N_
↳with tilde, U+00D1 ISOLat1 -->
225      <entity name="Ograve" cdata="&#210;"/> <!--latin capital letter O_
↳with grave, U+00D2 ISOLat1 -->
226      <entity name="Oacute" cdata="&#211;"/> <!--latin capital letter O_
↳with acute, U+00D3 ISOLat1 -->
227      <entity name="Ocirc" cdata="&#212;"/> <!--latin capital letter O_
↳with circumflex, U+00D4 ISOLat1 -->
228      <entity name="Otilde" cdata="&#213;"/> <!--latin capital letter O_
↳with tilde, U+00D5 ISOLat1 -->
229      <entity name="Ouml" cdata="&#214;"/> <!--latin capital letter O_
↳with diaeresis, U+00D6 ISOLat1 -->

```

(suite sur la page suivante)

(suite de la page précédente)

```

230      <entity name="times" cdata="&#215;"/> <!--multiplication sign,
↳U+00D7 ISOnum -->
231      <entity name="Oslash" cdata="&#216;"/> <!--latin capital letter O
↳with stroke = latin capital letter O slash, U+00D8 ISolat1 -->
232      <entity name="Ugrave" cdata="&#217;"/> <!--latin capital letter U
↳with grave, U+00D9 ISolat1 -->
233      <entity name="Uacute" cdata="&#218;"/> <!--latin capital letter U
↳with acute, U+00DA ISolat1 -->
234      <entity name="Ucirc" cdata="&#219;"/> <!--latin capital letter U
↳with circumflex, U+00DB ISolat1 -->
235      <entity name="Uuml" cdata="&#220;"/> <!--latin capital letter U
↳with diaeresis, U+00DC ISolat1 -->
236      <entity name="Yacute" cdata="&#221;"/> <!--latin capital letter Y
↳with acute, U+00DD ISolat1 -->
237      <entity name="THORN" cdata="&#222;"/> <!--latin capital letter
↳THORN, U+00DE ISolat1 -->
238      <entity name="szlig" cdata="&#223;"/> <!--latin small letter
↳sharp s = ess-zed, U+00DF ISolat1 -->
239      <entity name="agrave" cdata="&#224;"/> <!--latin small letter a
↳with grave = latin small letter a grave, U+00E0 ISolat1 -->
240      <entity name="aacute" cdata="&#225;"/> <!--latin small letter a
↳with acute, U+00E1 ISolat1 -->
241      <entity name="acirc" cdata="&#226;"/> <!--latin small letter a
↳with circumflex, U+00E2 ISolat1 -->
242      <entity name="atilde" cdata="&#227;"/> <!--latin small letter a
↳with tilde, U+00E3 ISolat1 -->
243      <entity name="auml" cdata="&#228;"/> <!--latin small letter a
↳with diaeresis, U+00E4 ISolat1 -->
244      <entity name="aring" cdata="&#229;"/> <!--latin small letter a
↳with ring above = latin small letter a ring, U+00E5 ISolat1 -->
245      <entity name="aelig" cdata="&#230;"/> <!--latin small letter ae =
↳latin small ligature ae, U+00E6 ISolat1 -->
246      <entity name="ccedil" cdata="&#231;"/> <!--latin small letter c
↳with cedilla, U+00E7 ISolat1 -->
247      <entity name="egrave" cdata="&#232;"/> <!--latin small letter e
↳with grave, U+00E8 ISolat1 -->
248      <entity name="eacute" cdata="&#233;"/> <!--latin small letter e
↳with acute, U+00E9 ISolat1 -->
249      <entity name="ecirc" cdata="&#234;"/> <!--latin small letter e
↳with circumflex, U+00EA ISolat1 -->
250      <entity name="euml" cdata="&#235;"/> <!--latin small letter e
↳with diaeresis, U+00EB ISolat1 -->
251      <entity name="igrave" cdata="&#236;"/> <!--latin small letter i
↳with grave, U+00EC ISolat1 -->
252      <entity name="iacute" cdata="&#237;"/> <!--latin small letter i
↳with acute, U+00ED ISolat1 -->
253      <entity name="icirc" cdata="&#238;"/> <!--latin small letter i
↳with circumflex, U+00EE ISolat1 -->
254      <entity name="iuml" cdata="&#239;"/> <!--latin small letter i
↳with diaeresis, U+00EF ISolat1 -->
255      <entity name="eth" cdata="&#240;"/> <!--latin small letter eth,
↳U+00F0 ISolat1 -->
256      <entity name="ntilde" cdata="&#241;"/> <!--latin small letter n
↳with tilde, U+00F1 ISolat1 -->
257      <entity name="ograve" cdata="&#242;"/> <!--latin small letter o
↳with grave, U+00F2 ISolat1 -->
258      <entity name="oacute" cdata="&#243;"/> <!--latin small letter o
↳with acute, U+00F3 ISolat1 -->

```

(suite sur la page suivante)

(suite de la page précédente)

```

259         <entity name="ocirc " cdata="&#244;"/> <!--latin small letter o_
↳with circumflex, U+00F4 ISolat1 -->
260         <entity name="otilde" cdata="&#245;"/> <!--latin small letter o_
↳with tilde, U+00F5 ISolat1 -->
261         <entity name="ouml" cdata="&#246;"/> <!--latin small letter o_
↳with diaeresis, U+00F6 ISolat1 -->
262         <entity name="divide" cdata="&#247;"/> <!--division sign, U+00F7_
↳ISOnum -->
263         <entity name="oslash" cdata="&#248;"/> <!--latin small letter o_
↳with stroke, = latin small letter o slash, U+00F8 ISolat1 -->
264         <entity name="ugrave" cdata="&#249;"/> <!--latin small letter u_
↳with grave, U+00F9 ISolat1 -->
265         <entity name="uacute" cdata="&#250;"/> <!--latin small letter u_
↳with acute, U+00FA ISolat1 -->
266         <entity name="ucirc" cdata="&#251;"/> <!--latin small letter u_
↳with circumflex, U+00FB ISolat1 -->
267         <entity name="uuml" cdata="&#252;"/> <!--latin small letter u_
↳with diaeresis, U+00FC ISolat1 -->
268         <entity name="yacute" cdata="&#253;"/> <!--latin small letter y_
↳with acute, U+00FD ISolat1 -->
269         <entity name="thorn" cdata="&#254;"/> <!--latin small letter_
↳thorn, U+00FE ISolat1 -->
270         <entity name="yuml" cdata="&#255;"/> <!--latin small letter y_
↳with diaeresis, U+00FF ISolat1 -->
271
272         <entity name="fnof" cdata="&#402;"/> <!--latin small f with hook_
↳= function = florin, U+0192 ISOtech -->
273
274         <!-- Greek -->
275         <entity name="Alpha" cdata="&#913;"/> <!--greek capital letter_
↳alpha, U+0391 -->
276         <entity name="Beta" cdata="&#914;"/> <!--greek capital letter_
↳beta, U+0392 -->
277         <entity name="Gamma" cdata="&#915;"/> <!--greek capital letter_
↳gamma, U+0393 ISOgrk3 -->
278         <entity name="Delta" cdata="&#916;"/> <!--greek capital letter_
↳delta, U+0394 ISOgrk3 -->
279         <entity name="Epsilon" cdata="&#917;"/> <!--greek capital letter_
↳epsilon, U+0395 -->
280         <entity name="Zeta" cdata="&#918;"/> <!--greek capital letter_
↳zeta, U+0396 -->
281         <entity name="Eta" cdata="&#919;"/> <!--greek capital letter eta,
↳U+0397 -->
282         <entity name="Theta" cdata="&#920;"/> <!--greek capital letter_
↳theta, U+0398 ISOgrk3 -->
283         <entity name="Iota" cdata="&#921;"/> <!--greek capital letter_
↳iota, U+0399 -->
284         <entity name="Kappa" cdata="&#922;"/> <!--greek capital letter_
↳kappa, U+039A -->
285         <entity name="Lambda" cdata="&#923;"/> <!--greek capital letter_
↳lambda, U+039B ISOgrk3 -->
286         <entity name="Mu" cdata="&#924;"/> <!--greek capital letter mu,
↳U+039C -->
287         <entity name="Nu" cdata="&#925;"/> <!--greek capital letter nu,
↳U+039D -->
288         <entity name="Xi" cdata="&#926;"/> <!--greek capital letter xi,
↳U+039E ISOgrk3 -->

```

(suite sur la page suivante)

(suite de la page précédente)

```

289      <entity name="Omicron" cdata="&#927;"/> <!--greek capital letter_
↳ omicron, U+039F -->
290      <entity name="Pi" cdata="&#928;"/> <!--greek capital letter pi,
↳ U+03A0 ISOgrk3 -->
291      <entity name="Rho" cdata="&#929;"/> <!--greek capital letter rho,
↳ U+03A1 -->
292      <!-- there is no Sigmaf, and no U+03A2 character either -->
293      <entity name="Sigma" cdata="&#931;"/> <!--greek capital letter_
↳ sigma, U+03A3 ISOgrk3 -->
294      <entity name="Tau" cdata="&#932;"/> <!--greek capital letter tau,
↳ U+03A4 -->
295      <entity name="Upsilon" cdata="&#933;"/> <!--greek capital letter_
↳ upsilon, U+03A5 ISOgrk3 -->
296      <entity name="Phi" cdata="&#934;"/> <!--greek capital letter phi,
↳ U+03A6 ISOgrk3 -->
297      <entity name="Chi" cdata="&#935;"/> <!--greek capital letter chi,
↳ U+03A7 -->
298      <entity name="Psi" cdata="&#936;"/> <!--greek capital letter psi,
↳ U+03A8 ISOgrk3 -->
299      <entity name="Omega" cdata="&#937;"/> <!--greek capital letter_
↳ omega, U+03A9 ISOgrk3 -->
300
301      <entity name="alpha" cdata="&#945;"/> <!--greek small letter_
↳ alpha, U+03B1 ISOgrk3 -->
302      <entity name="beta" cdata="&#946;"/> <!--greek small letter beta,
↳ U+03B2 ISOgrk3 -->
303      <entity name="gamma" cdata="&#947;"/> <!--greek small letter_
↳ gamma, U+03B3 ISOgrk3 -->
304      <entity name="delta" cdata="&#948;"/> <!--greek small letter_
↳ delta, U+03B4 ISOgrk3 -->
305      <entity name="epsilon" cdata="&#949;"/> <!--greek small letter_
↳ epsilon, U+03B5 ISOgrk3 -->
306      <entity name="zeta" cdata="&#950;"/> <!--greek small letter zeta,
↳ U+03B6 ISOgrk3 -->
307      <entity name="eta" cdata="&#951;"/> <!--greek small letter eta,
↳ U+03B7 ISOgrk3 -->
308      <entity name="theta" cdata="&#952;"/> <!--greek small letter_
↳ theta, U+03B8 ISOgrk3 -->
309      <entity name="iota" cdata="&#953;"/> <!--greek small letter iota,
↳ U+03B9 ISOgrk3 -->
310      <entity name="kappa" cdata="&#954;"/> <!--greek small letter_
↳ kappa, U+03BA ISOgrk3 -->
311      <entity name="lambda" cdata="&#955;"/> <!--greek small letter_
↳ lambda, U+03BB ISOgrk3 -->
312      <entity name="mu" cdata="&#956;"/> <!--greek small letter mu,
↳ U+03BC ISOgrk3 -->
313      <entity name="nu" cdata="&#957;"/> <!--greek small letter nu,
↳ U+03BD ISOgrk3 -->
314      <entity name="xi" cdata="&#958;"/> <!--greek small letter xi,
↳ U+03BE ISOgrk3 -->
315      <entity name="omicron" cdata="&#959;"/> <!--greek small letter_
↳ omicron, U+03BF NEW -->
316      <entity name="pi" cdata="&#960;"/> <!--greek small letter pi,
↳ U+03C0 ISOgrk3 -->
317      <entity name="rho" cdata="&#961;"/> <!--greek small letter rho,
↳ U+03C1 ISOgrk3 -->
318      <entity name="sigmaf" cdata="&#962;"/> <!--greek small letter_
↳ final sigma, U+03C2 ISOgrk3 -->

```

(suite sur la page suivante)

(suite de la page précédente)

```

319         <entity name="sigma" cdata="&#963;"/> <!--greek small letter_
↳sigma, U+03C3 ISOgrk3 -->
320         <entity name="tau" cdata="&#964;"/> <!--greek small letter tau,
↳U+03C4 ISOgrk3 -->
321         <entity name="upsilon" cdata="&#965;"/> <!--greek small letter_
↳upsilon, U+03C5 ISOgrk3 -->
322         <entity name="phi" cdata="&#966;"/> <!--greek small letter phi,
↳U+03C6 ISOgrk3 -->
323         <entity name="chi" cdata="&#967;"/> <!--greek small letter chi,
↳U+03C7 ISOgrk3 -->
324         <entity name="psi" cdata="&#968;"/> <!--greek small letter psi,
↳U+03C8 ISOgrk3 -->
325         <entity name="omega" cdata="&#969;"/> <!--greek small letter_
↳omega, U+03C9 ISOgrk3 -->
326         <entity name="thetasym" cdata="&#977;"/> <!--greek small letter_
↳theta symbol, U+03D1 NEW -->
327         <entity name="upsih" cdata="&#978;"/> <!--greek upsilon with hook_
↳symbol, U+03D2 NEW -->
328         <entity name="piv" cdata="&#982;"/> <!--greek pi symbol, U+03D6_
↳ISOgrk3 -->
329
330         <!-- General Punctuation -->
331         <entity name="bull" cdata="&#8226;"/> <!--bullet = black small_
↳circle, U+2022 ISOpub -->
332         <!-- bullet is NOT the same as bullet operator, U+2219 -->
333         <entity name="hellip" cdata="&#8230;"/> <!--horizontal ellipsis =
↳three dot leader, U+2026 ISOpub -->
334         <entity name="prime" cdata="&#8242;"/> <!--prime = minutes = feet,
↳ U+2032 ISOftech -->
335         <entity name="Prime" cdata="&#8243;"/> <!--double prime = seconds_
↳= inches, U+2033 ISOftech -->
336         <entity name="oline" cdata="&#8254;"/> <!--overline = spacing_
↳overscore, U+203E NEW -->
337         <entity name="frasl" cdata="&#8260;"/> <!--fraction slash, U+2044_
↳NEW -->
338
339         <!-- Letterlike Symbols -->
340         <entity name="weierp" cdata="&#8472;"/> <!--script capital P =
↳power set = Weierstrass p, U+2118 ISOamso -->
341         <entity name="image" cdata="&#8465;"/> <!--blackletter capital I_
↳= imaginary part, U+2111 ISOamso -->
342         <entity name="real" cdata="&#8476;"/> <!--blackletter capital R =
↳real part symbol, U+211C ISOamso -->
343         <entity name="trade" cdata="&#8482;"/> <!--trade mark sign,
↳U+2122 ISOnum -->
344         <entity name="alefsym" cdata="&#8501;"/> <!--alef symbol = first_
↳transfinite cardinal, U+2135 NEW -->
345         <!-- alef symbol is NOT the same as hebrew letter alef,
346         U+05D0 although the same glyph could be used to depict both_
↳characters -->
347
348         <!-- Arrows -->
349         <entity name="larr" cdata="&#8592;"/> <!--leftwards arrow, U+2190_
↳ISOnum -->
350         <entity name="uarr" cdata="&#8593;"/> <!--upwards arrow, U+2191_
↳ISOnum-->
351         <entity name="rarr" cdata="&#8594;"/> <!--rightwards arrow,
↳U+2192 ISOnum -->

```

(suite sur la page suivante)

(suite de la page précédente)

```

352      <entity name="darr" cdata="&#8595;"/> <!--downwards arrow, U+2193_
↳ ISOnum -->
353      <entity name="harr" cdata="&#8596;"/> <!--left right arrow,
↳ U+2194 ISOamsa -->
354      <entity name="crarr" cdata="&#8629;"/> <!--downwards arrow with_
↳ corner leftwards
355      = carriage return, U+21B5 NEW -->
356      <entity name="lArr" cdata="&#8656;"/> <!--leftwards double arrow,
↳ U+21D0 ISotech -->
357
358      <!-- ISO 10646 does not say that lArr is the same as the 'is implied_
↳ by' arrow
359      but also does not have any other character for that function. So ?
↳ lArr can
360      be used for 'is implied by' as ISotech suggests -->
361
362      <entity name="uArr" cdata="&#8657;"/> <!--upwards double arrow,
↳ U+21D1 ISOamsa -->
363      <entity name="rArr" cdata="&#8658;"/> <!--rightwards double arrow,
↳ U+21D2 ISotech -->
364
365      <!-- ISO 10646 does not say this is the 'implies' character but does_
↳ not have
366      another character with this function so ?
367      rArr can be used for 'implies' as ISotech suggests -->
368
369      <entity name="dArr" cdata="&#8659;"/> <!--downwards double arrow,
↳ U+21D3 ISOamsa -->
370      <entity name="hArr" cdata="&#8660;"/> <!--left right double arrow,
↳ U+21D4 ISOamsa -->
371
372      <!-- Mathematical Operators -->
373      <entity name="forall" cdata="&#8704;"/> <!--for all, U+2200_
↳ ISotech -->
374      <entity name="part" cdata="&#8706;"/> <!--partial differential,
↳ U+2202 ISotech -->
375      <entity name="exist" cdata="&#8707;"/> <!--there exists, U+2203_
↳ ISotech -->
376      <entity name="empty" cdata="&#8709;"/> <!--empty set = null set =
↳ diameter,U+2205 ISOamsa -->
377      <entity name="nabla" cdata="&#8711;"/> <!--nabla = backward_
↳ difference, U+2207 ISotech -->
378      <entity name="isin" cdata="&#8712;"/> <!--element of, U+2208_
↳ ISotech -->
379      <entity name="notin" cdata="&#8713;"/> <!--not an element of,
↳ U+2209 ISotech -->
380      <entity name="ni" cdata="&#8715;"/> <!--contains as member,
↳ U+220B ISotech -->
381
382      <!-- should there be a more memorable name than 'ni'? -->
383      <entity name="prod" cdata="&#8719;"/> <!--n-ary product = product_
↳ sign, U+220F ISOamsb -->
384
385      <!-- prod is NOT the same character as U+03A0 'greek capital letter pi
↳ ' though
386      the same glyph might be used for both -->
387

```

(suite sur la page suivante)

(suite de la page précédente)

```

388      <entity name="sum" cdata="&#8721;"/> <!--n-ary sumation, U+2211_
↪ISOamsb -->
389
390      <!-- sum is NOT the same character as U+03A3 'greek capital letter_
↪sigma'
391          though the same glyph might be used for both -->
392
393      <entity name="minus" cdata="&#8722;"/> <!--minus sign, U+2212_
↪ISOftech -->
394      <entity name="lowast" cdata="&#8727;"/> <!--asterisk operator, _
↪U+2217 ISOftech -->
395      <entity name="radic" cdata="&#8730;"/> <!--square root = radical_
↪sign, U+221A ISOftech -->
396      <entity name="prop" cdata="&#8733;"/> <!--proportional to, U+221D_
↪ISOftech -->
397      <entity name="infin" cdata="&#8734;"/> <!--infinity, U+221E_
↪ISOftech -->
398      <entity name="ang" cdata="&#8736;"/> <!--angle, U+2220 ISOams_ -->
399      <entity name="and" cdata="&#8743;"/> <!--logical and = wedge, _
↪U+2227 ISOftech -->
400      <entity name="or" cdata="&#8744;"/> <!--logical or = vee, U+2228_
↪ISOftech -->
401      <entity name="cap" cdata="&#8745;"/> <!--intersection = cap, _
↪U+2229 ISOftech -->
402      <entity name="cup" cdata="&#8746;"/> <!--union = cup, U+222A_
↪ISOftech -->
403      <entity name="int" cdata="&#8747;"/> <!--integral, U+222B ISOftech_
↪-->
404      <entity name="there4" cdata="&#8756;"/> <!--therefore, U+2234_
↪ISOftech -->
405      <entity name="sim" cdata="&#8764;"/> <!--tilde operator = varies_
↪with = similar to, U+223C ISOftech -->
406
407      <!-- tilde operator is NOT the same character as the tilde, U+007E,
408          although the same glyph might be used to represent both -->
409
410      <entity name="cong" cdata="&#8773;"/> <!--approximately equal to, _
↪U+2245 ISOftech -->
411      <entity name="asymp" cdata="&#8776;"/> <!--almost equal to = _
↪asymptotic to, U+2248 ISOamsr -->
412      <entity name="ne" cdata="&#8800;"/> <!--not equal to, U+2260_
↪ISOftech -->
413      <entity name="equiv" cdata="&#8801;"/> <!--identical to, U+2261_
↪ISOftech -->
414      <entity name="le" cdata="&#8804;"/> <!--less-than or equal to, _
↪U+2264 ISOftech -->
415      <entity name="ge" cdata="&#8805;"/> <!--greater-than or equal to, _
↪U+2265 ISOftech -->
416      <entity name="sub" cdata="&#8834;"/> <!--subset of, U+2282_
↪ISOftech -->
417      <entity name="sup" cdata="&#8835;"/> <!--superset of, U+2283_
↪ISOftech -->
418
419      <!-- note that nsup, 'not a superset of, U+2283' is not covered by_
↪the Symbol
420          font encoding and is not included. Should it be, for symmetry?
421          It is in ISOamsn -->

```

(suite sur la page suivante)

(suite de la page précédente)

```

422         <entity name="nsup" cdata="&#8836;"/> <!--not a subset of, U+2284_
423 ↪ ISOamsn -->
424         <entity name="sube" cdata="&#8838;"/> <!--subset of or equal to, _
425 ↪ U+2286 ISOtech -->
426         <entity name="supe" cdata="&#8839;"/> <!--superset of or equal to,
427 ↪ U+2287 ISOtech -->
428         <entity name="oplus" cdata="&#8853;"/> <!--circled plus = direct_
429 ↪ sum, U+2295 ISOamsb -->
430         <entity name="otimes" cdata="&#8855;"/> <!--circled times = _
431 ↪ vector product, U+2297 ISOamsb -->
432         <entity name="perp" cdata="&#8869;"/> <!--up tack = orthogonal to, _
433 ↪ = perpendicular, U+22A5 ISOtech -->
434         <entity name="sdot" cdata="&#8901;"/> <!--dot operator, U+22C5_
435 ↪ ISOamsb -->
436         <!-- dot operator is NOT the same character as U+00B7 middle dot -->
437
438         <!-- Miscellaneous Technical -->
439         <entity name="lceil" cdata="&#8968;"/> <!--left ceiling = apl_
440 ↪ upstile, U+2308 ISOamsc -->
441         <entity name="rceil" cdata="&#8969;"/> <!--right ceiling, U+2309_
442 ↪ ISOamsc -->
443         <entity name="lfloor" cdata="&#8970;"/> <!--left floor = apl_
444 ↪ downstile, U+230A ISOamsc -->
445         <entity name="rfloor" cdata="&#8971;"/> <!--right floor, U+230B_
446 ↪ ISOamsc -->
447         <entity name="lang" cdata="&#9001;"/> <!--left-pointing angle_
448 ↪ bracket = bra, U+2329 ISOtech -->
449         <!-- lang is NOT the same character as U+003C 'less than'
450         or U+2039 'single left-pointing angle quotation mark' -->
451         <entity name="rang" cdata="&#9002;"/> <!--right-pointing angle_
452 ↪ bracket = ket, U+232A ISOtech -->
453         <!-- rang is NOT the same character as U+003E 'greater than' or_
454 ↪ U+203A 'single right-pointing angle quotation mark' -->
455
456         <!-- Geometric Shapes -->
457         <entity name="loz" cdata="&#9674;"/> <!--lozenge, U+25CA ISOpub --
458 ↪ >
459
460         <!-- Miscellaneous Symbols -->
461         <entity name="spades" cdata="&#9824;"/> <!--black spade suit, _
462 ↪ U+2660 ISOpub -->
463         <!-- black here seems to mean filled as opposed to hollow -->
464         <entity name="clubs" cdata="&#9827;"/> <!--black club suit = _
465 ↪ shamrock, U+2663 ISOpub -->
466         <entity name="hearts" cdata="&#9829;"/> <!--black heart suit = _
467 ↪ valentine, U+2665 ISOpub -->
468         <entity name="diams" cdata="&#9830;"/> <!--black diamond suit, _
469 ↪ U+2666 ISOpub -->
470
471         <entity name="quot" cdata="&#34;"/> <!--quotation mark = APL_
472 ↪ quote, U+0022 ISOnum -->
473         <!-- Latin Extended-A -->
474         <entity name="OElig" cdata="&#338;"/> <!--latin capital ligature_
475 ↪ OE, U+0152 ISOlat2 -->
476         <entity name="oelig" cdata="&#339;"/> <!--latin small ligature_
477 ↪ oe, U+0153 ISOlat2 -->

```

(suite sur la page suivante)

(suite de la page précédente)

```

457      <!-- ligature is a misnomer, this is a separate character in some_
↳languages -->
458      <entity name="Scaron" cdata="&#352;" /> <!--latin capital letter_
↳S with caron, U+0160 ISolat2 -->
459      <entity name="scaron" cdata="&#353;" /> <!--latin small letter s_
↳with caron, U+0161 ISolat2 -->
460      <entity name="Yuml" cdata="&#376;" /> <!--latin capital letter Y_
↳with diaeresis, U+0178 ISolat2 -->
461
462      <!-- Spacing Modifier Letters -->
463      <entity name="circ" cdata="&#710;" /> <!--modifier letter_
↳circumflex accent, U+02C6 ISOpub -->
464      <entity name="tilde" cdata="&#732;" /> <!--small tilde, U+02DC_
↳ISODia -->
465
466      <!-- General Punctuation -->
467      <entity name="ensp" cdata="&#8194;" /> <!--en space, U+2002 ISOpub_
↳-->
468      <entity name="emsp" cdata="&#8195;" /> <!--em space, U+2003 ISOpub_
↳-->
469      <entity name="thinsp" cdata="&#8201;" /> <!--thin space, U+2009_
↳ISOpub -->
470      <entity name="zwnj" cdata="&#8204;" /> <!--zero width non-joiner,_
↳U+200C NEW RFC 2070 -->
471      <entity name="zwj" cdata="&#8205;" /> <!--zero width joiner,_
↳U+200D NEW RFC 2070 -->
472      <entity name="lrm" cdata="&#8206;" /> <!--left-to-right mark,_
↳U+200E NEW RFC 2070 -->
473      <entity name="rlm" cdata="&#8207;" /> <!--right-to-left mark,_
↳U+200F NEW RFC 2070 -->
474      <entity name="ndash" cdata="&#8211;" /> <!--en dash, U+2013 ISOpub_
↳-->
475      <entity name="mdash" cdata="&#8212;" /> <!--em dash, U+2014 ISOpub_
↳-->
476      <entity name="lsquo" cdata="&#8216;" /> <!--left single quotation_
↳mark, U+2018 ISOnum -->
477      <entity name="rsquo" cdata="&#8217;" /> <!--right single quotation_
↳mark, U+2019 ISOnum -->
478      <entity name="sbquo" cdata="&#8218;" /> <!--single low-9 quotation_
↳mark, U+201A NEW -->
479      <entity name="ldquo" cdata="&#8220;" /> <!--left double quotation_
↳mark, U+201C ISOnum -->
480      <entity name="rdquo" cdata="&#8221;" /> <!--right double quotation_
↳mark, U+201D ISOnum -->
481      <entity name="bdquo" cdata="&#8222;" /> <!--double low-9 quotation_
↳mark, U+201E NEW -->
482      <entity name="dagger" cdata="&#8224;" /> <!--dagger, U+2020 ISOpub_
↳-->
483      <entity name="Dagger" cdata="&#8225;" /> <!--double dagger, U+2021_
↳ISOpub -->
484      <entity name="permil" cdata="&#8240;" /> <!--per mille sign,_
↳U+2030 ISOTECH -->
485      <entity name="lsaquo" cdata="&#8249;" /> <!--single left-pointing_
↳angle quotation mark, U+2039 ISO proposed -->
486      <!-- lsaquo is proposed but not yet ISO standardized -->
487      <entity name="rsaquo" cdata="&#8250;" /> <!--single right-pointing_
↳angle quotation mark, U+203A ISO proposed -->

```

(suite sur la page suivante)

(suite de la page précédente)

```

488      <!-- rsaquo is proposed but not yet ISO standardized -->
489      <entity name="euro" cdata="&#8364;" /> <!--euro sign, U+20AC NEW -
490      ↪->
491      </html-entities>
492  </anti-samy-rules>

```

8.2.1.8 Fichier /vitam/conf/<composant>/vitam.conf

```

1  secret : {{plateforme_secret}}
2  filterActivation : {{ vitam_struct.secret_platform }}
3  {% if vitam_struct.vitam_component == vitam.processing.vitam_component %}
4  distributeurBatchSize: 800
5  workerBulkSize: 16
6  {% endif %}
7  {% if vitam_struct.vitam_component == vitam.metadata.vitam_component %}
8  storeGraphElementsPerFile: 10000
9  storeGraphOverlapDelay: 300
10 expireCacheEntriesDelay: 300
11 deleteIncompleteReconstructedUnitDelay: 2592000
12 migrationBulkSize: 10000
13 {% endif %}
14 distributionThreshold : 100000
15 eliminationAnalysisThreshold : 100000
16 eliminationActionThreshold : 10000
17 intervalDelayCheckIdle : 5000
18 maxDelayUnusedConnection : 5000
19 delayValidationAfterInactivity : 2500
20 tenants: [ "{{ vitam_tenant_ids | join(' ', ' ') }}" ]
21 adminTenant : {{vitam_tenant_admin}}
22 forceChunkModeInputStream : {{vitam_defaults.vitam_force_chunk_mode}}
23
24 {% if vitam_struct.vitam_component == vitam.worker.vitam_component %}
25 reclassificationMaxBulkThreshold: 1000
26 reclassificationMaxUnitsThreshold: 10000
27 reclassificationMaxGuildListSizeInLogbookOperation: 1000
28 {% endif %}
29
30 keywordMaxLength: 32766
31 textMaxLength: 32766
32
33 classificationLevel :
34   allowList : [{% for classification in classificationList %}{{ classification }}{% ↪
35   if not loop.last %},{% endif %}}{% endfor %}]
36   authorizeNotDefined: {{classificationLevelOptional}}
37
38 environmentName: {{ vitam_prefix_offer|default(vitam_site_name) }}

```

Ce fichier permet de définir le secret de plate-forme.

8.2.1.9 Fichier /vitam/conf/<composant>/vitam.metrics.conf

```

1 # Fichier de configuration des métriques
2 #
3 # Les différents clés disponibles pour ce fichier de configuration sont les_
  ↳ suivantes :
4 #
5 # metricsJersey: true / false           Active ou non les métriques Jersey
6 # metricsJVM: true / false             Active ou non les métriques JVM
7 #
8 # metricReporter: ELASTICSEARCH | LOGBACK | NONE           défini le_
  ↳ type de reporter
9 # metricReporterInterval: int > 0           défini l
  ↳ 'interval entre chaque reporting
10 # metricReporterIntervalUnit: TimeUnit (ex: SECONDS, MINUTES...) défini le_
  ↳ type d'interval
11 #
12 # Si le reporter est de type LOGBACK, la clé suivante est configurable:
13 # metricLogLevel: DEBUG | INFO | WARN | ERROR ...           défini le_
  ↳ niveau de log Logback
14 #
15 # Si le reporter est de type ELASTICSEARCH, la clé suivante est obligatoire :
16 #
17 # (un tableau avec les différentes adresses des bases ElasticSearch)
18 # metricReporterHosts:
19 #     - 127.0.0.1:9201
20 #     - 0.0.0.0:80
21 #     - 8.8.8.8:22
22
23 {% if (groups['hosts-logstash'] | length) > 0 %}
24 metricsJersey: true
25 metricsJVM: true
26
27 metricReporter: ELASTICSEARCH
28 metricReporterHosts:
29 {% for host in groups['hosts-elasticsearch-log'] %}
30     - "{{ hostvars[host]['ip_admin'] }}:{{ elasticsearch.log.port_http }}"
31 {% endfor %}
32 metricLogLevel: DEBUG
33 metricReporterInterval: 3
34 metricReporterIntervalUnit: MINUTES
35 {% endif %}

```

8.2.1.10 Fichier /vitam/conf/<composant>/java.security

```

1 # Use Bouncy Castle Provider when it is available
2 security.provider.9=org.bouncycastle.jce.provider.BouncyCastleProvider
3
4 # Override the default list of Centos 7 that disable Elliptic Curved Based Algorithms
5 jdk.tls.disabledAlgorithms="SSLv3, RC4, MD5withRSA, DH keySize < 768,RSA keySize <_
  ↳ 2048"

```

8.2.2 Access

8.2.2.1 access external

8.2.2.1.1 Présentation

Access-external est le composant d'interface entre *VITAM* et un *SIA* client, permettant de réaliser des recherches sur les objets archivés et les journaux. Il permet également quelques fonctions d'administration, en particulier les chargements des référentiels.

Rôle :

- Exposer les API publiques du système
- Sécuriser l'accès aux API de VITAM

8.2.2.1.2 Configuration / fichiers utiles

Les fichiers de configuration sont gérés par les procédures d'installation ou de mise à niveau de l'environnement *VITAM*. Se référer au *DIN*.

Les fichiers de configuration sont définis sous `/vitam/conf/access-external`.

8.2.2.1.2.1 Fichier `access-external.conf`

```
authentication: false
jettyConfig: jetty-config.xml
tenantFilter : true
```

8.2.2.1.2.2 Fichier `access-internal-client.conf`

```
serverHost: {{vitam.accessinternal.host}}
serverPort: {{vitam.accessinternal.port_service}}
```

8.2.2.1.2.3 Fichier `functional-administration-client.conf`

```
serverHost: {{vitam.functional_administration.host}}
serverPort: {{vitam.functional_administration.port_service}}
```

8.2.2.1.2.4 Fichier `ingest-internal-client.conf`

```
serverHost: {{vitam.ingestinternal.host}}
serverPort: {{vitam.ingestinternal.port_service}}
```

8.2.2.1.2.5 Fichier `internal-security-client.conf`

```
serverHost: {{ vitam.security_internal.host }}
serverPort: {{ vitam.security_internal.port_service }}
secure: false
```

8.2.2.1.3 Opérations

- Démarrage du service

En tant qu'utilisateur root : `systemctl start vitam-access-external`

- Arrêt du service

En tant qu'utilisateur root : `systemctl stop vitam-access-external`

- Sauvegarde du service

Ce service ne nécessite pas de sauvegarde particulière.

- Supervision du service

Contrôler le retour HTTP 200 sur l'URL <protocole web https ou https>://<host>:<port>/access-external/v1/status

Contrôler le retour HTTP 200 sur l'URL <protocole web https ou https>://<host>:<port admin>/admin/v1/status

- Exports

N/A

- gestion de la capacité

N/A

- actions récurrentes

- cas des batches

N/A

8.2.2.2 access-internal

8.2.2.2.1 Présentation du composant

Access-internal est le composant *VITAM*, permettant de réaliser des recherches et consultations sur les objets archivés et les journaux. Il permet également de modifier les informations d'un *ArchiveUnit*.

Rôle :

- Permettre l'accès aux données du système VITAM

Fonction :

- Exposition des fonctions de recherche d'archives offertes par metadata ;
- Exposition des fonctions de parcours de journaux offertes par logbook.

8.2.2.2.2 Configuration / fichiers utiles

Les fichiers de configuration sont gérés par les procédures d'installation ou de mise à niveau de l'environnement *VITAM*. Se référer au *DIN*.

Les fichiers de configuration sont définis sous `/vitam/conf/access`.

8.2.2.2.1 Fichier `access-internal.conf`

Ce fichier permet de définir l'URL d'accès au metadata server.

```
urlMetaData: {{vitam.metadata | client_url}}
urlWorkspace: {{vitam.workspace | client_url}}
urlProcessing: {{vitam.processing | client_url}}
jettyConfig: jetty-config.xml
```

8.2.2.2.2 Fichier `storage-client.conf`

Ce fichier permet de définir l'accès au storage-engine.

```
serverHost: {{ vitam.storageengine.host }}
serverPort: {{ vitam.storageengine.port_service }}
```

8.2.2.2.3 Fichier `metadata-client.conf`

```
serverHost: {{ vitam.metadata.host }}
serverPort: {{ vitam.metadata.port_service }}
```

8.2.2.2.4 Fichier `functional-administration-client.conf`

```
serverHost: {{ vitam.functional_administration.host }}
serverPort: {{ vitam.functional_administration.port_service }}
```

8.2.2.3 Opérations

- Démarrage du service

En tant qu'utilisateur root : `systemctl start vitam-access-internal`

- Arrêt du service

En tant qu'utilisateur root : `systemctl stop vitam-access-internal`

- Sauvegarde du service

Ce service ne nécessite pas de sauvegarde particulière.

- Supervision du service

Contrôler le retour HTTP 200 sur l'URL <protocole web https ou https>://<host>:<port>/access/v1/status

Contrôler le retour HTTP 200 sur l'URL <protocole web https ou https>://<host>:<port admin>/admin/v1/status

- Exports

N/A

- gestion de la capacité

N/A

- actions récurrentes

- cas des batches

N/A

8.2.3 Batch-Report

8.2.3.1 Présentation

Le composant batch-report permet de stocker des données de traitements de masse (en particulier, élimination) pour les agréger sous forme de rapports.

Ce module utilise une base de données MongoDB « report », dans laquelle sont stockées, sous différentes collections (entre autres, EliminationActionObjectGroup et EliminationActionUnit), les données.

Ce module est appelé par le composant « worker » pour collecter les données durant les *workflows* d'élimination, entre autres.

Ce module est ensuite appelé par le composant « worker » pour restituer les données agrégées sous forme de rapports.

8.2.3.2 Configuration

Ce document spécifie la configuration (fichiers de config) pour lancer le services de batch-report.

Tous ces fichiers de configuration sont placés dans le répertoire `/vitam/conf/batch-report`.

8.2.3.2.1 Fichier `batch-report.conf`

```
# Configuration MongoDB
mongoDbNodes:
{% for server in groups['hosts-mongos-data'] %}
- dbHost: {{hostvars[server]['ip_service']}}
  dbPort: {{ mongodb.mongos_port }}
{% endfor %}
dbName: report
dbAuthentication: {{ mongodb.mongo_authentication }}
dbUserName: {{ mongodb['mongo-data'].report.user }}
dbPassword: {{ mongodb['mongo-data'].report.password }}
jettyConfig: jetty-config.xml

workspaceUrl: {{ vitam.workspace | client_url }}
```

8.2.3.3 Client batch-report

Pour la création d'un client batch-report, nous avons besoin aussi du fichier de configuration `batch-report-client.conf` qui précise le serveur host et la porte du serveur où le client se connecte pour les requêtes.

8.2.3.4 Opérations

- Démarrage du service

En tant qu'utilisateur `root` : `systemctl start vitam-batch-report`

- Arrêt du service

En tant qu'utilisateur root : `systemctl stop vitam-batch-report`

- Sauvegarde du service

Ce service ne nécessite pas de sauvegarde particulière.

- Supervision du service

Contrôler le retour HTTP 200 sur l'URL `<protocole web https ou https>://<host>:<port admin>/admin/v1/status`

- Exports

N/A

- gestion de la capacité

N/A

- actions récurrentes

- cas des batches

N/A

8.2.4 common-plugin

8.2.4.1 Présentation du composant

common-plugin est le composant permettant de réaliser des plugins sans appel à des package privé . Rôle :

- l'objet de ce common-plugin n'est pas que de fournir des interfaces à implémenter mais aussi les classes d'implémentations imposées par Vitam pour réaliser des plugins.

Fonction :

- Exposition interfaces à implémenter et les classes d'implémentations pour réaliser des plugins .

8.2.4.2 Classes utiles

L'Objectif de Plugin Common est d'inclure tous les classes utiles afin de créer un plugin à partir de ce package .

Les classes de model sont définis sous `/vitam/common/model`.

8.2.4.2.1 Classe Item Status

Ce classe permet de retourner le statut d'un Item.

8.2.4.2.2 Classe VitamAutoCloseable

Le mot clé try-with-resources garantit que chaque ressource sera fermée lorsqu'elle n'est plus utilisée. Une ressource et un objet qui implémente l'interface VitamAutoCloseable. Il est donc possible d'utiliser une instance de ces interfaces avec le mot clé try-with-resources.

Les classes de common parameter sont définis sous `/vitam/common/parameter`.

8.2.4.2.3 Classe ParameterHelper

Ce classe permet de faire un check sur les paramètres et avoir le tenant parameter de session vitam .

8.2.4.2.4 Classe `VitamParameter`

Cet interface permet d'aider à créer des nouveaux paramètres liés au classes .

Les classes de common exception sont définis sous `/vitam/processing/common/exception`.

8.2.4.2.5 Classe `ProcessingException`

Ce classe est le classe père de tous les Vitam Processing Exception .

Les classes de model common processing sont définis sous `/vitam/processing/common/model`.

8.2.4.2.6 Classe `IOParameter`

Ce class permet de définir les paramètres Input et Output pour une action et une step .

8.2.4.2.7 Classe `ProcessingUri`

Ce classe permet de formater le processing URI .

8.2.4.2.8 Classe `UriPrefix`

C'est le Handler IO

Les classes des paramètres common sont définis sous `/vitam/processing/common/parameter`.

8.2.4.2.9 Classe `AbstractWorkerParameters`

C'est une implémentation abstraite de tous les paramètres de workers .

8.2.4.2.10 Classe `DefaultWorkerParameters`

Ce classe permet de définir les paramètres par défaut d'un worker.

8.2.4.2.11 Classe `WorkerParameterName`

Ce classe inclut une énumération avec tous les noms des paramètres d'un worker .

8.2.4.2.12 Classe `WorkerParameters`

Ce classe permet de définir les paramètres de worker.

8.2.4.2.13 Classe `WorkerParametersDeserializer`

Ce classe permet de définir les paramètres d'un worker deserializer.

8.2.4.2.14 Classe WorkerParametersFactory

Ce classe permet de définir les paramètres d'un worker Factory.

8.2.4.2.15 Classe WorkerParametersSerializer

Ce classe permet de définir les paramètres de Worker Serializer.

Les classes de model sont définis sous /vitam/worker/common.

8.2.4.2.16 Interface HandlerIO

Cet interface permet de définir les paramètres in et out de tous les Handlers.

Les classes de l'api sont définis sous /vitam/worker/core/api.

8.2.4.2.17 Classe WorkerAction

C'est l'interface contrat de tous les actions Handler event. Un action Handler doit implémenter cette interface .

Les classes de l'implémentation sont définis sous /vitam/worker/core/impl.

8.2.4.2.18 Classe HandlerIOImpl

Ce classe définit les paramètres in et out d'un Handler

How to use : Pour créer un Plugin :

- extends Abstract Class Action Handler
- implementer l'interface VitamAutoCloseable pour garantir qu'une ressource sera fermée lorsqu'elle n'est plus utilisée.
- Un constructeur par défaut
- **redéfinir la méthode execute de l'Action Handler :**
 - Paramètre WorkerParameters et Handler IO
 - type de retour Item Status
 - throws Processing Exception
- **faire l'override de méthode CheckMandatoryIOParameter**
 - Paramètre Handler IO
 - throws Processing Exception

8.2.5 Common

8.2.5.1 Présentation

8.2.5.2 Format Identifiers

Les services d'identification de formats peuvent être déployés sur tous les serveurs applicatifs vitam.

8.2.5.2.1 Configuration des services d'identification des formats

Dans `/vitam/conf` du serveur applicatif où sont déployés les services d'identification de formats, il faut un fichier **format-identifiers.conf**. C'est un fichier YAML de configuration des services d'identification de format. Il possède les configurations des services que l'on souhaite déployer sur le serveur.

Le code suivant contient un exemple de toutes les configurations possibles :

```
siegfried-local:
type: SIEGFRIED
client: http
host: localhost
port: 55800
rootPath: /root/path
versionPath: /root/path/version/folder
createVersionPath: false
mock:
type: MOCK
```

- **Le service Mock :**
 - identifié par *mock*
 - *type* : le type de service déployé : *MOCK*
- **Le service Siegfried :**
 - identifié par *siegfried-local*
 - *type* : le type de service déployé : *SIEGFRIED*
 - *client* : type de client (pour le moment seul *http* existe).
 - *host* : le host du serveur siegfried déployé (devrait être le host du serveur courant)
 - *port* : le port du serveur siegfried déployé
 - *rootPath* : la racine sur laquelle le service Siegfried doit résoudre les fichiers à tester (ex : « /data »)
 - *versionPath* : le chemin vers un dossier vide pour renvoyer la version (Doit posséder des droits en lecture)
 - *createVersionPath* : Si *false* le dossier doit pré-exister sur le serveur sur lequel tourne Siegfried. Sinon, le client siegfried tente de créer automatiquement le dossier en local.

NOTE : Chaque serveur est en charge de décrire la configuration nécessaire

8.2.6 Functional administration

8.2.6.1 Présentation

8.2.6.2 Configuration / fichiers utiles

Les fichiers de configuration sont gérés par les procédures d'installation ou de mise à niveau de l'environnement *VITAM*. Se référer au *DIN*.

Les fichiers de configuration sont définis sous `/vitam/conf/functional-administration`.

8.2.6.2.1 Fichier functional-administration.conf

ce fichier permet de définir l'URL d'accès au access server.

```
# Configuration MongoDB
mongoDbNodes:
{% for host in groups['hosts-mongos-data'] %}
- dbHost: {{hostvars[host]['ip_service']}}
  dbPort: {{ mongodb.mongos_port }}
{% endfor %}
dbName: masterdata
dbAuthentication: {{ mongodb.mongo_authentication }}
dbUserName: {{ mongodb['mongo-data'].functionalAdmin.user }}
dbPassword: {{ mongodb['mongo-data'].functionalAdmin.password }}

#Basic Authentication
adminBasicAuth:
- userName: {{ admin_basic_auth_user }}
  password: {{ admin_basic_auth_password }}

jettyConfig: jetty-config.xml
workspaceUrl: {{vitam.workspace | client_url}}
processingUrl: {{vitam.processing | client_url}}

# Elasticsearch
clusterName: {{ vitam_struct.cluster_name }}
elasticsearchNodes:
{% for host in groups['hosts-elasticsearch-data'] %}
- hostName: {{hostvars[host]['ip_service']}}
  tcpPort: {{ elasticsearch.data.port_tcp }}
{% endfor %}

# ExternalId configuration
listEnableExternalIdentifiers:
  0:
    - INGEST_CONTRACT
    - ACCESS_CONTRACT
    - ARCHIVE_UNIT_PROFILE
  1:
    - INGEST_CONTRACT
    - ACCESS_CONTRACT
    - PROFILE
    - SECURITY_PROFILE
    - CONTEXT

listMinimumRuleDuration:
  2:
    AppraisalRule : 1 year
```

8.2.6.2.2 Passage des identifiants des référentiels en mode esclave

La génération des identifiants des référentiels est gérée par Vitam quand 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

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

Note : Cette modification de comportement est réalisable post-installation. Une interruption temporaire de service est à prévoir (redémarrage du service `vitam-functional-administration`).

Par défaut, tous les services référentiels de **VITAM** fonctionnent en mode maître. Pour désactiver le mode maître de Vitam, il faut modifier le fichier de configuration `/vitam/conf/functional-administration/functional-administration.conf`.

```
# ExternalId configuration

listEnableExternalIdentifiers:
0:
  - INGEST_CONTRACT
  - ACCESS_CONTRACT
1:
  - INGEST_CONTRACT
  - ACCESS_CONTRACT
  - PROFILE
  - SECURITY_PROFILE
  - CONTEXT
```

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 :

- `INGEST_CONTRACT` : contrats d'entrée
- `ACCESS_CONTRACT` : contrats d'accès
- `PROFILE` : profils SEDA
- `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

Note : se référer au métier pour ces choix.

Avertissement : Cette modification implique le redémarrage du/des composants (si mono-instance ou multi-instances du composant).

Prudence : En mode « esclave », il est fortement recommandé de faire débiter les référentiels avec d'autres chaînes de caractères que celle définies en mode « maître ».

Prudence : Ne pas oublier de répercuter cette modification sur le site secondaire

8.2.6.2.3 Configuration du Functional administration

functional-administration.conf : Fichier Yaml de configuration du server *worker*. Il possède une propriété :

- **listMinimumRuleDuration** : la durée minimum de chaque type de règle par tenant

```
listMinimumRuleDuration:
  2:
    AppraisalRule : 1 year
```

8.2.6.3 Opérations

- Démarrage du service

En tant qu'utilisateur root : `systemctl start vitam-functional-administration`

- Arrêt du service

En tant qu'utilisateur root : `systemctl stop vitam-functional-administration`

- Sauvegarde du service

Ce service ne nécessite pas de sauvegarde particulière.

- Supervision du service

Contrôler le retour HTTP 200 sur l'URL <protocole web https ou https>://<host>:<port>/functional-administration/v1/status

Contrôler le retour HTTP 200 sur l'URL <protocole web https ou https>://<host>:<port admin>/admin/v1/status

- Exports

N/A

- gestion de la capacité

N/A

- actions récurrentes

- cas des batches

N/A

8.2.7 Hello World Plugin

8.2.7.1 Présentation

Le composant hello-world-plugin est un exemple qui montre comment développer un plugin custom. Il suffit donc de prendre ce projet maven comme exemple et d'adapter le plugin à souhait.

Remarque : Sur le pom de ce module, seule la dépendance vers common-plugin est nécessaire.

```
<dependency>
  <groupId>fr.gouv.vitam</groupId>
  <artifactId>common-plugin</artifactId>
  <version>${vitam.version}</version>
</dependency>
```

Vous pouvez bien sûr ajouter d'autres dépendances qui servent à votre plugin custom.

8.2.7.1.1 Comment intégrer votre plugins dans vitam ?

Après avoir développé votre plugins en suivant les consignes ci-dessus, il faut faire ce qui suit :

- Générer votre jar
- Copier votre jar manuellement dans le dossier /vitam/conf/worker/plugins-workspace/ Ou bien copier le dans le dossier de déploiement ansible ~/vitam/deployment/ansible-vitam/roles/vitam/files/worker/plugins-workspace/
- Modifier le fichier plugins.json qui se trouve soit dans le dossier /vitam/conf/worker/plugins.json en production ou bien dans le dossier de déploiement ansible qui se trouve :~/vitam/deployment/ansible-vitam/roles/vitam/files/worker/plugins.json Ajouter une entrée comme suit :

```
"HELLO_WORLD_PLUGIN": {
  "className": "fr.vitam.plugin.custom.HelloWorldPlugin",
  "propertiesFile": "hello_world_plugin.properties",
  "jarName": "hello-world-plugin-1.14.0-SNAPSHOT.jar"
}
```

Avertissement : jarName doit contenir uniquement le nom du jar avec extension “.jar”

A présent sur n'importe quel workflow, vous pouvez ajouter une action ayant comme « actionKey » la clé de votre plugin. Dans cet exemple : actionKey=HELLO_WORLD_PLUGIN

8.2.7.1.2 Créer un nouveau workflow

Tout d'abord création d'un nouveau workflow :

- Créer un nouveau workflow peut se résumer juste à copier un workflow existant et modifier son *identifier* et son *workerGroupId* pour, par exemple, utiliser des machines plus puissantes pour ce workflow
- L'identifier d'un workflow doit être UNIQUE, sinon un workflow existant portant le même identifiant sera remplacé par le nouveau.
- La valeur de typeProc n'est pas actuellement dynamique (veuillez vous référer à l'enum LogbookType-Process pour voir les différentes valeurs possibles)
- La valeur actionKey doit être soit le handler name d'un plugin ou handler existant, soit celui d'un nouveau plugin.

Exemple d'un workflow :

```
{
  "id": "SampleIngestWorkflow",
  "name": "Sample Ingest Workflow",
  "identifier": "SAMPLE_PROCESS_SIP_UNITARY",
```

(suite sur la page suivante)

(suite de la page précédente)

```

"typeProc": "INGEST",
"comment": "Sample Ingest Workflow V6",
"steps": [
  {
    "workerGroupId": "DefaultWorker",
    "stepName": "STP_INGEST_CONTROL_SIP",
    "behavior": "BLOCKING",
    "distribution": {
      "kind": "REF"
    },
    "actions": [
      {
        "action": {
          "actionKey": "HELLO_WORLD_PLUGIN",
          "behavior": "BLOCKING",
          "in": [
            {
              "name": "var_name",
              "uri": "VALUE:Hello World"
            }
          ]
        }
      }
    ]
  }
]

```

Avvertissement : Le fichier *workflow* doit être un fichier json avec comme extension (.json) sinon le fichier ne sera pas pris en compte.

8.2.7.1.3 Comment ajouter un nouveau workflow dans vitam ?

Il faut d'abord créer un fichier json avec un nom de votre choix et ayant la forme de l'exemple ci-dessus. Veuillez vous référer aux différents workflow existants pour avoir plus d'information.

Il faut ensuite copier ce fichier (CustomWorkflow.json) dans :

- En production : Manuellement dans le dossier /vitam/conf/processing/workflows/
- Via ansible : Dans le dossier ~/vitam/deployment/ansible-vitam/roles/vitam/files/processing/workflows/

8.2.7.1.4 Comment ajouter la traduction de clés des Plugins ?

On peut dans n'importe quel service vitam ajouter dans le dossier conf les deux fichiers suivants : - vitam-logbook-messages_fr.properties - vitam-error-messages.properties Ces deux fichiers garantissent la traduction des clés.

Pour les nouveaux plugins ajoutés, le fichier properties qui est à l'intérieur du jar n'est visible que par le service (worker). Pour qu'on puisse avoir ces clés traduites, le fichier vitam-logbook-messages_fr.properties doit contenir les traductions des clés de ce nouveau plugin. Il faut copier ce fichier dans le dossier de conf de processing s'il n'existe pas.


```
HELLO_WORLD_PLUGIN=Test d'un plugin Hello World
HELLO_WORLD_PLUGIN.OK=Test d'un plugin Hello World réalisé avec succès
HELLO_WORLD_PLUGIN.KO=Échec lors du test d'un plugin Hello World
HELLO_WORLD_PLUGIN.FATAL=Erreur fatale lors du test d'un plugin Hello World
HELLO_WORLD_PLUGIN.WARNING=Avertissement lors du test d'un plugin Hello_
↪World
```

8.2.7.1.5 Comment appeler le nouveau workflow ?

En utilisant l'API d'ingest et en passant les paramètres suivant :

- X_CONTEXT_ID : l'identifier de votre workflow (dans l'exemple ci-dessus SAMPLE_PROCESS_SIP_UNITARY)
- X_ACTION : votre action (RESUME, NEXT)

Le reste se fait automatiquement par le *back office*.

8.2.7.1.6 Remarques

- L'ajout d'un workflow dans processing en production ne nécessite pas de redémarrage. Un thread passe chaque heure configure pour recharger les derniers workflow (ajoutés ou modifiés)
- L'ajout d'un jar dans les workers et les fichiers properties nécessitent, cependant, le redémarrage des workers et des services concernés.

8.2.7.1.7 Sécurité

Les plugins externes sont exécutés au même niveau de sécurité que ceux interne à [VITAM](#). L'isolation de l'exécution des plugins externes n'est pas assurée par [VITAM](#). C'est donc à l'exploitant de garantir la sécurité des plugins intégrés.

8.2.8 ihm-demo

8.2.8.1 Présentation

Cette IHM a été développée pour des fins de tests de VITAM.

Rôle :

- Permettre une utilisation basique de VITAM, notamment sans SIA

Fonctions :

- Représentation des arborescences et des graphes
- Formulaire dynamiques
- Suivi des opérations
- Gestion des référentiels

8.2.8.2 Configuration / fichiers utiles

Les fichiers de configuration sont gérés par les procédures d'installation ou de mise à niveau de l'environnement [VITAM](#). Se référer au [DIN](#).

Les fichiers de configuration sont définis sous `/vitam/conf/ihm-demo`.

8.2.8.2.1 Fichier access-external-client.conf

ce fichier permet de définir l'URL d'accès au access server.

```
serverHost: {{ vitam.accessexternal.host }}
serverPort: {{ vitam.accessexternal.port_service }}
secure: true
sslConfiguration :
  keystore :
    - keyPath: {{vitam_folder_conf}}/keystore_{{ vitam_struct.vitam_component }}.p12
      keyPassword: {{keystores.client_external.ihm_demo}}
  truststore :
    - keyPath: {{vitam_folder_conf}}/truststore_{{ vitam_struct.vitam_component }}.jks
      keyPassword: {{truststores.client_external}}
hostnameVerification: true
```

8.2.8.2.2 Fichier ihm-demo.conf

```
serverHost: {{ ip_service }}
port: {{ vitam_struct.port_service }}

baseUrl: "{{ vitam_struct.baseurl }}"
staticContent: {{ vitam_struct.static_content }}
baseUri: /{{vitam_struct.baseuri}}
secureMode:
{% for realm in vitam_struct.authentication_realms %}
- {{ realm }}
{% endfor %}

jettyConfig: jetty-config.xml
authentication: true
enableXsrFilter: true
enableSession: true

allowedMediaTypes:
{% for mediaType in vitam_struct.allowedMediaTypes %}
- type: {{ mediaType.type }}
  subtype: {{ mediaType.subtype }}
{% endfor %}
```

tenants : liste des tenants disponibles sur l'ihm-demo.

8.2.8.2.3 Fichier ingest-external-client.conf

```
serverHost: {{ vitam.ingestexternal.host }}
serverPort: {{ vitam.ingestexternal.port_service }}
secure: true
sslConfiguration :
  keystore :
    - keyPath: {{vitam_folder_conf}}/keystore_{{ vitam_struct.vitam_component }}.p12
      keyPassword: {{keystores.client_external.ihm_demo}}
  truststore :
    - keyPath: {{vitam_folder_conf}}/truststore_{{ vitam_struct.vitam_component }}.jks
```

(suite sur la page suivante)

(suite de la page précédente)

```
keyPassword: {{truststores.client_external}}
hostnameVerification: true
```

8.2.8.2.4 Fichier shiro.ini

```
# =====
# Shiro INI configuration
# =====

[main]
# Objects and their properties are defined here,
# Such as the securityManager, Realms and anything
# else needed to build the SecurityManager

# Cache Manager
builtInCacheManager = org.apache.shiro.cache.MemoryConstrainedCacheManager

# Security Manager
securityManager.cacheManager = $builtInCacheManager

sessionManager = org.apache.shiro.web.session.mgt.DefaultWebSessionManager
securityManager.sessionManager = $sessionManager
securityManager.sessionMode = native
securityManager.sessionManager.globalSessionTimeout = {{ vitam_struct.session_timeout_
↳ }}
securityManager.sessionManager.sessionIdUrlRewritingEnabled = false
securityManager.sessionManager.sessionIdCookie.secure = {{ vitam_struct.secure_cookie_
↳ }}
securityManager.rememberMeManager.cookie.secure = {{ vitam_struct.secure_cookie }}
securityManager.rememberMeManager.cookie.httpOnly = true

# Notice how we didn't define the class for the FormAuthenticationFilter ('authc') -
↳ it is instantiated and available already:
authc.loginUrl = /#!/login

# credentialsMatcher
sha256Matcher = org.apache.shiro.authc.credential.Sha256CredentialsMatcher

{% if "iniRealm" in vitam_struct.authentication_realms %}
iniRealm.credentialsMatcher = $sha256Matcher
{% endif %}

{% if "ldapRealm" in vitam_struct.authentication_realms %}
contextFactory = org.apache.shiro.realm.ldap.JndiLdapContextFactory
contextFactory.url = {{ ldap_authentication.ldap_protocol }}://{{ ldap_
↳ authentication.ldap_server }}:{{ ldap_authentication.ldap_port }}
{% if ldap_authentication.ldap_login is defined and ldap_authentication.ldap_pwd_
↳ is defined %}
{% if ldap_authentication.ldap_login != "" and ldap_authentication.ldap_pwd != ""
↳ %}
contextFactory.systemUsername = {{ ldap_authentication.ldap_login }}
contextFactory.systemPassword = {{ ldap_authentication.ldap_pwd }}
```

(suite sur la page suivante)

(suite de la page précédente)

```

{% endif %}
{% endif %}

ldapRealm = fr.gouv.vitam.common.auth.core.realm.LdapRealm
ldapRealm.ldapContextFactory = $contextFactory
ldapRealm.searchBase = "{{ ldap_authentication.ldap_base }}"
ldapRealm.groupRequestFilter = {{ ldap_authentication.ldap_group_request }}
ldapRealm.userDnTemplate = {{ ldap_authentication.ldap_userDn_Template }}
ldapRealm.groupRolesMap = "{{ ldap_authentication.ldap_admin_group }}":"admin", "{{ ldap_authentication.ldap_user_group }}":"user", "{{ ldap_authentication.ldap_guest_group }}":"guest"
{% endif %}

x509 = fr.gouv.vitam.common.auth.web.filter.X509AuthenticationFilter

x509.useHeader = False

x509credentialsMatcher = fr.gouv.vitam.common.auth.core.authc.
↳X509CredentialsSha256Matcher

{% if "x509Realm" in vitam_struct.authentication_realms %}
x509Realm = fr.gouv.vitam.common.auth.core.realm.X509KeystoreFileWithRoleRealm
x509Realm.grantedKeyStoreName = {{ vitam_folder_conf }}/grantedstore_ihm-demo.jks
x509Realm.grantedKeyStorePassphrase = {{ password_grantedstore }}
x509Realm.trustedKeyStoreName = {{ vitam_folder_conf }}/truststore_ihm-demo.jks
x509Realm.trustedKeyStorePassphrase = {{ password_truststore }}
x509Realm.credentialsMatcher = $x509credentialsMatcher
x509Realm.certificateDnRoleMapping = "CN=userAdmin,O=Vitam,L=Paris":"admin",
↳"CN=userUser,O=Vitam,L=Paris,C=FR":"user"
{% endif %}

securityManager.realms = {% for realm in vitam_struct.authentication_realms %}{% if
↳not loop.first %},{% endif %}${{ realm }}{% endfor %}

{% if "iniRealm" in vitam_struct.authentication_realms %}

[users]
# The 'users' section is for simple deployments
# # when you only need a small number of statically-defined
# # set of User accounts.
# #username = password
{% for item in vitam_users %}
  {{ item.login }}={{ item.password|hash('sha256') }}, {{ item.role }}
{% endfor %}

{% endif %}

[roles]
admin = *
user = messages:*, archiveseach:*, logbook:*, ingest:*, archiveupdate:*,
↳archiveunit:*, ingests:read, admin:formats:read, admin:rules:read, admin:accession-
↳register:read, logbookunitlifecycles:*, logbookobjects Lifecycles:*, clear:delete,
↳check:read, traceability:content:read, accesscontracts:read, profiles:read,
↳contracts:read, contexts:read, archiveunitprofiles:read, ontologies:read,
↳accessionregisterssymbolic:read
guest = archiveseach:*, archiveunit:*, units:*, unit:*, admin:accession-
↳register:read, accesscontracts:read

```

(suite sur la page suivante)

```
[urls]
# make sure the end-user is authenticated. If not, redirect to the 'authc.loginUrl'
↳above,
# and after successful authentication, redirect them back to the original account
↳page they
# were trying to view:
/v1/api/login = anon
/v1/api/logout = logout
/v1/api/messages/logbook = anon
/v1/api/tenants = anon
/v1/api/securemode = anon
/v1/api/admintenant = anon
/v1/api/permissions = x509
/v1/api/** = authc, x509
/#/** = authc
```

8.2.8.3 Configuration de apache shiro

8.2.8.4 Présentation authentification via LDAP et via certificat

Afin de pouvoir authentifier des clients via une base de données LDAP il suffit de bien configurer shiro. Pour ce faire, Vitam utilise le fichier shiro.ini qui a la forme suivante.

```
[main]
contextFactory = org.apache.shiro.realm.ldap.JndiLdapContextFactory
contextFactory.url = ldap://localhost:389
contextFactory.systemUsername = cn=admin,dc=example,dc=org
contextFactory.systemPassword = password

realm = fr.gouv.vitam.common.security.rest.LdapRealm
realm.ldapContextFactory = $contextFactory
realm.searchBase = "dc=example,dc=org"
realm.groupRequestFilter = (&(objectClass=groupOfNames)(member={0}))
realm.userDnTemplate = uid={0},dc=example,dc=org
realm.groupRolesMap = "cn=gadmins,dc=example,dc=org":"admin", "cn=gusers,dc=example,
↳dc=org":"user", "cn=gadmins,dc=example,dc=org":"guest"

securityManager.realms = $realm
```

```
x509 = fr.gouv.vitam.common.auth.web.filter.X509AuthenticationFilter
x509.useHeader = false
x509credentialsMatcher = fr.gouv.vitam.common.auth.core.authc.
↳X509CredentialsSha256Matcher
x509Realm = fr.gouv.vitam.common.auth.core.realm.X509KeystoreFileWithRoleRealm
x509Realm.grantedKeyStoreName = /vitam/conf/ihm-demo/grantedstore_ihm-demo.jks
x509Realm.grantedKeyStorePassphrase = azerty12
x509Realm.trustedKeyStoreName = /vitam/conf/ihm-demo/truststore_ihm-demo.jks
x509Realm.trustedKeyStorePassphrase = azerty10
x509Realm.credentialsMatcher = $x509credentialsMatcher
x509Realm.certificateDnRoleMapping = "CN=userAdmin,O=Vitam,L=Paris":"admin",
↳"CN=userUser,O=Vitam,L=Paris,C=FR":"user"
securityManager.realms = $x509Realm
```

8.2.8.5 Décryptage de shiro.ini

[main] Contient la déclaration des options et mappings dans l'authentification ldap :

- contextFactory.url : url du serveur ldap ;
- contextFactory.systemUsername : identifiant de l'utilisateur ;
- contextFactory.systemPassword : mot de passe ;
- realm.searchBase : le domaine de recherche dans LDAP ;
- realm.groupRequestFilter : chaque utilisateur est déclaré dans un groupe, cette requête sert à chercher les groupes de l'utilisateur ;
- realm.userDnTemplate : le modèle pour traduire un identifiant de l'utilisateur en DN (distinguished name) dans ldap ;
- realm.groupRolesMap : le mapping entre le DN des group de l'utilisateur et les rôles dans ihm ;
- x509Realm.grantedKeyStoreName : le fichier grantedstore ;
- x509Realm.trustedKeyStoreName : le fichier trustedstore ;
- x509Realm.certificateDnRoleMapping : le mapping entre le DisplayName de certificat et les rôles dans ihm.

Note : on peut déclarer plusieurs groupes qui ont la même rôle admin avec cette syntaxe :

```
"groupeA" : "admin", "groupeB" : "admin", "groupeC" : "admin"
```

8.2.8.6 Opérations

- Démarrage du service

En tant qu'utilisateur root : `systemctl start vitam-ihm-demo`

- Arrêt du service

En tant qu'utilisateur root : `systemctl stop vitam-ihm-demo`

- Sauvegarde du service

Ce service ne nécessite pas de sauvegarde particulière.

- Supervision du service

Contrôler le retour HTTP 200 sur l'URL <protocole web https ou https>://<host>:<port>/ihm-demo/v1/status

- Gestion des utilisateurs

Les utilisateurs sont actuellement gérés via le fichier `shiro.ini`, dans la section `[users]`.

- Créer un utilisateur

Lancer la commande shell suivante pour générer le mot de passe :

```
echo -n <motdepasse> | sha256sum
```

Copier le résultat.

Ensuite, éditer le fichier `/vitam/conf/ihm-demo/shiro.ini` et ajouter, dans la section `[users]`, la ligne suivante :

```
<login de l'utilisateur>=<résultat de la commande de génération de mot de_
↳passe précédente>
```

Pour terminer, relancer le service `vitam-ihm-demo` par la commande :

```
systemctl restart vitam-ihm-demo
```

- Supprimer un utilisateur

Dans la section `[users]`, enlever la ligne correspondant à l'utilisateur à supprimer. Pour terminer, relancer le service `vitam-ihm-demo` par la commande :

```
systemctl restart vitam-ihm-demo
```

- Exports

N/A

- gestion de la capacité

N/A

- actions récurrentes
- cas des batches

N/A

8.2.9 ihm-recette

8.2.9.1 Présentation

Cette IHM a été développée pour des fins de validation de VITAM. Elle permet de réaliser des tests de non-régression, mais également des actions sur le contenu des bases de données.

Danger : Cette IHM ne doit PAS être déployée dans un environnement de production !

8.2.9.2 Configuration / fichiers utiles

Les fichiers de configuration sont gérés par les procédures d'installation ou de mise à niveau de l'environnement *VITAM*. Se référer au *DIN*.

Les fichiers de configuration sont définis sous `/vitam/conf/ihm-recette`.

8.2.9.2.1 Fichier `access-external-client.conf`

ce fichier permet de définir l'URL d'accès au access server.

```
serverHost: {{ vitam.accessexternal.host }}
serverPort: {{ vitam.accessexternal.port_service }}
secure: true
sslConfiguration :
  keystore :
    - keyPath: {{vitam_folder_conf}}/keystore_{{ vitam_struct.vitam_component }}.p12
      keyPassword: {{keystores.client_external.ihm_recette}}
  truststore :
    - keyPath: {{vitam_folder_conf}}/truststore_{{ vitam_struct.vitam_component }}.jks
      keyPassword: {{truststores.client_external}}
hostnameVerification: false
```

8.2.9.2.2 Fichier driver-location.conf

```
driverLocation: {{vitam_folder_lib}}
```

8.2.9.2.3 Fichier driver-mapping.conf

```
driverMappingPath: {{vitam_folder_data}}/
delimiter: ;
```

8.2.9.2.4 Fichier functional-administration-client.conf

```
serverHost: {{ vitam.functional_administration.host }}
serverPort: {{ vitam.functional_administration.port_service }}
```

8.2.9.2.5 Fichier ihm-recette-client.conf

```
serverHost: {{ vitam_struct.host }}
serverPort: {{ vitam_struct.port_service }}
```

8.2.9.2.6 Fichier ihm-recette.conf

```
serverHost: {{ ip_service }}
port: {{ vitam_struct.port_service }}

baseUrl: "/{{ vitam_struct.baseuri }}"
baseUri: "/{{ vitam_struct.baseuri }}"
staticContent: "{{ vitam_struct.static_content }}"

jettyConfig: jetty-config.xml
authentication: true
enableXsrFilter: true
enableSession: true

secureMode:
{% for securemode in vitam_struct.secure_mode %}
- {{securemode}}
{% endfor %}
sipDirectory: {{ vitam_folder_data }}/test-data
performanceReportDirectory: {{ vitam_folder_data }}/report/performance

testSystemSipDirectory: {{ vitam_folder_data }}/test-data/system
testSystemReportDirectory: {{ vitam_folder_data }}/report/system
ingestMaxThread: {{ ansible_processor_cores * ansible_processor_threads_per_core + 1 }}
↪}

#
workspaceUrl: {{vitam.workspace | client_url}}
```

(suite sur la page suivante)

(suite de la page précédente)

```
# Configuration MongoDB
mongoDbNodes:
{% for server in groups['hosts-mongos-data'] %}
- dbHost: {{hostvars[server]['ip_service']}}
  dbPort: {{ mongodb.mongos_port }}
{% endfor %}
# Actually need this field for compatibility
dbName: admin
# @integ: parametrize it !
masterdataDbName: masterdata
logbookDbName: logbook
metadataDbName: metadata
dbAuthentication: {{ mongodb.mongo_authentication }}
dbUserName: {{ mongodb['mongo-data']['admin']['user'] }}
dbPassword: {{ mongodb['mongo-data']['admin']['password'] }}

# ElasticSearch
clusterName: {{ vitam_struct.cluster_name }}
elasticsearchNodes:
{% for server in groups['hosts-elasticsearch-data'] %}
- hostName: {{hostvars[server]['ip_service']}}
  tcpPort: {{ elasticsearch.data.port_tcp }}
{% endfor %}
```

8.2.9.2.7 Fichier ingest-external-client.conf

```
serverHost: {{ vitam.ingestexternal.host }}
serverPort: {{ vitam.ingestexternal.port_service }}
secure: true
sslConfiguration :
  keystore :
    - keyPath: {{vitam_folder_conf}}/keystore_{{ vitam_struct.vitam_component }}.p12
      keyPassword: {{ keystores.client_external.ihm_recette }}
  truststore :
    - keyPath: {{vitam_folder_conf}}/truststore_{{ vitam_struct.vitam_component }}.jks
      keyPassword: {{ truststores.client_external }}
hostnameVerification: false
```

8.2.9.2.8 Fichier shiro.ini

```
[main]

{% if vitam_struct.secure_mode == 'x509' %}
x509 = fr.gouv.vitam.common.auth.web.filter.X509AuthenticationFilter

x509.useHeader = {{vitam_ssl_user_header}}

x509credentialsMatcher = fr.gouv.vitam.common.auth.core.authc.
↳X509CredentialsSha256Matcher

x509Realm = fr.gouv.vitam.common.auth.core.realm.X509KeystoreFileRealm
x509Realm.grantedKeyStoreName = {{vitam_folder_conf}}/grantedstore_ihm-recette.jks
```

(suite sur la page suivante)

(suite de la page précédente)

```

x509Realm.grantedKeyStorePassphrase = {{password_grantedstore}}
x509Realm.trustedKeyStoreName = {{vitam_folder_conf}}/truststore_ihm-recette.jks
x509Realm.trustedKeyStorePassphrase = {{password_truststore}}
x509Realm.credentialsMatcher = $x509credentialsMatcher
securityManager.realm = $x509Realm
securityManager.subjectDAO.sessionStorageEvaluator.sessionStorageEnabled = false
[urls]
/v1/api/** = x509

{% else %}
# Objects and their properties are defined here,
# Such as the securityManager, Realms and anything
# else needed to build the SecurityManager
# credentialsMatcher
sha256Matcher = org.apache.shiro.authc.credential.Sha256CredentialsMatcher
iniRealm.credentialsMatcher = $sha256Matcher
# Cache Manager
builtInCacheManager = org.apache.shiro.cache.MemoryConstrainedCacheManager
# Security Manager
securityManager.cacheManager = $builtInCacheManager
sessionManager = org.apache.shiro.web.session.mgt.DefaultWebSessionManager
securityManager.sessionManager = $sessionManager
securityManager.sessionMode=native
securityManager.sessionManager.globalSessionTimeout = {{ vitam_struct.session_timeout_
↪ }}
securityManager.sessionManager.sessionIdUrlRewritingEnabled = false
securityManager.sessionManager.sessionIdCookie.secure = {{ vitam_struct.secure_cookie_
↪ }}
securityManager.rememberMeManager.cookie.secure = {{ vitam_struct.secure_cookie }}
securityManager.rememberMeManager.cookie.httpOnly = true
# Notice how we didn't define the class for the FormAuthenticationFilter ('authc') -
↪ it is instantiated and available already:
authc.loginUrl = /#!/login
[users]
# The 'users' section is for simple deployments
# when you only need a small number of statically-defined
# set of User accounts.
#username = password
{% for item in vitam_users %}
{% if item.role == "admin" %}
{{item.login}}={{item.password|hash('sha256')}}
{% endif %}
{% endfor %}
[roles]
# The 'roles' section is for simple deployments
# when you only need a small number of statically-defined
# roles.
[urls]
# make sure the end-user is authenticated. If not, redirect to the 'authc.loginUrl'
↪ above,
# and after successful authentication, redirect them back to the original account
↪ page they
# were trying to view:
/v1/api/login = anon
/v1/api/logout = logout
/v1/api/securemode = anon
/** = authc

```

(suite sur la page suivante)

```
{% endif %}
```

8.2.9.2.9 Fichier static-offer.json

```
{% if vitam.storageofferdefault.https_enabled==true %}
  {% set protocol = 'https' %}
{% else %}
  {% set protocol = 'http' %}
{% endif %}
[
{% for item in vitam_strategy %}
{
  "id" : "{{ item.name }}.service.{{ item.vitam_site_name |default(vitam_site_name) }}.{{ consul_domain }}",
  "baseUrl" : "{{ protocol }}://{{ item.name }}.service.{{ item.vitam_site_name |default(vitam_site_name) }}.{{ consul_domain }}:{{ vitam.storageofferdefault.port_service }}",
  {% if item.asyncRead is defined %} "asyncRead": {{item.asyncRead|lower}}, {% endif %}
  "parameters" : {
    {% if vitam.storageofferdefault.https_enabled==true %}
    "keyStore-keyPath": "{{ vitam_folder_conf }}/keystore_storage.p12",
    "keyStore-keyPassword": "{{ keystores.client_storage.storage }}",
    "trustStore-keyPath": "{{ vitam_folder_conf }}/truststore_storage.jks",
    "trustStore-keyPassword": "{{ truststores.client_storage }}"
    {% endif %}
  }
}
{% if not loop.last %},
{% endif %}
{% endfor %}
]
```

8.2.9.2.10 Fichier static-strategy.json

```
{
  "id" : "default",
  "hot" : {
    "copy" : {{ vitam_strategy|length }},
    "offers" : [
      {% for item in vitam_strategy %}
      {
        "id" : "{{ item.name }}.service.{{ item.vitam_site_name |default(vitam_site_name) }}.{{ consul_domain }}"
        {% if item.referent is defined %}{% if item.referent|lower == "true" %}, "referent" : true{% endif %}{% endif %}{% if item.status is defined %}, "status" : "{{ item.status|upper }}" {% endif %}{% if not loop.last %},{% endif %}
        {% endfor %}
      }
    ]
  }
}
```

8.2.9.2.11 Fichier storage-client.conf

```
serverHost: {{ vitam.storageengine.host }}
serverPort: {{ vitam.storageengine.port_service }}
```

8.2.9.2.12 Fichier storage.conf

```
urlWorkspace: {{ vitam.workspace | client_url }}
timeoutMsPerKB: 100
jettyConfig: jetty-config.xml
zippingDirecorty: {{ vitam_folder_data }}/storage_archives
loggingDirectory: {{ vitam_folder_log }}
```

8.2.9.2.13 Fichier storage-offer.conf

```
strategy_name=[{% for item in vitam_strategy %}"{{ item.name }}.service.{{ consul_
↳domain }}" {% if not loop.last %},{% endif %} {% endfor %}]
```

8.2.9.2.14 Fichier tnr.conf

```
urlWorkspace: {{ vitam.workspace | client_url }}
tenantsTest: [ "0" ]
vitamSecret: {{ plateforme_secret }}
tenants: [ "{{ vitam_tenant_ids | join(' ', ' ') }}" ]
adminTenant: {{ vitam_tenant_admin }}
```

8.2.9.3 Opérations

- Démarrage du service

En tant qu'utilisateur root : `systemctl start vitam-ihm-recette`

- Arrêt du service

En tant qu'utilisateur root : `systemctl stop vitam-ihm-recette`

- Sauvegarde du service

Ce service ne nécessite pas de sauvegarde particulière.

- Supervision du service

Contrôler le retour HTTP 200 sur l'URL <protocole web https ou https>://<host>:<port admin>/admin/v1/status

- Gestion des utilisateurs

Les utilisateurs sont actuellement gérés via le fichier `shiro.ini`, dans la section `[users]`.

- Créer un utilisateur

Lancer la commande shell suivante pour générer le mot de passe :

```
echo -n <motdepasse> | sha256sum
```

Copier le résultat.

Ensuite, éditer le fichier `/vitam/conf/ihm-recette/shiro.ini` et ajouter, dans la section `[users]`, la ligne suivante :

```
<login de l'utilisateur>=<résultat de la commande de génération de mot de_
↳passe précédente>
```

Pour terminer, relancer le service `vitam-ihm-recette` par la commande :

```
systemctl restart vitam-ihm-recette
```

- Supprimer un utilisateur

Dans la section `[users]`, enlever la ligne correspondant à l'utilisateur à supprimer. Pour terminer, relancer le service `vitam-ihm-recette` par la commande :

```
systemctl restart vitam-ihm-recette
```

- Exports

N/A

- gestion de la capacité

N/A

- actions récurrentes

- cas des batches

N/A

8.2.10 Ingest

8.2.10.1 Introduction

Ce document présente les configurations pour utiliser les différents modules de *ingest*.

8.2.10.2 ingest-external

8.2.10.2.1 Présentation

Ingest-external est le composant d'interface entre *VITAM* et un *SIA* client, permettant de réaliser des entrées d'archives dans *VITAM*.

Rôle :

- Exposer les API publiques du système
- Sécuriser l'accès aux API de VITAM

8.2.10.2.2 Configuration / fichiers utiles

Les fichiers de configuration sont gérés par les procédures d'installation ou de mise à niveau de l'environnement *VITAM*. Se référer au *DIN*.

Les fichiers de configuration sont définis sous `/vitam/conf/ingest-external`.

8.2.10.2.2.1 Fichier ingest-external.conf

```
path: {{ vitam_folder_data }}
jettyConfig: jetty-config.xml
authentication: false
tenantFilter : true
antiVirusScriptName: scan-{{ vitam_struct.antivirus }}.sh
timeoutScanDelay: {{ vitam_struct.timeoutScanDelay }}
baseUploadPath: {{ vitam_struct.upload_dir }}
successfulUploadDir: {{ vitam_struct.success_dir }}
failedUploadDir: {{ vitam_struct.fail_dir }}
fileActionAfterUpload: {{ vitam_struct.upload_final_action }}
```

Ce fichier contient un appel au shell d'antivirus (par défaut, ClamAV) ; se reporter au *DIN*.

Il est possible, dans le cas de fichiers SIP volumineux, d'héberger des fichiers directement dans ingest-external (valeur de la directive baseUploadPath). Ces fichiers doivent être accessibles et utilisables par le *user* système vitam.

Les options associées à cette fonctionnalité peuvent être paramétrées dans le fichier deployment/environment/group_vars/all/vitam_vars.yml avant installation de Vitam.

La directive fileActionAfterUpload accepte les valeurs :

- NONE : le fichier reste
- MOVE : déplace le fichiers vers les valeurs des directives successfulUploadDir (en cas de succès de l'ingest) et failedUploadDir (en cas de non-succès de l'ingest)
- DELETE : le fichier est supprimé en cas de succès de l'ingest uniquement

A charge à l'exploitant de bien gérer l'espace disque de ces répertoires (il faut penser aux ingests en échecs par exemple).

Se reporter au manuel de développement pour l'appel d'API associé.

8.2.10.2.2.2 Fichier ingest-internal-client.conf

```
serverHost: {{ vitam.ingestinternal.host }}
serverPort: {{ vitam.ingestinternal.port_service }}
```

8.2.10.2.2.3 Fichier internal-security-client.conf

```
serverHost: {{ vitam.security_internal.host }}
serverPort: {{ vitam.security_internal.port_service }}
secure: {{ vitam.security_internal.https_enabled }}
```

8.2.10.2.2.4 Fichier format-identifiers.conf

```
siegfried-local:
  type: SIEGFRIED
  client: http
  host: localhost
  port: {{ siegfried.port }}
  rootPath: {{ vitam_folder_data }}/
  versionPath: {{ vitam_folder_data }}/version/folder
```

8.2.10.2.2.5 Fichier functional-administration-client.conf

```
serverHost: {{ vitam.functional_administration.host }}
serverPort: {{ vitam.functional_administration.port_service }}
```

8.2.10.2.2.6 Fichier scan-clamav.sh

Ce script de *scan* appelle l'antivirus (par défaut, clamAV ; ce paramètre est surchargeable à l'installation ; se référer au terme 'DIN' pour plus de précisions) pour détecter les virus.

```
#!/bin/sh

#####
# Role: #
# Scan un single file using clamav anti-virus #
#####
# Args: #
# - file to scan #
#####
# Return: #
# - 0: scan OK - no virus #
RET_NOTVIRUS=0
# - 1: virus found and corrected #
RET_VIRUS_FOUND_FIXED=1
# - 2: virus found but not corrected #
RET_VIRUS_FOUND_NOTFIXED=2
# - 3: Fatal scan not performed #
RET_FAILURE=3
# stdout : names of virus found (1 per line) if virus found ; #
#          failure description if failure #
# stderr : full ouput of clamav #
#####

# Default return code : scan NOK
RET=3
OUTPUT_DIR=$(mktemp -d)
if [ $# -ne 1 ]; then # Argument number must be one
    echo "ERROR : $# parameter(s) provided, only one parameter is needed"
else # one argument, let's go
    if [ ! -f "$1" ]; then # if the file wich will be scan is existing, keep going
        echo "ERROR : \"$1\" doesn't exist"
    else
        clamscan -z --config-file=/etc/clamd.d/scan.conf "$1" 1> ${OUTPUT_
→DIR}/stdout 2> ${OUTPUT_DIR}/stderr # scanning the file and store the output OUTPUT
        RET=$? # return code of clamscan

        # Always output clamscan outputs to our own stderr
        (>&2 cat ${OUTPUT_DIR}/stdout ${OUTPUT_DIR}/stderr)

        if [ ${RET} -eq ${RET_VIRUS_FOUND_FIXED} ] ; then
            RET=2 # if virus found clamscan return 1; the script must_
→return 2

            (>&1 cat ${OUTPUT_DIR}/stdout | grep `basename ${1}` | cut -
→d ' ' -f 2) # sending the list of virus to our own stdout
            elif [ ${RET} -eq 2 ] ; then
```

(suite sur la page suivante)

(suite de la page précédente)

```

RET=3 # if scan not performed clamscan return 2; the script_
↪must return 3
        (>&1 cat ${OUTPUT_DIR}/stdout | grep `basename ${1}` | cut -
↪d ' ' -f 2-) # sending the failure reason to our own stdout
        fi

        if [ -f "${OUTPUT_DIR}/stdout" ]
        then
                rm ${OUTPUT_DIR}/stdout
        fi
        if [ -f "${OUTPUT_DIR}/stderr" ]
        then
                rm ${OUTPUT_DIR}/stderr
        fi
        fi
fi
rmdir ${OUTPUT_DIR}
exit ${RET}

```

8.2.10.2.3 Opérations

- Démarrage du service

En tant qu'utilisateur root : `systemctl start vitam-ingest-external`

- Arrêt du service

En tant qu'utilisateur root : `systemctl stop vitam-ingest-external`

- Sauvegarde du service

Ce service ne nécessite pas de sauvegarde particulière.

- Supervision du service

Contrôler le retour HTTP 200 sur l'URL <protocole web https ou https>://<host>:<port>/ingest-external/v1/status

Contrôler le retour HTTP 200 sur l'URL <protocole web https ou https>://<host>:<port>/admin>/admin/v1/status

- Exports

N/A

- gestion de la capacité

N/A

- actions récurrentes

- cas des batches

N/A

8.2.10.3 ingest-internal

8.2.10.3.1 Présentation

Rôle :

- Permettre l'entrée d'une archive SEDA dans le SAE

Fonctions :

- Upload HTTP de fichiers au format SEDA
- Sas de validation antivirus des fichiers entrants
- Persistance du SEDA dans workspace
- Lancement des workflows de traitements liés à l'entrée dans processing

8.2.10.3.2 Configuration / fichiers utiles

Les fichiers de configuration sont gérés par les procédures d'installation ou de mise à niveau de l'environnement *VITAM*. Se référer au *DIN*.

Les fichiers de configuration sont définis sous `/vitam/conf/ingest-internal`.

8.2.10.3.2.1 Fichier `ingest-internal.conf`

```
workspaceUrl: {{vitam.workspace | client_url}}
processingUrl: {{vitam.processing | client_url}}
jettyConfig: jetty-config.xml
```

Ce fichier précise les URLs pour les services « Processing » et « Workspace », et la configuration du serveur jetty.

8.2.10.3.2.2 Fichier `storage-client.conf`

```
serverHost: {{ vitam.storageengine.host }}
serverPort: {{ vitam.storageengine.port_service }}
```

8.2.10.3.3 Opérations

- Démarrage du service

En tant qu'utilisateur root : `systemctl start vitam-ingest-internal`

- Arrêt du service

En tant qu'utilisateur root : `systemctl stop vitam-ingest-internal`

- Sauvegarde du service

Ce service ne nécessite pas de sauvegarde particulière.

- Supervision du service

Contrôler le retour HTTP 200 sur l'URL `<protocole web https ou https>://<host>:<port>/ingest-internal/v1/status`

Contrôler le retour HTTP 200 sur l'URL `<protocole web https ou https>://<host>:<port admin>/admin/v1/status`

- Exports

N/A

- gestion de la capacité

N/A

- actions récurrentes

- cas des batches

N/A

8.2.11 Security-Internal

8.2.11.1 Introduction

Ce document présente la configuration pour le module security-internal.

8.2.11.2 security-internal-exploitation

Ce document spécifie la configuration (fichiers de config) pour lancer le services de security-internal.

1. Serveur security-internal :

Pour lancer le serveur security-internal, deux fichier config suivant sont nécessaires :

- security-internal.conf : Contient la configuration du serveur MongoDB, du serveur jetty, les tenants, ainsi que la configuration de l'authentification personae pour les permissions des endpoints externes de Vitam.
- jetty-config.xml : contenant le information pour lancer le serveur jetty de security-internal. Ce fichier de jetty précise aussi la configuration TSL du mode SSL du serveur : les keystores et password pour les load, les algorithmes de chiffrement supportés ...
- personal-certificate-permissions.conf : Configuration des permissions nécessitant une authentification personae ou ne nécessitant pas d'authentification personae.

2. Client security-internal :

Pour la création d'un client security-internal, nous avons besoin aussi le fichier de configuration internal-security-client.conf qui précise le serveur host et la porte du serveur ou le client se connectent pour les requêtes.

Tous ces fichiers de configuration seront mis dans le répertoire /vitam/config. Les exemples de ces fichiers se trouvent dans les répertoires de src/test/resources correspondants.

Les fichiers de configuration sont définis sous /vitam/conf/security-internal.

8.2.11.2.1 Fichier security-internal.conf

ce fichier permet de définir l'URL d'accès au access server.

```
# Configuration MongoDB
mongoDbNodes:
{% for host in groups['hosts-mongos-data'] %}
- dbHost: {{hostvars[host]['ip_service']}}
  dbPort: {{ mongodb.mongos_port }}
{% endfor %}
```

(suite sur la page suivante)

(suite de la page précédente)

```

dbName: identity
dbAuthentication: {{ mongodb.mongo_authentication }}
dbUserName: {{ mongodb['mongo-data'].securityInternal.user }}
dbPassword: {{ mongodb['mongo-data'].securityInternal.password }}

jettyConfig: jetty-config.xml

personalCertificatePermissionConfig: personal-certificate-permissions.conf

#Basic Authentication
adminBasicAuth:
- userName: {{ admin_basic_auth_user }}
  password: {{ admin_basic_auth_password }}

```

8.2.11.2.2 Fichier `personal-certificate-permissions.conf`

```

# Personal certification configuration for endpoint permissions

permissionsRequiringPersonalCertificate:

permissionsWithoutPersonalCertificate:
- 'dipexport:create'
- 'dipexportv2:create'
- 'dipexport:id:dip:read'
- 'logbookobjectslifecycles:id:read'
- 'logbookoperations:read'
- 'logbookoperations:id:read'
- 'logbookunitlifecycles:id:read'
- 'units:read'
- 'units:id:read:json'
- 'units:id:update'
- 'units:id:objects:read:json'
- 'units:id:objects:read:binary'
- 'units:update'
- 'unitsWithInheritedRules:read'
- 'units:rules:update'
- 'accesscontracts:create:json'
- 'accesscontracts:read'
- 'accesscontracts:id:read'
- 'accesscontracts:id:update'
- 'accessionregisters:read'
- 'accessionregisters:id:accessionregisterdetails:read'
- 'agencies:create'
- 'agencies:read'
- 'agencies:id:read'
- 'agenciesfile:check'
- 'agenciesreferential:id:read'
- 'audits:create'
- 'contexts:create:json'
- 'contexts:read'
- 'contexts:id:read'

```

(suite sur la page suivante)

(suite de la page précédente)

```

- 'contexts:id:update'
- 'distributionreport:id:read'
- 'formats:read'
- 'formats:create'
- 'formats:id:read'
- 'formatsfile:check'
- 'ingestcontracts:create:json'
- 'ingestcontracts:read'
- 'ingestcontracts:id:read'
- 'ingestcontracts:id:update'
- 'operations:read'
- 'operations:id:read:status'
- 'operations:id:read'
- 'operations:id:update'
- 'operations:id:delete'
- 'profiles:create:binary'
- 'profiles:create:json'
- 'profiles:read'
- 'profiles:id:read:json'
- 'profiles:id:update:binaire'
- 'profiles:id:read:binary'
- 'profiles:id:update:json'
- 'rules:read'
- 'rules:create'
- 'rules:id:read'
- 'rulesfile:check'
- 'rulesreport:id:read'
- 'rulesreferential:id:read'
- 'securityprofiles:create:json'
- 'securityprofiles:read'
- 'securityprofiles:id:read'
- 'securityprofiles:id:update'
- 'traceability:id:read'
- 'traceabilitychecks:create'
- 'workflows:read'
- 'ingests:create'
- 'ingests:local:create'
- 'ingests:id:archivetransfertreply:read'
- 'ingests:id:manifests:read'
- 'switchindex:create'
- 'reindex:create'
- 'evidenceaudit:check'
- 'archiveunitprofiles:create:binary'
- 'archiveunitprofiles:create:json'
- 'archiveunitprofiles:read'
- 'archiveunitprofiles:id:read:json'
- 'archiveunitprofiles:id:update:json'
- 'ontologies:create:binary'
- 'ontologies:create:json'
- 'ontologies:read'
- 'ontologies:id:read:json'
- 'ontologies:id:read:binary'
- 'ontologies:id:update:json'
- 'reclassification:update'
- 'rectificationaudit:check'
- 'storageaccesslog:read:binary'
- 'objects:read'

```

(suite sur la page suivante)

(suite de la page précédente)

```
- 'elimination:analysis'
- 'elimination:action'
- 'forcepause:check'
- 'removeforcepause:check'
- 'probativevalue:check'
- 'probativevalue:create'
- 'accessionregisterssymbolic:read'
- 'griffins:create'
- 'preservationScenarios:create'
- 'griffins:read'
- 'griffin:read'
- 'preservationScenarios:read'
- 'preservationScenario:read'
- 'preservation:update'
- 'batchreport:id:read'
- 'preservationreport:id:read'
- 'logbookoperations:create'
```

8.2.11.3 Opérations

- Démarrage du service

En tant qu'utilisateur root : `systemctl start vitam-security-internal`

- Arrêt du service

En tant qu'utilisateur root : `systemctl stop vitam-security-internal`

- Sauvegarde du service

Ce service ne nécessite pas de sauvegarde particulière.

- Supervision du service

Contrôler le retour HTTP 200 sur l'URL <protocole web https ou https>://<host>:<port>/vitam-security-internal/v1/status

Contrôler le retour HTTP 200 sur l'URL <protocole web https ou https>://<host>:<port admin>/admin/v1/status

- Exports

N/A

- gestion de la capacité

N/A

- actions récurrentes

- cas des batches

N/A

8.2.12 Logbook

8.2.12.1 Présentation

8.2.12.2 Logbook Exploitation

8.2.12.2.1 Configuration du Logbook

logbook.conf : fichier Yaml de configuration du serveur *logbook*. Celle-ci possède une propriété :

- **alertEvents** : configuration des alertes de sécurité

une alerte est déclenchée soit sur l'analyse du couple {evType,outCome} soit sur celle du {outDetail}

1. Dans le cas du déclenchement sur l'analyse du couple {evType, outCome}

```
- evType: 'CHECK_HEADER.CHECK_CONTRACT_INGEST'
  outcome: 'KO'
```

2. Dans le cas du déclenchement sur l'analyse du {outComeDetail}

```
- outDetail: 'CHECK_HEADER.CHECK_CONTRACT_INGEST.KO'
```

3. La liste des détections de l'alerte

- non conformité de la base des règles de gestion au référentiel enregistré (CHECK_RULES)
- refus d'entrée d'un SIP pour des raisons d'inadéquation de contrats (CHECK_HEADER.CHECK_CONTRACT_INGEST)
- soumission d'un SIP avec une classification incompatible avec la plateforme (CHECK_CLASSIFICATION_LEVEL)
- valeur de durée dans les règle de gestion inférieure à la durée minimum (CHECK_RULES.MAX_DURATION_EXCEEDS)
- refus d'un accès avec les droits personae (STP_PERSONAL_CERTIFICATE_CHECK)
- absence de sécurisation des journaux sur 12h (TODO)

Les fichiers de configuration sont gérés par les procédures d'installation ou de mise à niveau de l'environnement *VITAM*. Se référer au *DIN*.

Les fichiers de configuration sont définis sous /vitam/conf/logbook.

8.2.12.2.2 Fichier logbook.conf

```
# Configuration MongoDB
mongoDbNodes:
{% for server in groups['hosts-mongos-data'] %}
- dbHost: {{hostvars[server]['ip_service']}}
  dbPort: {{ mongodb.mongos_port }}
{% endfor %}
dbName: logbook
dbAuthentication: {{ mongodb.mongo_authentication }}
dbUserName: {{ mongodb['mongo-data'].logbook.user }}
dbPassword: {{ mongodb['mongo-data'].logbook.password }}
```

(suite sur la page suivante)

(suite de la page précédente)

```

jettyConfig: jetty-config.xml
pl2LogbookPassword: {{keystores.timestamping.secure_logbook}}
pl2LogbookFile: keystore_secure-logbook.pl2
workspaceUrl: {{ vitam.workspace | client_url }}
processingUrl: {{ vitam.processing | client_url }}

# ElasticSearch
clusterName: {{ vitam_struct.cluster_name }}
elasticsearchNodes:
{% for server in groups['hosts-elasticsearch-data'] %}
- hostName: {{hostvars[server]['ip_service']}}
  tcpPort: {{ elasticsearch.data.port_tcp }}
{% endfor %}

#Basic Authentication
adminBasicAuth:
- userName: {{ admin_basic_auth_user }}
  password: {{ admin_basic_auth_password }}

## Configuration for logbook coherence check
# list of operations that generate LFC
opWithLFC: [
  "PROCESS_SIP_UNITARY",
  "FILINGScheme",
  "HOLDINGScheme",
  "UPDATE_RULES_ARCHIVE_UNITS",
  "PROCESS_AUDIT",
  "STP_UPDATE_UNIT"]
# list of events not declared in wf
opEventsNotInWf: [
  "STP_SANITY_CHECK_SIP",
  "SANITY_CHECK_SIP",
  "CHECK_CONTAINER",
  "STP_UPLOAD_SIP"
]
# list of events to skip for OP-LFC check
opLfcEventsToSkip: [
  "STP_SANITY_CHECK_SIP", "SANITY_CHECK_SIP", "CHECK_CONTAINER", "STP_UPLOAD_SIP",
  ↳ "ATR_NOTIFICATION", "ROLL_BACK",
  "STORAGE_AVAILABILITY_CHECK", "ACCESSION_REGISTRATION",
  "ROLL_BACK", "ATR_NOTIFICATION", "COMMIT_LIFE_CYCLE_OBJECT_GROUP", "COMMIT_LIFE_
  ↳ CYCLE_UNIT",
  "LIST_OBJECTGROUP_ID", "REPORT_AUDIT",
  "LIST_ARCHIVE_UNITS", "LIST_RUNNING_INGESTS"]

# Configuration des alertes de securite
alertEvents:
- evType: 'CHECK_HEADER.CHECK_CONTRACT_INGEST'
  outcome: 'KO'
- evType: 'CHECK_RULES.MAX_DURATION_EXCEEDS'
  outcome: 'KO'
- evType: 'CHECK_RULES'
  outcome: 'KO'
- outDetail: 'CHECK_CLASSIFICATION_LEVEL.KO'
- outDetail: 'STP_PERSONAL_CERTIFICATE_CHECK.KO'

# Traceability params

```

(suite sur la page suivante)

(suite de la page précédente)

```
operationTraceabilityTemporizationDelay: {{ vitam.logbook.
↪operationTraceabilityTemporizationDelay }}
lifecycleTraceabilityTemporizationDelay: {{ vitam.logbook.
↪lifecycleTraceabilityTemporizationDelay }}
lifecycleTraceabilityMaxEntries: {{ vitam.logbook.lifecycleTraceabilityMaxEntries }}
```

8.2.12.2.3 Fichier functional-administration-client.conf

```
serverHost: {{ vitam.functional_administration.host }}
serverPort: {{ vitam.functional_administration.port_service }}
```

8.2.12.2.4 Fichier logbook-client.conf

```
serverHost: {{ vitam.logbook.host }}
serverPort: {{ vitam.logbook.port_service }}
```

8.2.12.2.5 Fichier securisationDaemon.conf

```
tenants: [ "{{ vitam_tenant_ids | join(' ', '') }}" ]
```

8.2.12.2.6 Fichier storage-client.conf

```
serverHost: {{ vitam.storageengine.host }}
serverPort: {{ vitam.storageengine.port_service }}
```

8.2.12.2.7 Fichier traceabilityAudit.conf

```
tenants: [ "{{ vitam_tenant_ids | join(' ', '') }}" ]
nbDay: 1
timesEachDay: 24
```

8.2.12.3 Opérations

- Démarrage du service

En tant qu'utilisateur root : `systemctl start vitam-logbook`

- Arrêt du service

En tant qu'utilisateur root : `systemctl stop vitam-logbook`

- Sauvegarde du service

Ce service ne nécessite pas de sauvegarde particulière.

- Supervision du service

Contrôler le retour HTTP 200 sur l'URL <protocole web https ou https>://<host>:<port>/logbook/v1/status

Contrôler le retour HTTP 200 sur l'URL <protocole web https ou https>://<host>:<port>/admin>/admin/v1/status

- Exports

N/A

- gestion de la capacité

N/A

- actions récurrentes

- cas des batches

N/A

8.2.13 Metadata

8.2.13.1 Présentation

8.2.13.2 Configuration / fichiers utiles

8.2.13.2.1 Fichier `metadata.conf`

```
workspaceUrl: {{vitam.workspace | client_url}}

# Configuration MongoDB
mongoDbNodes:
{% for server in groups['hosts-mongos-data'] %}
- dbHost: {{hostvars[server]['ip_service']}}
  dbPort: {{ mongodb.mongos_port }}
{% endfor %}
dbName: metadata
dbAuthentication: {{ mongodb.mongo_authentication }}
dbUserName: {{ mongodb['mongo-data'].metadata.user }}
dbPassword: {{ mongodb['mongo-data'].metadata.password }}

jettyConfig: jetty-config.xml

# Elasticsearch
clusterName: {{ vitam_struct.cluster_name }}
elasticsearchNodes:
{% for server in groups['hosts-elasticsearch-data'] %}
- hostName: {{hostvars[server]['ip_service']}}
  tcpPort: {{ elasticsearch.data.port_tcp }}
{% endfor %}

#Basic Authentication
adminBasicAuth:
- userName: {{ admin_basic_auth_user }}
  password: {{ admin_basic_auth_password }}
```

8.2.13.2.2 Fichier `functional-administration-client.conf`

```
serverHost: {{ vitam.functional_administration.host }}
serverPort: {{ vitam.functional_administration.port_service }}
```

8.2.13.2.3 Fichier `storage-client.conf`

```
serverHost: {{ vitam.storageengine.host }}
serverPort: {{ vitam.storageengine.port_service }}
```

8.2.13.3 Opérations

- Démarrage du service

En tant qu'utilisateur root : `systemctl start vitam-metadata`

- Arrêt du service

En tant qu'utilisateur root : `systemctl stop vitam-metadata`

- Sauvegarde du service

Ce service ne nécessite pas de sauvegarde particulière.

- Supervision du service

Contrôler le retour HTTP 200 sur l'URL <protocole web https ou https>://<host>:<port>/
metadata/v1/status

Contrôler le retour HTTP 200 sur l'URL <protocole web https ou https>://<host>:<port
admin>/admin/v1/status

- Exports

N/A

- gestion de la capacité

N/A

- actions récurrentes

- cas des batches

N/A

8.2.14 Processing

8.2.14.1 Introduction

8.2.14.1.1 But de cette documentation

Le but de cette documentation est d'expliquer la configuration et l'exploitation de ce module.

8.2.14.2 Processing

Nom de l'image docker : **processing**

Dans cette image est déployé le module processing

8.2.14.2.1 Configuration du worker

Dans `/vitam/conf` :

1. **processing.conf** : Fichier Yaml de configuration du server *processing*. Il possède une propriété :
 - **jettyConfig** : emplacement du fichier de configuration XML *jetty* (exemple `jetty-config.xml`)
 - **urlWorkspace** : URL d'accès au service distant *workspace* (exemple `http://localhost:8088`)
 - **urlMetadata** : URL d'accès au service distant *metadata* (exemple `http://localhost:8088`)
2. **logbook-client.conf** : Fichier de configuration du client qui communique avec le **logbook**. Il contient les propriétés suivantes :
 - **serverHost** : host distant du service logbook
 - **serverPort** : port distant du service logbook
3. **server-identity.conf** : identification du serveur
4. **logback.xml** : configuration des logs

8.2.14.2.2 Supervision du service

Contrôler le retour HTTP 200 et identité du serveur (cf *server-identity.conf*) sur l'URL <protocole web https ou https>://<host>:<port>/processing/v1/status

8.2.14.3 Configuration / fichiers utiles

Les fichiers de configuration sont gérés par les procédures d'installation ou de mise à niveau de l'environnement *VITAM*. Se référer au *DIN*.

Les fichiers de configuration sont définis sous `/vitam/conf/processing`.

8.2.14.3.1 Fichier `processing.conf`

```
urlMetadata: {{ vitam_struct | client_url }}
urlWorkspace: {{ vitam.workspace | client_url }}
jettyConfig: jetty-config.xml
workflowRefreshPeriod: 1
processingCleanerPeriod: 1
```

8.2.14.3.2 Fichier `version.conf`

```
binaryDataObjectVersions:
- BinaryMaster
- Dissemination
- Thumbnail
- TextContent
physicalDataObjectVersions:
- PhysicalMaster
- Dissemination
```

8.2.14.3.3 Fichier `storage-client.conf`

```
serverHost: {{ vitam.storageengine.host }}
serverPort: {{ vitam.storageengine.port_service }}
```

8.2.14.3.4 Fichier `metadata-client.conf`

```
serverHost: {{ vitam.metadata.host }}
serverPort: {{ vitam.metadata.port_service }}
```

8.2.14.4 Opérations

- Démarrage du service

En tant qu'utilisateur root : `systemctl start vitam-processing`

- Arrêt du service

En tant qu'utilisateur root : `systemctl stop vitam-processing`

- Sauvegarde du service

Ce service ne nécessite pas de sauvegarde particulière.

- Supervision du service

Contrôler le retour HTTP 200 sur l'URL <protocole web https ou https>://<host>:<port>/processing/v1/status

Contrôler le retour HTTP 200 sur l'URL <protocole web https ou https>://<host>:<port>admin>/admin/v1/status

- Exports

N/A

- gestion de la capacité

N/A

- actions récurrentes

- cas des batches

N/A

8.2.15 Storage

8.2.15.1 Introduction

8.2.15.1.1 But de cette documentation

Le but de cette documentation est d'expliquer la configuration et l'exploitation des modules :

- **storage-engine**
- **storage-offer-default**

8.2.15.2 storage-engine

8.2.15.2.1 Présentation

Rôle :

- Stockage des données (Méta Données, Objets Numériques et journaux SAE et de l'archive)

Fonctions :

- Utilisation de stratégie de stockage (abstraction par rapport aux offres de stockage sous-jacentes)
- Gestion des différentes offres de stockage

8.2.15.2.2 Storage Engine

Nom de l'image docker : **storage-engine**

Dans cette image sont déployés :

- le moteur de stockage (storage-engine)
- l'implémentation du driver correspondant à l'offre de stockage par défaut (storage-offer-default)

8.2.15.2.2.1 Configuration du moteur de stockage

Dans **/vitam/conf** :

1. **storage-engine.conf** : Fichier Yaml de configuration du server *storage-engine*. Il possède une propriété :
 - **urlWorkspace** : URL d'accès au service distant *workspace* (exemple <http://localhost:8088>)
2. **driver-location.conf** : Fichier Yaml de configuration du DriverManager, Il permet de définir l'emplacement où sont stockés les fichiers JAR contenant les implémentations des différents drivers pour les différentes offres. Il possède une seule propriété :
 - **driverLocation** : emplacement des jars (chemin absolu de préférence)
3. **driver-mapping.conf** : Fichier Yaml de configuration du DriverManager (persistance de l'association driver / offre). Pour le moment, ce fichier de configuration contient le chemin d'accès aux fichiers qui définissent le mapping driver<->offre, plus tard il évoluera sans doute pour prendre en compte des données en base et donc contenir la configuration d'accès à la base. Il contient deux propriétés :
 - **driverMappingPath** : Définit l'emplacement des fichiers de persistance (au jour d'aujourd'hui on a 1 seul driver/offre, donc 1 seul fichier de persistance sera présent). La propriété doit finir par « / ».
 - **delimiter** : Définit le « délimiteur » (CSV style) des fichiers.
4. **static-offer.json** : Contient la description de l'offre "default" au format JSON (un jour sera sans doute dans une base de données). En PJ un exemple de ce fichier. La propriété *baseUrl* et *parameters* nécessitent d'être remplacés. Et la propriété *parameters* doit contenir *keystore*, *trustore* et leur mot de passe que le storage driver va utiliser pour la vérification de l'authentification. Il s'agit de l'URL d'accès à l'offre de stockage "default". Exemple :

```
{
  "id" : "default",
  "baseUrl" : "https://localhost:8088",
  "parameters" : {
    "user" : "bob"
    "keyStore-keyPath": "src/test/resources/storage-test/tls/client/client.p12",
    "keyStore-keyPassword": "vitam2016",
    "trustStore-keyPath": "src/test/resources/storage-test/tls/server/truststore.jks",
    "trustStore-keyPassword": "tazerty",
    "referent": "true"
  }
}
```

To remove TLS support :

- change « https » to « http » in **baseUrl**

```
{
  "id" : "default",
  "baseUrl" : "http://localhost:8088",
  "parameters" : {
    "user" : "bob"
  }
}
```

To define « referent » offer :

- choose **exactly one** offer by adding parameter **referent**

```
[
  {
    "id" : "default",
    "baseUrl" : "http://localhost:8088",
    "parameters" : {
      "user" : "bob",
      "referent": "true"
    }
  },
  {
    "id" : "offer2",
    "baseUrl" : "http://localhost:8089",
    "parameters" : {
      "user" : "bob"
    }
  }
]
```

- change storage-default-offer.json to disable authentication

```
jettyConfig: jetty-config-nossl.xml
authentication : false
```

- change the jetty-config-nossl.xml of the offer (CAS Manager) to not include any TLS configuration

5. **static-strategy.json** : Contient les informations de la stratégie de stockage (1 seule pour le moment). Ce fichier n'est pas à modifier.

```
{
  "id" : "default",
  "hot" : {
    "copy" : 1,
    "offers" : [
      {"id" : "default"}
    ]
  }
}
```

6. **server-identity.conf** : identification du serveur

7. **logback.xml** : configuration des logs

8.2.15.2.2.2 Configuration du driver de l'offre de stockage par défaut

Dans **/vitam/data** :

1. **fr.gouv.vitam.storage.offers.workspace.driver.DriverImpl** : Il s'agit du fichier de persistance. Il contient l'identifiant de l'offre associée au driver (plus tard potentiellement DES offres associées) : « *default* ». Il DOIT être placé dans le répertoire défini dans le fichier *driver-mapping.conf*.

Dans **/vitam/lib** :

1. **storage-driver-default.jar** : Il s'agit d'un jar contenant l'implémentation du Driver vitam pour l'offre « *storage-offer-default* ». Ce jar DOIT être placé dans le dossier défini dans la propriété *driverLocation* du fichier *driver-location.conf*. Par défaut il est chargé en tant que dépendance du projet.

8.2.15.2.2.3 Supervision du service

Contrôler le retour HTTP 200 et identité du serveur (cf *server-identity.conf*) sur l'URL <protocole web https ou https>://<host>:<port>/storage/v1/status

8.2.15.2.3 Configuration / fichiers utiles

Les fichiers de configuration sont gérés par les procédures d'installation ou de mise à niveau de l'environnement *VITAM*. Se référer au *DIN*.

Les fichiers de configuration sont définis sous **/vitam/conf/storage-engine**.

8.2.15.2.3.1 Fichier **driver-location.conf**

```
driverLocation: {{vitam_folder_lib}}
```

8.2.15.2.3.2 Fichier **driver-mapping.conf**

```
driverMappingPath: {{vitam_folder_data}}/
delimiter: ;
```

8.2.15.2.3.3 Fichier static-offer.json

```
{% if vitam.storageofferdefault.https_enabled==true %}
  {% set protocol = 'https' %}
{% else %}
  {% set protocol = 'http' %}
{% endif %}
[
{% for item in vitam_strategy %}
{
  "id" : "{{ item.name }}.service.{{ item.vitam_site_name |default(vitam_site_name) }}.{{ consul_domain }}",
  "baseUrl" : "{{ protocol }}://{{ item.name }}.service.{{ item.vitam_site_name |default(vitam_site_name) }}.{{ consul_domain }}:{{ vitam.storageofferdefault.port_service }}",
  {% if item.asyncRead is defined %} "asyncRead": {{item.asyncRead|lower }}, {% endif %}
  "parameters" : {
    {% if vitam.storageofferdefault.https_enabled==true %}
    "keyStore-keyPath": "{{ vitam_folder_conf }}/keystore_storage.p12",
    "keyStore-keyPassword": "{{ keystores.client_storage.storage }}",
    "trustStore-keyPath": "{{ vitam_folder_conf }}/truststore_storage.jks",
    "trustStore-keyPassword": "{{ truststores.client_storage }}"
    {% endif %}
  }
}
{% if not loop.last %},
{% endif %}
{% endfor %}
]
```

8.2.15.2.3.4 Fichier static-strategy.json

```
{
  "id" : "default",
  "hot" : {
    "copy" : {{ vitam_strategy|length }},
    "offers" : [
      {% for item in vitam_strategy %}
      {
        "id" : "{{ item.name }}.service.{{ item.vitam_site_name |default(vitam_site_name) }}.{{ consul_domain }}"
        {% if item.referent is defined %}{% if item.referent|lower == "true" %}, "referent" : true{% endif %}{% endif %}{% if item.status is defined %}, "status" : "{{ item.status|upper }}" {% endif %}{% if not loop.last %},{% endif %}
      }
      {% endfor %}
    ]
  }
}
```

8.2.15.2.3.5 Fichier storage-engine.conf

```
urlWorkspace: {{ vitam.workspace | client_url }}
timeoutMsPerKB: {{ vitam.storageengine.timeoutMsPerKB }}
```

(suite sur la page suivante)

(suite de la page précédente)

```
jettyConfig: jetty-config.xml
zippingDirecorty: {{vitam_folder_data}}/storage_archives
loggingDirectory: {{vitam_folder_log}}
p12LogbookPassword: {{keystores.timestamping.secure_storage}}
p12LogbookFile: keystore_{{vitam_timestamp_usage}}.p12
storageTraceabilityOverlapDelay: {{ vitam.storageengine.
↪storageTraceabilityOverlapDelay }}
restoreBulkSize: {{vitam.storageengine.restoreBulkSize}}
minBatchThreadPoolSize: {{vitam.storageengine.minBatchThreadPoolSize}}
maxBatchThreadPoolSize: {{vitam.storageengine.maxBatchThreadPoolSize}}
batchDigestComputationTimeout: {{vitam.storageengine.batchDigestComputationTimeout}}
offerSynchronizationBulkSize: {{vitam.storageengine.offerSynchronizationBulkSize}}
offerSyncThreadPoolSize: {{vitam.storageengine.offerSyncThreadPoolSize}}

#Basic Authentication
adminBasicAuth:
- userName: {{ admin_basic_auth_user }}
  password: {{ admin_basic_auth_password }}
```

8.2.15.2.4 Opérations

- Démarrage du service

En tant qu'utilisateur root : `systemctl start vitam-storage`

- Arrêt du service

En tant qu'utilisateur root : `systemctl stop vitam-storage`

- Sauvegarde du service

Ce service ne nécessite pas de sauvegarde particulière.

- Supervision du service

Contrôler le retour HTTP 200 sur l'URL <protocole web https ou https>://<host>:<port>/storage/v1/status

Contrôler le retour HTTP 200 sur l'URL <protocole web https ou https>://<host>:<port>admin>/admin/v1/status

- Exports

N/A

- gestion de la capacité

N/A

- actions récurrentes

- cas des batches

N/A

8.2.15.2.4.1 access-log

Le log des accès est généré lors d'un accès à l'objet (fichier numérique), que ce soit par téléchargement de l'objet ou export d'un DIP. Les accès à l'unité archivistique ne sont pas concernés.

Exemple de log généré lors de l'export d'un DIP d'une unité archivistique ayant un GOT contenant un objet

```
{ "eventDateTime": "2019-01-11T12:50:53.344", "xRequestId":
  ↳ "aeaaaaaaaaachfmo4dabyw6aliht3q74aaaaaq", "applicationId": "MyApplicationId-ChangeIt",
  ↳ "objectIdentifier": "aeaaaaaaaaahk2vrsabz26alhywthyoaaaaba", "size": "11", "qualifier":
  ↳ "BinaryMaster", "version": "1", "contextId": "CT-000001", "contractId": "ContratTNR",
  ↳ "archivesId": "aeaqaiaaahk2vrsabz26alhythzbaaaaea" }
```

Structure des logs :

- « eventDateTime » : date et heure de l'accès au format AAAA-MM-JJTHH :MM :SS.[digits de millisecondes]
- « xRequestId » : identifiant de l'opération d'export du DIP
- « applicationId » : identifiant de l'application ayant demandé l'export du DIP
- « objectIdentifier » : identifiant de l'objet auquel on a accédé
- « size » : taille en octets de l'objet
- « qualifier » : usage de l'objet
- « version » : version de l'usage de l'objet
- « contextId » : identifiant du contexte utilisé pour l'accès
- « contractId » : identifiant du contrat utilisé pour l'accès
- « archivesId » : identifiant de l'unité archivistique dont dépend le groupe d'objets contenant l'objet auquel on a accédé

Selon le paramétrage du contrat d'accès (AccessLog ACTIVE/INACTIVE), l'accès à un objet sera journalisé ou non. Par défaut, l'accès n'est pas journalisé.

Pour l'heure système en cours, ces fichiers sont présents sur les machines hébergeant le composant **storage** sous l'arborescence `/vitam/log/storage/access-log/`. Chaque fichier est nommé tel que :

`<tenant>_<date>_<id opération>.log`

Le *timer* `systemD vitam-storage-accesslog-backup` effectue la pérenisation sur offre de ces fichiers chaque heure. Dès lors, les *accesslog* sont accessibles dans des *containers* nommés `<environnement>_<tenant>_storageaccesslog`.

Exemple en stockage filesystem pour un environnement nommé `int` : `/vitam/data/offer/container/int_<tenant>_storageaccesslog/`

8.2.15.3 offer

8.2.15.3.1 Présentation

Ce composant est une déclinaison des offres de stockage sur FileSystem et CEPH.

Rôle :

- Fournir une offre de stockage par défaut permettant la persistance des objets sur un système de fichier local

Fonctions :

- Offre de stockage fournie par défaut
- Stockage simple des objets numériques sur un système de fichiers local

Par tenant VITAM déclaré, 17 *containers* sont créés :

- accessionregisterdetail
- accessionregistersymbolic
- backup
- backup_operation
- check_logbookreports
- dip
- distribution_reports
- logbook
- manifest
- object
- objectGroup
- report
- rules
- storageaccesslog
- storagelog
- storagetraceability
- unit

selon la norme <vitam_site_name>_<tenant>_<container> (R9 et plus) ou <tenant>_<container> (R7 et migrations depuis R7).

8.2.15.3.2 Storage Offer Default

Nom de l'image docker : **storage-offer-default**

Dans cette image est déployée l'offre de stockage par défaut utilisant le workspace.

8.2.15.3.2.1 Configuration de l'offre de stockage

1. **default-storage.conf** : Fichier Yaml de configuration du service. Contient les propriétés suivantes :

- **contextPath** : context path du server (mettre / par défaut)
- **storagePath** : chemin sur le filesystem sur lequel sont stockés les objets (/vitam/data).

2. **server-identity.conf** : identification du serveur

3. **logback.xml** : configuration des logs

8.2.15.3.2.2 Supervision du service

Contrôler le retour HTTP 200 et identité du serveur (cf *server-identity.conf*) sur l'URL <protocole web https ou https>://<host>:<port>/offer/v1/status

8.2.15.3.3 Configuration / fichiers utiles

Les fichiers de configuration sont gérés par les procédures d'installation ou de mise à niveau de l'environnement *VITAM*. Se référer au [DIN](#).

Les fichiers de configuration sont définis sous /vitam/conf/offer.

8.2.15.3.3.1 Fichier default-offer.conf

```
contextPath: /
# Smile : TODO : remove storagePath from this file
storagePath: {{vitam_folder_data}}
jettyConfig: jetty-config.xml
authentication : {{ vitam_struct.https_enabled }}
# Configuration MongoDB
mongoDbNodes:
{% for server in groups['hosts-mongos-offer'] %}
{% if hostvars[server]['mongo_cluster_name'] == offer_conf or inventory_hostname ==
↪ 'localhost' %}
- dbHost: {{hostvars[server]['ip_service']}}
  dbPort: {{ mongodb.mongos_port }}
{% endif %}
{% endfor %}
dbName: offer
dbAuthentication: {{ mongodb.mongo_authentication }}
dbUserName: {{ mongodb[offer_conf].offer.user }}
dbPassword: {{ mongodb[offer_conf].offer.password }}
```

8.2.15.3.3.2 Fichier default-storage.conf

```
provider: {{ vitam_offers[offer_conf]["provider"] }}

{% if vitam_offers[offer_conf]["provider"] in ["filesystem","filesystem-hash"] %}
storagePath: {{vitam_folder_data}}
{% endif %}

{% if vitam_offers[offer_conf]["provider"] in ["openstack-swift","openstack-swift-v2",
↪ "openstack-swift-v3"] %}
swiftKeystoneAuthUrl: {{ vitam_offers[offer_conf]["swiftKeystoneAuthUrl"] | default("
↪ ") }}
swiftDomain: {{ vitam_offers[offer_conf]["swiftDomain"] | default("") }}
swiftProjectName: {{ vitam_offers[offer_conf]["swiftProjectName"] | default("") }}
swiftUser: {{ vitam_offers[offer_conf]["swiftUser"] | default("") }}
swiftPassword: {{ vitam_offers[offer_conf]["swiftPassword"] | default("") }}
swiftUrl: {{ vitam_offers[offer_conf]["swiftUrl"] | default("") }}
swiftTrustStore: {{vitam_folder_conf}}/truststore-{{ vitam_struct.vitam_component }}.
↪ jks
swiftTrustStorePassword: {{ password_truststore }}
swiftMaxConnectionsPerRoute: {{ vitam_offers[offer_conf]["swiftMaxConnectionsPerRoute
↪ "] | default(200) }}
swiftMaxConnections: {{ vitam_offers[offer_conf]["swiftMaxConnections"] |
↪ default(1000) }}
swiftConnectionTimeout: {{ vitam_offers[offer_conf]["swiftConnectionTimeout"] |
↪ default(200000) }}
swiftReadTimeout: {{ vitam_offers[offer_conf]["swiftReadTimeout"] | default(200000) }}
swiftHardRenewTokenDelayBeforeExpireTime: {{ vitam_offers[offer_conf][
↪ "swiftHardRenewTokenDelayBeforeExpireTime"] | default(60) }}
swiftSoftRenewTokenDelayBeforeExpireTime: {{ vitam_offers[offer_conf][
↪ "swiftSoftRenewTokenDelayBeforeExpireTime"] | default(300) }}
{% endif %}
```

(suite sur la page suivante)

(suite de la page précédente)

```

{% if vitam_offers[offer_conf]["provider"] == "amazon-s3-v1" %}
s3RegionName: {{ vitam_offers[offer_conf]["s3RegionName"] }}
s3Endpoint: {{ vitam_offers[offer_conf]["s3Endpoint"] }}
s3AccessKey: {{ vitam_offers[offer_conf]["s3AccessKey"] }}
s3SecretKey: {{ vitam_offers[offer_conf]["s3SecretKey"] }}
s3PathStyleAccessEnabled: {{ vitam_offers[offer_conf]["s3PathStyleAccessEnabled"] |
↳default(true) }}
s3SignerType: {{ vitam_offers[offer_conf]["s3SignerType"] | default("") }}
s3MaxConnections: {{ vitam_offers[offer_conf]["s3MaxConnections"] | default(50) }}
s3ConnectionTimeout: {{ vitam_offers[offer_conf]["s3ConnectionTimeout"] |
↳default(10000) }}
s3SocketTimeout: {{ vitam_offers[offer_conf]["s3SocketTimeout"] | default(50000) }}
s3RequestTimeout: {{ vitam_offers[offer_conf]["s3RequestTimeout"] | default(0) }}
s3ClientExecutionTimeout: {{ vitam_offers[offer_conf]["s3ClientExecutionTimeout"] |
↳default(0) }}
s3TrustStore: {{vitam_folder_conf}}/truststore_{{ vitam_struct.vitam_component }}.jks
s3TrustStorePassword: {{ password_truststore }}
{% endif %}

{% if vitam_offers[offer_conf]["provider"] in ["tape-library"] %}
tapeLibraryConfiguration:
  inputFileStorageFolder: "{{ vitam_folder_data }}/offer/inputFiles"
  inputTarStorageFolder: "{{ vitam_folder_data }}/offer/inputTars"
  outputTarStorageFolder: "{{ vitam_folder_data }}/offer/outputTars"
  maxTarEntrySize: {{ vitam_offers[offer_conf]["tapeLibraryConfiguration"] [
↳"maxTarEntrySize"] | default(100000) }}
  maxTarFileSize: {{ vitam_offers[offer_conf]["tapeLibraryConfiguration"] [
↳"maxTarFileSize"] | default(1000000) }}
  useSudo: {{ vitam_offers[offer_conf]["tapeLibraryConfiguration"] ["useSudo"] |
↳default('false') }}
  forceOverrideNonEmptyCartridges: {{ vitam_offers[offer_conf] [
↳"tapeLibraryConfiguration"] ["forceOverrideNonEmptyCartridges"] | default('false') }}

  topology:
    buckets:
{% for bucket in vitam_offers[offer_conf]["topology"]["buckets"] %}
      {{bucket.name}}:
        tenants: {{bucket.tenants}}
        tarBufferingTimeoutInMinutes: {{bucket.tarBufferingTimeoutInMinutes}}
{% endfor %}

    tapeLibraries:
{% for library in vitam_offers[offer_conf]["tapeLibraries"] %}
      {{ library.name }}:
        robots:
{% for robot in library.robots %}
          -
            device: {{ robot.device }}
            mtxPath: "{{ robot.mtxPath }}"
            timeoutInMilliseconds: {{ robot.timeoutInMilliseconds }}
{% endfor %}
        drives:
{% for drive in library.drives %}
          -
            index: {{ drive.index }}
            device: {{ drive.device }}
            mtPath: "{{ drive.mtPath }}"

```

(suite sur la page suivante)

(suite de la page précédente)

```

        ddPath: "{{ drive.ddPath }}"
        tarPath: "{{ drive.tarPath }}"
        timeoutInMilliseconds: {{ drive.timeoutInMilliseconds }}
    {% endfor %}
{% endfor %}
{% endif %}

```

L'arborescence de stockage des fichiers dans l'offre est décrite dans le [DAT](#).

8.2.15.3.4 Opérations

- Démarrage du service

En tant qu'utilisateur root : `systemctl start vitam-offer`

- Arrêt du service

En tant qu'utilisateur root : `systemctl stop vitam-offer`

- Sauvegarde du service

Ce service ne nécessite pas de sauvegarde particulière.

- Supervision du service

Contrôler le retour HTTP 200 sur l'URL <protocole web https ou https>://<host>:<port>/offer/v1/status

Contrôler le retour HTTP 200 sur l'URL <protocole web https ou https>://<host>:<port>admin>/admin/v1/status

- Exports

N/A

- gestion de la capacité

N/A

- actions récurrentes

- cas des batches

N/A

8.2.16 Technical administration

8.2.16.1 Présentation

8.2.17 Worker

8.2.17.1 Introduction

Le but de cette documentation est d'expliquer la configuration et l'exploitation du module **worker**.

8.2.17.2 Configuration / fichiers utiles

Les fichiers de configuration sont gérés par les procédures d'installation ou de mise à niveau de l'environnement [VITAM](#). Se référer au [DIN](#).

Les fichiers de configuration sont définis sous `/vitam/conf/worker`.

8.2.17.2.1 Fichier batch-report-client.conf

```
serverHost: {{ vitam.batchreport.host }}
serverPort: {{ vitam.batchreport.port_service }}
secure: {{ vitam.batchreport.https_enabled }}
```

8.2.17.2.2 Fichier format-identifiers.conf

Ce fichier permet de définir l'URL d'accès à Siegfried.

```
siegfried-local:
  type: SIEGFRIED
  client: http
  host: localhost
  port: {{ siegfried.port }}
  rootPath: {{ vitam_folder_tmp }}/
  versionPath: {{ vitam_folder_data }}/version/folder
```

8.2.17.2.3 Fichier functional-administration-client.conf.j2

Ce fichier permet de définir l'accès à functional-administration.

```
serverHost: {{ vitam.functional_administration.host }}
serverPort: {{ vitam.functional_administration.port_service }}
```

8.2.17.2.4 Fichier metadata-client.conf

Ce fichier permet de définir l'accès au metadata.

```
serverHost: {{ vitam.metadata.host }}
serverPort: {{ vitam.metadata.port_service }}
```

8.2.17.2.5 Fichier storage-client.conf

Ce fichier permet de définir l'accès au storage.

```
serverHost: {{ vitam.storageengine.host }}
serverPort: {{ vitam.storageengine.port_service }}
```

8.2.17.2.6 Fichier verify-timestamp.conf

```
#Configuration - verify timestamp
p12LogbookPassword: {{ keystores.timestamping.secure_logbook }}
p12LogbookFile: keystore_secure-logbook.p12
```

8.2.17.2.7 Fichier `version.conf`

binaryDataObjectVersions:

- BinaryMaster
- Dissemination
- Thumbnail
- TextContent

physicalDataObjectVersions:

- PhysicalMaster
- Dissemination

8.2.17.2.8 Fichier `worker.conf`

Ce fichier permet de définir le paramétrage du composant worker.

```
# Configuration processing
# HERE MUST BE MY (WORKER) current configuration
registerServerHost: {{ ip_service }}
registerServerPort: {{ vitam_struct.port_service }}
# Configuration handler
processingUrl: {{vitam.processing | client_url}}
urlMetadata: {{vitam.metadata | client_url}}
urlWorkspace: {{vitam.workspace | client_url}}
# Configuration jetty
jettyConfig: jetty-config.xml
#Configuration parallele
capacity: {{vitam_worker_capacity}}
{% if vitam_worker_workerFamily is defined %}
workerFamily: {{vitam_worker_workerFamily}}
{% endif %}
```

Paramètres obligatoires :

- **processingUrl** : URL de connexion au compsoant Vitam processing
- **urlMetadata** : URL de connexion au composant VITAM metadata
- **urlWorkspace** : URL de connexion au composant VITAM workspace
- **registerServerHost** : host ou le worker déployé
- **registerServerPort** : port ou le worker déployé
- **jettyConfig** : le fichier config jetty associé au service du worker

Paramètres optionnels :

- **workerFamily** : la famille dont le worker appartenant en fonction de tache exécutée
- **capacity** : capacité du worker en mode parallèle de tache (par défaut à 1 dans l'ansiblerie, si non définie)

8.2.17.3 Opérations

- Démarrage du service

En tant qu'utilisateur root : `systemctl start vitam-workspace`

- Arrêt du service

En tant qu'utilisateur root : `systemctl stop vitam-workspace`

- Sauvegarde du service

Ce service ne nécessite pas de sauvegarde particulière.

- Supervision du service

Contrôler le retour HTTP 200 sur l'URL <protocole web https ou https>://<host>:<port>/workspace/v1/status

Contrôler le retour HTTP 200 sur l'URL <protocole web https ou https>://<host>:<port>admin>/admin/v1/status

- Exports

N/A

- gestion de la capacité

N/A

- actions récurrentes

- cas des batches

N/A

8.2.18 Workspace

8.2.18.1 Présentation

Rôle :

- Fourniture d'un espace pour l'échange de fichiers (et faire un appel par pointeur lors des appels entre composants) entre les différents composants de Vitam

Fonctions :

- Utilisation du moteur de stockage dans un mode minimal (Opérations CREATE, READ, DELETE sur 1 seule offre de stockage)

8.2.18.2 Configuration / fichiers utiles

Les fichiers de configuration sont gérés par les procédures d'installation ou de mise à niveau de l'environnement *VITAM*. Se référer au *DIN*.

Les fichiers de configuration sont définis sous /vitam/conf/workspace.

8.2.18.2.1 Fichier workspace.conf

```
storagePath: {{vitam_folder_data}}
jettyConfig: jetty-config.xml
provider: filesystem
```

8.2.18.3 Opérations

- Démarrage du service

En tant qu'utilisateur root : `systemctl start vitam-workspace`

- Arrêt du service

En tant qu'utilisateur root : `systemctl stop vitam-workspace`

- Sauvegarde du service

Ce service ne nécessite pas de sauvegarde particulière.

- Supervision du service

Contrôler le retour HTTP 200 sur l'URL `<protocole web https ou https>://<host>:<port>/workspace/v1/status`

Contrôler le retour HTTP 200 sur l'URL `<protocole web https ou https>://<host>:<port>/admin>/admin/v1/status`

- Exports

N/A

- gestion de la capacité

N/A

- actions récurrentes

- cas des batches

N/A

Intégration d'une application externe dans Vitam

9.1 Prérequis

L'application externe devra être en mesure de requêter les composants *VITAM* `ingest-external` et `access-external` sur leurs ports de service respectifs par le protocole HTTPS.

Il faut donc prévoir une ouverture de flux réseau pour le protocole TCP (selon l'infrastructure en place) sur les ports de service des composants *VITAM* `ingest-external` et `access-external`.

La sécurisation des connexions HTTP avec les applications externes est déléguée aux composants *VITAM* `ingest-external` et `access-external` (ou bien éventuellement à un reverse-proxy, selon l'infrastructure de déploiement *VITAM* retenue).

La création d'un certificat TLS client pour l'application externe est requise afin de permettre l'habilitation et l'authentification applicative des requêtes émises depuis l'application externe vers les composants *VITAM* `ingest-external` et `access-external`.

9.2 Intégration de certificats clients de VITAM

9.2.1 Authentification applicative SIA

Le certificat client de l'application externe devra également être ajouté dans la base de données du composant `security-internal`, afin de permettre la gestion des habilitations et l'authentification applicative de l'application externe.

- Editer le fichier `environments/group_vars/all/vitam_security.yml` et ajouter le(s) entrée(s) supplémentaire(s) (sous forme répertoire/fichier.crt) dans la directive `admin_context_certs`
- Déposer le certificat client de l'application dans `environments/certs/client-external/clients/external`

Avertissement : Bien vérifier qu'aucun certificat ou <i>CA</i> non-souhaité ne soit présent dans <code>environments/certs/*</code>

Lorsque l'authentification du client, par le protocole TLS, est réalisée, la **CA** du certificat client doit être déployée dans le *truststore* des composants **VITAM** `ingest-external` et `access-external`. SI le certificat client est passé dans le *header* http (cas de l'authentification applicative par *header* http X-SSL-CLIENT-CERT), le certificat client n'est alors pas utilisé dans la négociation TLS et il n'est donc pas nécessaire d'inclure la **CA** associée dans le *truststore* des composants **VITAM** `ingest-external` et `access-external`.

Vitam fournit une arborescence de répertoires permettant d'ajouter automatiquement la chaîne de certification client dans les bons *keystores*.

- Déposer les certificats de la chaîne de certification client dans *environments/certs/client-external/ca*. Ensuite, si ce n'est pas déjà fait, il reste à positionner les autres certificats, puis générer les keystores. Pour cela, se référer au *DIN*, chapitre concernant la gestion des certificats.

9.2.2 Authentification *personae*

Les exemple suivants permettent d'ajouter ou supprimer un certificat présent sous `/path/to/certificate`.

9.2.2.1 Ajout d'un certificat pour l'authentification Personae

```
curl -XPOST -H "Content-type: application/octet-stream" --data-binary @/path/to/
↪certificate 'http://<ip admin security-internal>:<port admin security-internal>/v1/
↪api/personalCertificate'
```

9.2.2.2 Suppression d'un certificat pour l'authentification Personae

```
curl -XDELETE -H "Content-type: application/octet-stream" --data-binary @/path/to/
↪certificate 'http://<ip admin security-internal>:<port admin security-internal>/v1/
↪api/personalCertificate'
```

9.3 Déploiement des keystores

9.3.1 Vitam n'est pas encore déployé

Déployer Vitam en suivant la procédure indiquée dans le *DIN*.

9.3.2 Vitam est déjà déployé

Suivre la procédure de la section *Mise à jour des certificats* (page 9).

10.1 Analyse de premier niveau

Cette section a pour but de présenter les premiers outils à utiliser pour réaliser une analyse de premier niveau, en cas de problème avec la solution logicielle *VITAM*.

10.1.1 Etat par Consul

Se connecter à l'IHM de Consul et recenser les états des composants de la solution logicielle *VITAM*.

Services

<div> All (206) ✓ Passing (206) ⚠ Warning (0) ✖ Critical (0) </div> <div> <input type="text" value="Search by name"/> <input type="button" value="Q"/> </div>		
Service	Node Health	Tags
access-external	✓ 4	
access-internal	✓ 4	
cerebro	✓ 3	
consul	✓ 6	
elastic-kibana-interceptor	✓ 4	
elasticsearch-data	✓ 12	
elasticsearch-log	✓ 12	
functional-administration	✓ 4	
ihm-demo	✓ 4	
ihm-recette	✓ 4	
ingest-external	✓ 6	
ingest-internal	✓ 4	

A l'heure actuelle, tous les composants doivent avoir un statut de couleur verte. Si ce n'est pas le cas :

1. seul un composant est KO, alors redémarrer le composant incriminé
2. si plusieurs services sont KO, suivre la procédure de redémarrage de VITAM
3. si tous les « check-DNS » (visible dans le détail des checks de chaque service) sont KO, s'assurer que, sur les machines hébergeant VITAM, le fichier `/etc/resolv.conf` contient, en début de fichier, la ligne :
`nameserver 127.0.0.1.`

10.1.2 Etat par Kibana

Se connecter à Kibana, aller dans « Dashboards ». Cliquer sur le bouton « Load Saved Dashboard » et sélectionner « Composants VITAM ». Eventuellement, changer la résolution (en haut à droite, par défaut, réglé sur les 15 dernières minutes).

Sur « pie-logback-error-level », cliquer sur la section de camembert d'intérêt (ERROR) et regarder, en bas de page, les éventuelles erreurs remontées dans Kibana.

10.2 Playbook ansible pour échanger avec le support

Afin de simplifier et minimiser les échanges, un playbook à fins d'exploitation/support a été développé.

Celui-ci comprend :

- la récupération des informations machines de la solution logicielle *VITAM*
- l'état Consul des composants
- la récupération des traces applicatives (fichiers log) de moins de 2 jours
- l'état des clusters Elasticsearch

- la possibilité, au choix de l'exploitant, de fournir également les clés publiques des certificats et compacte l'ensemble des données collectées, tout en purgeant le répertoire temporaire de récupération des données. Ce fichier est alors à envoyer par *mail* au support.

La commande pour générer le fichier est à lancer depuis le répertoire `deployment` :

```
ansible-playbook -i environments/<fichier d'inventaire> ansible-vitam-exploitation/
troubleshoot.yml --vault-password-file vault_pass.txt
```

Questions Fréquemment Posées

11.1 Présentation

Cette section a vocation à répertorier les différents problèmes rencontrés et apporter la solution la plus appropriée ; elle est amenée à être régulièrement mise à jour pour répertorier les problèmes rencontrés.

11.2 Retour d'expérience / cas rencontrés

11.2.1 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 :

- Centos ¹⁵
- Debian ¹⁶

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

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

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

Lors de la perte d'un noeud d'un cluster elasticseach, puis du retour de ce noeud, certains shards d'elasticseach 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

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

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`):

```
{
  "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](#)¹⁷.

11.2.4 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](#)¹⁸). 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.

11.2.5 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](#)¹⁹ et [mongotop](#)²⁰.

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

<https://www.elastic.co/guide/en/elasticsearch/reference/current/cluster-reroute.html>
<https://www.elastic.co/guide/en/elasticsearch/reference/1.4/states.html>
<https://docs.mongodb.com/manual/reference/program/mongostat/>
<https://docs.mongodb.com/manual/reference/program/mongotop/>

- 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** ²¹ :

```
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**²²) :

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

11.2.6 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 :

- `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.

²¹<https://docs.mongodb.com/manual/tutorial/troubleshoot-replica-sets>
²²<https://docs.mongodb.com/manual/reference/program/mongoperf/>

11.2.7 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.

11.2.8 Problème d'ingest et/ou d'access

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

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

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

11.2.9 Erreur d'inconsistance des données MongoDB / ES

En cas de détection d'un problème de synchronisation des données entre les bases de données Elasticsearch-data (cluster d'indexation dédié aux données métier) et les bases de données MongoDB-data (replicaset MongoDB stockant les données métier de Vitam) avec un message d'erreur du type : « An internal data consistency error has been detected », la procédure suivante pourra être appliquée : *Réindexation* (page 29).

12.1 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.

12.2 Gestion des anomalies en production

Les anomalies empêchant le bon fonctionnement de la solution VITAM déjà déployée dans un système en production sont gérées par le programme VITAM selon un processus dédié. Il reprend la terminologie du « Contrat de Service VITAM ».

12.2.1 Numérotation des versions

A partir de la version 1 de la solution VITAM, la numérotation des versions du logiciel est du type *X.Y.Z(-P)* selon les principes suivants :

- X : version majeure de la solution VITAM. Elle suit le calendrier des versions majeures, construit de concert avec les partenaires.
- Y : version mineure de la solution VITAM. Elle suit le calendrier des itérations, typiquement une itération dure trois semaines.
- Z : version bugfix de la solution VITAM. Elle suit le calendrier des itérations, typiquement une itération à chaque trois semaines.
- Seules les versions maintenues continuent de bénéficier de nouvelles versions bugfix.
- P : patch de la solution VITAM. Un patch correspond à la mise à disposition, entre deux releases, de binaires et/ou fichiers de configuration et de déploiement, pour corriger des bugs bloquants.
- Seules les versions maintenues continuent de bénéficier de patches.

12.2.2 Mise à disposition du logiciel

La solution VITAM est mise à disposition des partenaires selon le calendrier suivant :

- Des *releases* sont mises à disposition des partenaires et du grand public régulièrement, typiquement une release pour cinq itérations de développement. Il s'agit alors de la version mineure courante. Pour rappel, la version 1.0.0 correspond à la release 6 (*R6*).
- Les versions *bugfix* de chaque version maintenue sont mises à disposition des partenaires et du grand public régulièrement, à chaque itération (s'il y a eu des anomalies corrigées dans la période).
- Les patches de chaque version maintenue sont mis à disposition des partenaires à chaque fois qu'une anomalie de production critique est identifiée et corrigée. Les correctifs correspondant aux patches sont ensuite inclus dans une version bugfix ultérieure.

12.2.3 Gestion des patches

L'objectif d'un patch est de rétablir au plus vite le fonctionnement en production des systèmes partenaires. La livraison se limite ainsi aux packages (RPM / DEB) concernés par la correction, avec les fichiers de déploiement et de configuration nécessaires. Les instructions pour « patcher » l'appliquatif sont également mises à disposition, en fonction du périmètre impacté (simple arrêt / relance ; purges ; scripts de déploiement...).

Les patches sont mis à disposition des partenaires sur un dépôt en ligne. L'objectif est d'offrir la possibilité pour les partenaires d'automatiser la récupération des packages mis à jour, et éventuellement de pouvoir reconstituer un packaging complet de Vitam.

Note : Ce choix de gestion de patches implique des numéros de version qui pourront être différents entre chaque paquet. Le réalignement se fait au niveau des versions bugfix ou mineures.

La mise à disposition du code source du patch est considérée comme moins critique et se réalise dans un second temps, sur Github.

Table des figures

1	Vue d'ensemble d'un déploiement <i>VITAM</i> : zones, composants	7
---	--	---

Liste des tableaux

1	Documents de référence VITAM	2
---	--	---

A

API, [2](#)
AU, [2](#)

B

BDD, [3](#)

C

CA, [3](#)
CAS, [3](#)
CCFN, [3](#)
CN, [3](#)
COTS, [3](#)
CRL, [3](#)

D

DAT, [3](#)
DC, [3](#)
DEX, [3](#)
DIN, [3](#)
DMV, [3](#)
DNS, [3](#)
DNSSEC, [3](#)
DSL, [3](#)
DUA, [3](#)

E

EBIOS, [3](#)
ELK, [3](#)

I

IHM, [3](#)
IP, [3](#)

J

JRE, [3](#)
JVM, [3](#)

L

LAN, [3](#)

LFC, [3](#)
LTS, [3](#)

M

M2M, [3](#)
MitM, [3](#)

N

NoSQL, [4](#)

O

OAIS, [4](#)
OS, [4](#)
OWASP, [4](#)

P

PCA, [4](#)
PDMA, [4](#)
PKI, [4](#)
PRA, [4](#)

R

REST, [4](#)
RGI, [4](#)
RPM, [4](#)

S

SAE, [4](#)
SEDA, [4](#)
SGBD, [4](#)
SIA, [4](#)
SIEM, [4](#)
SIP, [4](#)
SSH, [4](#)
Swift, [4](#)

T

TNR, [4](#)
TTL, [4](#)

U

UDP, [4](#)

UID, [4](#)

V

VITAM, [4](#)

W

WAF, [4](#)

WAN, [4](#)