Outillage "open-source" pour de l'analyse post-mortem et réponse aux incidents

Emmanuel Mesnard
RSSI - Université de Reims Champagne-Ardenne
Direction du Numérique
Campus du Moulin de la Housse
51100 REIMS

Corentin Garcia
Alternant en Master 2 en cybersécurité (2020-2021) - Université de Reims Champagne-Ardenne
Direction du Numérique
Campus du Moulin de la Housse
51100 REIMS

Résumé

Les incidents de sécurité se multiplient depuis des années à un rythme effréné. Bien souvent, l'urgence est de remettre en état rapidement, passant sous le radar le besoin d'investigation, pour déterminer l'origine d'un incident. Il est donc important de s'intéresser à la notion de DFIR (Digital Forensic and Incident Response).

Depuis 2019, les retours sur les listes communautaires montrent que le besoin opérationnel reste toujours d'actualité afin de savoir comment rechercher des éléments de preuve d'un poste compromis.

Nous vous présenterons dans cet exposé l'objectif d'avoir un outillage "open-source" possible afin de pouvoir commencer à faire de l'analyse post-mortem et de la réponse aux incidents.

Nous essayerons d'exposer une méthodologie envisageable dans cette démarche de collecte de preuves avant de présenter des outils permettant de réaliser cette collecte.

Enfin, nous proposerons divers outils libres accessibles par tous et leur usage dans chaque étape et chaque aspect d'une analyse forensique (analyse de disque dur, analyse de mémoire vive), notamment avec un focus particulier en termes de retour d'expérience sur le produit Velociraptor que nous avons commencé à déployer au sein du parc machine. Ce dernier permet à distance d'avoir une vision très fine d'une machine (fichiers, base de registre, processus en cours, ...) et même de capturer la mémoire volatile à distance.

Nous conclurons sur l'intérêt de la mise en place de ces outils au sein de nos établissements dans une démarche d'amélioration continue sur la visibilité des "Endpoints" dans notre parc.

Mots-clefs

DFIR, EDR, HIDS, volatility, velociraptor, forensic, réponse, incident, endpoint

1 Introduction

Ces dernières années le rythme des incidents de sécurité s'est accéléré de manière expansive. Mais, depuis la pandémie du COVID-19 en 2020 ponctuée par les différentes périodes de confinement en mode télé-travail, les incidents de sécurité se sont multipliés dans tous les secteurs d'activité. Nous voyons que nos communautés ESR et EN sont également de plus en plus prises pour cibles. La liste des RSSI a vu considérablement augmenter les sujets liés à la recherche de marqueurs de compromission ou IOC[1]. Nos différentes instances représentées par le CERT-RENATER, le COSSIM ou l'ANSSI nous communiquent à intervalle régulier des éléments techniques à rechercher dans nos organismes.

Bien souvent, la question récurrente sur ces listes est : *Mais comment faire pour rechercher ces marqueurs au sein de nos parcs de machines ?*

Ainsi, nous allons présenter dans cet article une introduction à la notion de DFIR[2] (*Digital Forensic and Incident Response*). Voir comment les deux notions sont étroitement liées. Nous essayerons de voir différentes méthodologies possibles dans la démarche de collecte de preuves. Puis, nous ferons un focus tout particulier sur l'outil *Velociraptor* et sur ses différentes possibilités. Nous enchaînerons sur une présentation de divers outils libres et accessibles par tous dans différents aspects d'une analyse post-mortem ou encore forensique. Enfin, nous tenterons en perspective de voir comment améliorer au sein de notre communauté l'échange d'information techniques.

2 DFIR : Digital Forensic and Incident Response

Nous allons nous intéresser tout particulièrement à l'article de G.Arcas dans MISC [1] qui permet de poser les bases.

En effet, cette ressource présente succinctement différents guides et méthodes par rapport à l'organisation et la conduite d'une réponse à incidents. Des organismes tels que le NIST[3] à travers son *Computer Security Incident Handling Guide* [2], l'ENISA[4] et son *Good Practice Guide for Incident Management* [3] ou bien encore l'ISO/IEC via la *norme ISO 27035 et ses parties 1 et 2* [4], [5] ont décrit le management des incidents de sécurité et la conduite d'une réponse aux incidents.

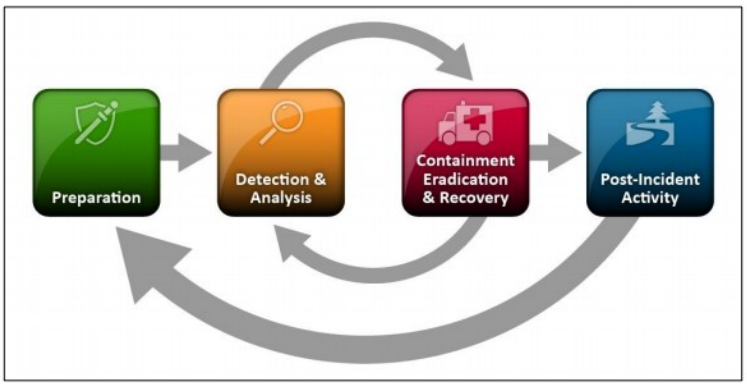

Figure 1: Cycle de vie d'une réponse à incident (© NIST SP 800-61 R2)

1 IOC : Indicators Of Compromise

2 DFIR : Digital Forensic and Incident Response

3 NIST : National Institute of Standards and Technology

4 ENISA : European Union Agency for Cybersecurity

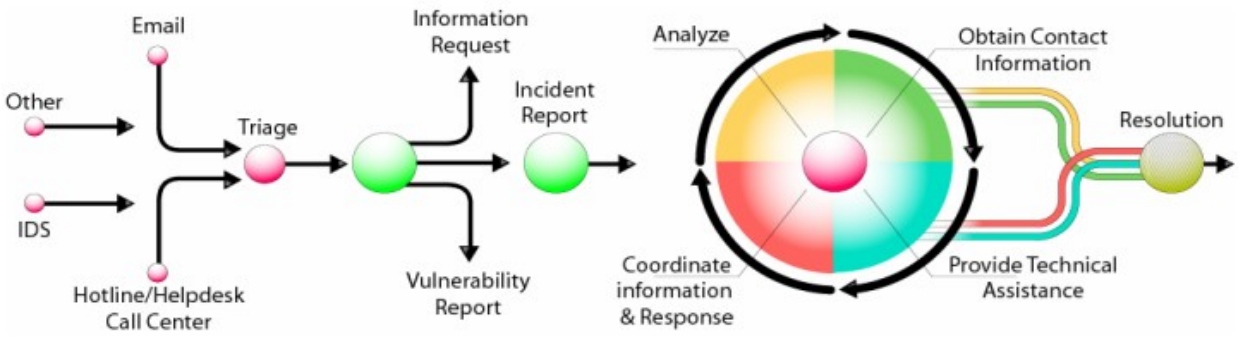

Figure 2: Cycle de vie de la gestion des incidents (© CERT/CC)

Les figures 1 et 2 montrent bien le cycle de vie d'une réponse/gestion d'incident à travers un *workflow* relativement similaire. Une source de donnée technique et/ou humaine pourra déclencher une qualification vers un rapport d'incident.

La figure 11 et la figure 12 représentées en annexe permettent de mettre en évidence que la réponse à incident et l'analyse d'un incident peuvent se faire en parallèle. En effet, pendant une phase d'incident, une chose importante à mettre en place est la collecte de preuves sous différentes formes (capture de la mémoire volatile, copie du disque dur, récupération des logs, capture de traces réseaux, …).

Pour résumer, la phase de *Réponse à Incident* se passe souvent dans l'urgence pour circonscrire l'incident le plus rapidement possible et se préserver d'une contamination à tout le réseau de l'organisme (par exemple dans le cas d'un *ransomware*). La phase *Investigation Numérique* est menée pour comprendre les causes de l'incident. Elle peut se faire en parallèle et les résultats trouvés peuvent ainsi fournir des éléments techniques (adresses IP, condensat de fichiers ou de base de registre,…) qui n'avaient pas encore été détectés. Ces éléments pourront être communiqués en termes de règles de protection à différents outils de sécurité tels que : Firewall, Antivirus, sonde IDS/IPS[5], SIEM[6], ...

Ainsi, le document *Gestion des incidents de sécurité du système d'information* [6] du CLUSIF décrit une méthodologie possible sur la gestion des incidents de sécurité sur les différentes phases que sont la qualification, la réponse immédiate, les investigations, la communication et le traitement.

5 IDS/IPS : Intrusion and Detection System / Intrusion and Prevention System

6 SIEM : Security Information and Event Management

3 « Velociraptor »

3.1 Introduction à la genèse de l'outil

Avant de détailler l'outil *Velociraptor*, il est important de présenter son auteur Michael Cohen (ancien ingénieur *Google)*. Il a développé de multiples logiciels libres orientés *forensic*. Le plus important d'entre tous est le produit GRR[7] (*Google Rapid Response*). De multiples articles et présentations le décrivent fort bien ([7], [8] et [9]).

Néanmoins, deux choses sont à noter : l'auteur a démissionné de chez *Google* et qui a pendant un temps abandonné le projet. Michael Cohen a naturellement, comme tout projet libre et open-source le permet, dupliqué et réécrit le code. Il a d'ailleurs créé son entreprise et une communauté s'est formée autour de ce produit.

3.2 Présentation de l'outil

Fonctionnalités principales :

- collecter de l'information ;
- monitorer des événements, une machine ;
- lancer des recherches à distance sur un périmètre précis.

Velociraptor est écrit en Go et s'installe très rapidement et simplement sur un serveur. Il fonctionne avec une architecture de type client/serveur avec l'installation d'un agent sur le système cible (Windows, Linux, Mac OSX). Il présente l'avantage de consommer peu de ressources (CPU, RAM) sur l'hôte distant.

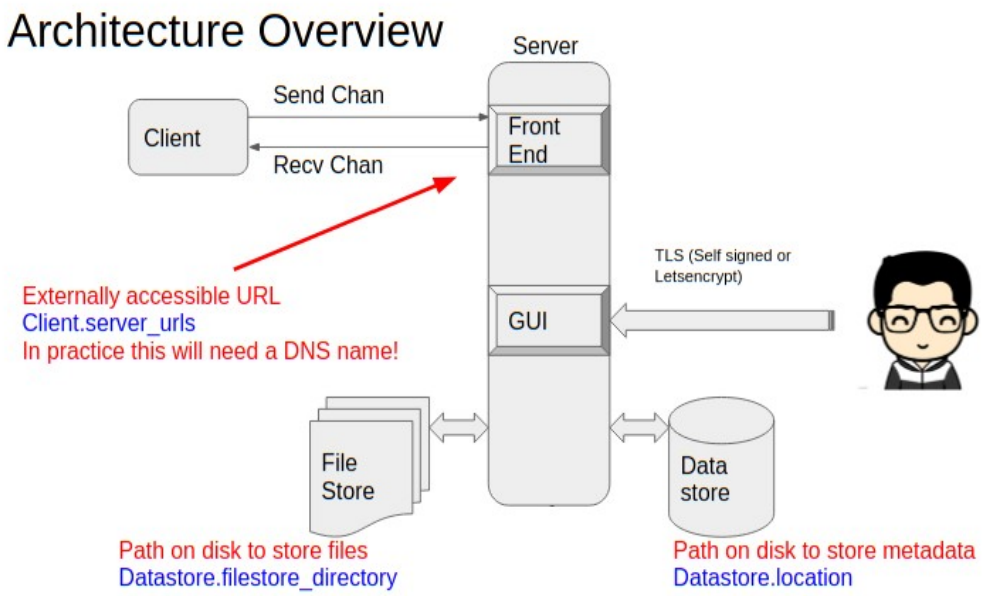

Figure 3: © Velociraptor : architecture client/serveur

7 GRR : Google Rapid Response

La figure 3 ci-dessus montre bien la relation client/serveur. Un utilisateur se connecte au serveur avec des droits de type RBAC[8] (*Administrator*, *Reader*, *API*, *Analyst*, *Investigator*, *Artefact_writer*) permettant pour une équipe dédiée de travailler avec les droits afférents.

3.2.1 Phase d'installation et de paramétrage

Cette phase d'installation et de paramétrage est très facile, il suffit de suivre les différentes questions :

```
./velociraptor config generate -i
```

Pour créer un binaire de type serveur ou client sous Linux :

```
./velociraptor --config server.config.yaml frontend -v
./velociraptor --config client.config.yaml client -v
```

Pour créer un binaire de type client sous Windows, il y a plusieurs étapes à suivre :

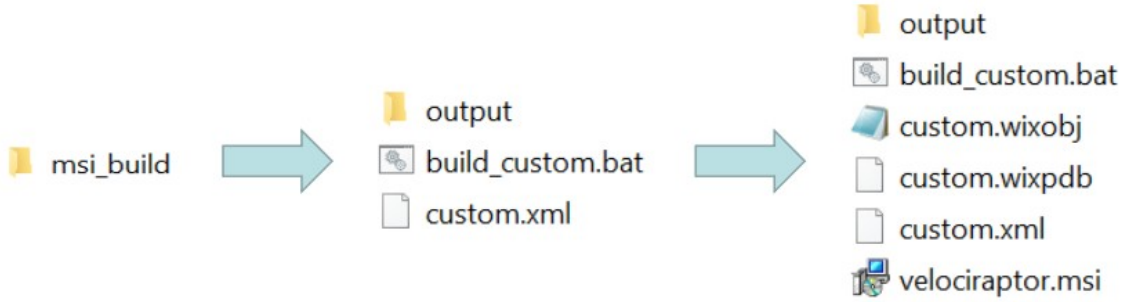

Figure 4: Velociraptor : Framework Wix pour optimiser le package MSI

Il faut utiliser le framework Wix, représenté via la figure 4, pour optimiser la création du package MSI. Cette optimisation se fait via le fichier *custom.xml* notamment en modifiant la version du produit, mais aussi dans notre cas en forçant une valeur fixe au champ *Id* afin de pouvoir faire le déploiement sans problème avec l'outil SCCM[9] de Microsoft. Il faut aussi lui indiquer le fichier *client.config.yaml*. Ainsi, le package MSI finalisé intégrera de facto tout le paramétrage adéquat pour une communication client/serveur nécessaire au bon déploiement au sein de notre organisme.

3.2.2 Interface web

L'interface web permet d'interagir et de manager complètement le serveur *Velociraptor*. Nous allons décrire plus en détail les parties intéressantes du menu contextuel comme indiqué à la figure 13 en annexe :

- **Home** est la page d'accueil de *Velociraptor* (et avec le *monitoring*) représentée à la figure 5 ;
- **Hunt Manager** est la page pour lancer des chasses représentées par des requêtes VQL[10] ;
- **View Artifacts** est la page pour visualiser et/ou modifier des *artefacts* ;
- **Notebooks** est la page pour créer rapidement des requêtes VQL et les tester immédiatement ;

8 RBAC : Role-based access control

9 SCCM : System Center Configuration Manager

10 VQL : Velociraptor Query Language

– **Virtual Filesystem** est la page permettant de montrer l'arborescence du système de fichiers de l'hôte distant ;

– **Collected Artifacts** est la page des résultats collectés suite à un *hunt ;*

– **Server Events** ou **Clients Events** sont les pages représentant les événements du serveur ou de l'hôte observé.

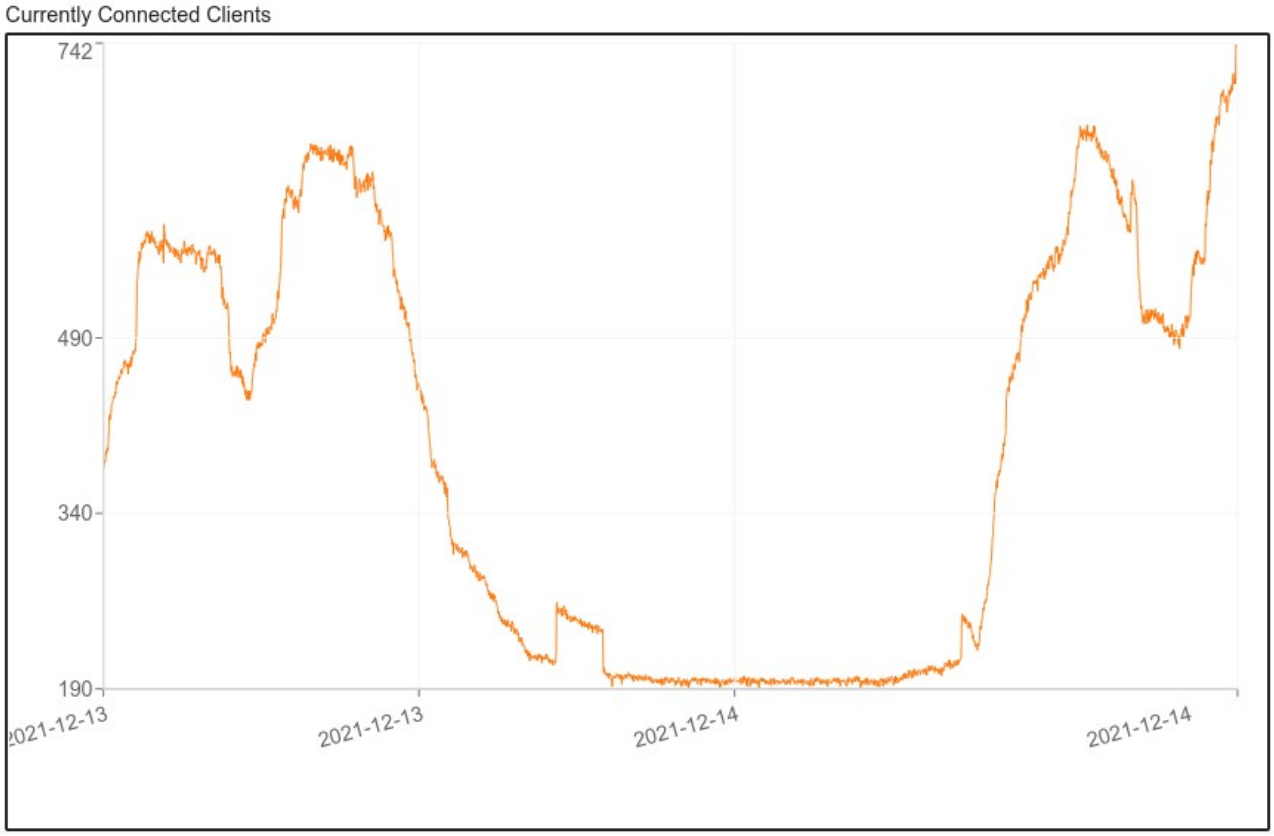

Figure 5: Velociraptor - monitoring des postes connectés

Il est intéressant de vouloir par exemple, faire des requêtes sur un ensemble de postes ayant des caractéristiques communes. Dans ce cas il est à noter l'intérêt des *labels* comme le montre, par exemple, la figure 14 en annexe.

3.2.3 Langage VQL (Velociraptor Query Language)

Le langage de requête dédié est ici le VQL, il permet entre autres :

– de collecter et analyser des informations depuis des clients ;

– de contrôler et générer des informations directement sur les clients ;

– de gérer le serveur.

L'exemple de requête VQL :

Figure 6: © Velociraptor syntaxe de base

La requête commence par un mot-clé **SELECT**, suivi d'une liste de sélecteurs de colonnes, puis du mot-clé **FROM** et d'un plug-in VQL prenant potentiellement des arguments. Enfin, nous avons un mot-clé **WHERE** suivi d'une expression de filtre.

3.2.4 Les artefacts

Un *artefact* est un fichier YAML composé de requêtes en VQL. Il répond à la requête :

- en collectant des données ;
- en consultant des données déjà collectées.

Il peut contenir alors :

- des variables ;
- des outils externes ;
- des conditions modifiables.

Les possibilités d'utilisation d'un *artefact* sont variées :

- génération des condensats de fichiers ;
- récupération de trames réseaux ;
- réalisation d'une capture mémoire d'une machine ;
- recherche d'un fichier/répertoire.

Nous avons créé plusieurs *artefacts* pour nos besoins notamment les deux mis avec les requêtes VQL en annexe. Ces deux *artefacts* permettent :

- *Linux.Volatility.Create.Profile* : de créer automatiquement le profil adéquat du poste client sélectionné en fonction de son kernel (uniquement Debian/Ubuntu pour l'instant) ;
- *Linux.Memory.Acquisition* : de capturer la mémoire à distance sur le poste client et avec la possibilité ou non d'utiliser l'*artefact* au-dessus pour créer le profile *Volatility* correspondant, puis de rapatrier la capture mémoire sur le serveur et en le supprimant du client.

Par ailleurs, nous avons créé un *artefact* dédié pour mettre des *labels* en fonction de la nomenclature des noms de machines.

Cette partie sur les *artefacts* est très importante et très intéressante. Elle change à chaque évolution du produit, par exemple avec les *artefacts* par défaut ainsi représentés à travers la figure 15, mais

aussi avec la contribution à la communauté de l'outil *Velociraptor* via le dépôt d'*artefacts*. Nous avons ainsi mis à disposition nos deux *artefacts* sur le dépôt centralisé *Artefact Exchange de Velociraptor* à [10] et [11].

3.2.5 Cas d'usage d'une recherche de hash

Nous allons prendre le cas d'usage qui revient régulièrement sur la liste des RSSI à savoir les marqueurs que peuvent fournir le CERT-RENATER et le COSSIM à nos intentions. Le but, bien évidemment, est de permettre une investigation en interne sur des marqueurs potentiels au sein de nos parcs.

Une possibilité technique avec l'outil *Velociraptor* est l'utilisation de trois *artefacts* qui successivement vont permettre de faire une chasse ou *hunt* sur notre parc.

Ainsi, en utilisant les *artefacts* suivants :

– *Generic.Forensic.LocalHashes.Init ;*

– *Generic.Forensic.LocalHashes.Glob ;*

– *Generic.Forensic.LocalHashes.Query.*

Nous pouvons consécutivement :

– créer une base de données de type SQLite sur un poste client comme représenté en résultat via la figure 7 ;

– remplir cette base en analysant tout le disque dur ou seulement un dossier très précisément, sur tous les fichiers ou seulement sur un type de fichiers précis comme exposé en annexe à la figure 16 ;

– rechercher un condensat d'un fichier de type MD5 ou SHA1 ou SHA256 à travers cette base de données distantes qui renverra les résultats sur le serveur si la recherche est fructueuse comme décrit par les figures 8 et 9.

FullPath	FlowId	ClientId	Fqdn
/tmp/hashdb.sqlite	F.C6NOB2HDKSERG	C.08fe07498aeae81d	clt-dfir-test

Figure 7: Velociraptor - Création d'une base de données de type SQLite

Custom.URCA.Generic.Forensic.LocalHashes.Query

Hashes	+	Hash
	+	XXX
CommaDelimitedHashes	fd4b0d1aef29a97ad08c6b322f5cb714, ba518c65b0addefa60a799a91a29fe8481071cf9, 22a28b270a58db7deb7fd17f38ec80719cccb0eaf8d3b333c1e9c541e40282e1	
HashDb	hashdb.sqlite	

Figure 8: Velociraptor - Recherche de condensat

Path	MD5	Size	Timestamp	FlowId	ClientId	Fqdn
/home/forensic/.bash_history	fd4b0d1aef29a97ad08c6b322f5cb714	637	2021-12-07T15:55:04Z	F.C6NOI48QM5R40	C.08fe07498aeae81d	clt-dfir-test
/home/forensic/Documents/article_jres_2021.odt		8220	2021-12-07T15:55:04Z	F.C6NOI48QM5R40	C.08fe07498aeae81d	clt-dfir-test
/home/forensic/Téléchargements/velociraptor_0.6.2_client.deb		15927192	2021-12-07T15:58:09Z	F.C6NOI48QM5R40	C.08fe07498aeae81d	clt-dfir-test

Figure 9: Velociraptor - Résultat d'un condensat MD5 correspondant à la recherche

Pour finir, nous avons modifié un *artefact* afin d'automatiser le remplissage de la base de données toutes les minutes sur les postes distants.

3.2.6 Bilan de Velociraptor

Cet outil est très intéressant dans le cadre de la réponse à incident et investigation numérique notamment depuis la version 0.6.1 sortie courant août 2021. Il permet surtout de pouvoir mettre en quarantaine un poste à distance à travers l'icône sur la figure 10, de personnaliser un message pour le poste client puis de le retirer de la quarantaine.

Figure 10: Velociraptor - mise en quarantaine d'un poste

Ainsi, seule la communication entre le serveur *Velociraptor* et le poste client reste établie, les autres communications sont interdites sur le poste client. Dans le cas d'une réponse à incident, nous pouvons donc en cas de suspicion ou d'un fait avéré sur un poste, le bloquer à distance pour éviter une éventuelle propagation au reste du réseau. Cela permet d'investiguer et/ou de collecter plus sereinement sur le poste avant une éventuelle réinstallation complète.

L'outil installé sur les postes clients est léger et le paramétrage par défaut d'un serveur permet d'avoir plus de 10.000 postes clients connectés dessus.

4 « Volatility »

4.1 Présentation de l'outil

Volatility est un framework écrit en Python et qui est la référence pour analyser des captures mémoires de systèmes d'exploitation différents (Windows, Linux, Mac OSX), et il est porté par la fondation *Volatility* [12]. Il est capable de supporter différents types de capture mémoire tels que :

- Raw linear sample (dd)
- Hibernation file (from Windows 7 and earlier)
- Crash dump file
- VirtualBox ELF64 core dump
- VMware saved state and snapshot files
- EWF format (E01)

– LiME format

– Mach-O file format

– QEMU virtual machine dumps

4.2 Exemple d'usage sous Linux

Après avoir fait un *Hunt* avec *Velociraptor* sur un poste client Linux, nous récupérons deux fichiers : le profil pour *Volatility* et la capture mémoire. Il est d'usage de copier le profil dans un des dossiers présents :

```
/usr/share/volatility/volatility/plugins/overlays/linux/
ou
/usr/lib/python2.7/dist-packages/volatility/plugins/overlays/linux
```

Nous vérifions ensuite la bonne prise en compte du profil par l'outil via :

```
volatility  --info | grep "Profile" | grep Linux
Volatility Foundation Volatility Framework 2.6
LinuxDebian-JRES2021x64                     - A Profile for Linux Debian-JRES2021 x64
```

Nous pouvons ensuite investiguer sur la mémoire au moment de la capture :

– sur les processus en cours :

```
volatility -f dump.raw --profile=LinuxDebian-JRES2021x64 linux_pslist
```

– sur les connexions réseaux existantes :

```
volatility -f dump.raw --profile=LinuxDebian-JRES2021x64 linux_netstat
```

– sur l'historique du terminal :

```
volatility -f dump.raw --profile=LinuxDebian-JRES2021x64 linux_bash
```

– sur les fichiers ouverts :

```
volatility -f dump.raw --profile=LinuxDebian-JRES2021x64 linux_enumerate_files > files.log
```

– ou encore de récupérer un fichier ouvert en se basant sur les *Offsets* :

```
cat files.log |grep Document
0xffff8af678fce2e8                    1054941 /home/forensic/Documents
0xffff8af678b07828                    1049329 /home/forensic/Documents/article_jres_2021.odt

volatility -f dump.raw --profile=LinuxDebian-JRES2021x64 linux_find_file -i 0xffff8af678b07828 -O
article_jres_2021.odt
```

Une liste exhaustive des plugins disponibles pour *Volatility* est visible dans les ouvrages [13] ou [14]. D'autres exemples de ce qu'il est possible de faire avec *Volatility* sont notamment décrits respectivement dans [15] pour Windows, [16] pour Linux, [17] pour MacOSX et [18] pour Android.

Cela reste un outil individuel mais dans une démarche de mutualisation au sein d'une équipe dédiée à l'analyse forensique, un outil tel que *Orochi* [19] pourrait être intéressant à mettre en place avec une base commune pour mutualiser les recherches, les découvertes, les annotations,... . C'est une interface graphique basée sur des technologies web tout en se reposant sur l'outil *Volatility*.

5 « Autopsy »

5.1 Présentation de l'outil

Un très bon article de W. Bonnaventure [21] décrit l'analyse des disques durs en faisant, entre autres, une comparaison de différents logiciels. Ainsi, *Autopsy* fait partie de cette liste-là. Ce genre d'outil permet :

- d'analyser la table de partition et les différentes partitions d'un disque dur ;
- de naviguer à travers l'arborescence des dossiers et chemins ;
- de visualiser un historique de navigateur web ;
- de voir les fichiers présents (vidéos, images, textes, …).

Concernant les fichiers, il y en a deux types : les fichiers présents dans un système de fichiers et les fichiers effacés qui ont été récupérés par le logiciel. Le *carving* consiste à reconstruire des fichiers en analysant les secteurs d'un disque en l'absence de métadonnées provenant du système de fichiers. Cette opération permet de récupérer des documents partiels ou complets qui ont été effacés par le système ou l'utilisateur.

La présentation de Brian Carrier à l'OSDFCon[11] de 2019 [22] permet d'introduire rapidement *Autopsy* sur ses différentes possibilités. Il permet de voir l'activité récente d'une machine.

Il est extrêmement puissant à plusieurs titres :

- les analystes forensique peuvent collaborer sur la même affaire en même temps ;
- une base de données centrale indexe les textes et stocke les différents *artefacts* rencontrés ;
- les évènements sont mis sous forme de frise chronologique ;
- une corrélation avec des *artefacts* déjà rencontrés sur des affaires antérieures.

La liste est encore longue sur toutes les possibilités de cet outil. Régulièrement de nouveaux modules permettent de faire évoluer le produit sur des besoins précis (analyse iOS, analyse Android, analyse de drone, détection de fichiers chiffrés, …). Des exemples plus concrets sont disponibles dans l'ouvrage de G. Johansen [23].

6 Conclusion

Cet article décrit brièvement notre démarche et l'utilisation de différents outils open-source dans une stratégie de réponse à incident et d'analyse forensique. La mise en place d'un outil tel que *Velociraptor* au sein de nos établissements a son sens afin de pouvoir collecter rapidement de l'information à distance avant une remise rapide du poste de travail en configuration initiale par les services d'informatique de proximité. Par ailleurs, cela permet de commencer à collecter plus d'événements, d'informations en cas d'incidents et sans les compétences en interne nécessaires sur la phase d'analyse forensique. Il pourra être judicieux de communiquer ces données au CERT-

11 Open Source Digital Forensics Conference

RENATER et au COSSIM dans une démarche d'amélioration de la réponse à incident avec le partage d'informations techniques.

Nous pouvons néanmoins mettre un bémol à ce genre de produit. Il nécessite d'être installé sur tout le parc avec un agent. Nous savons pertinemment que plus de la moitié des machines dans nos parcs ne sont pas gérées par l'établissement car ils ne nous appartiennent pas : ordinateurs portables, tablettes numériques, smartphones des étudiants et/ou du personnel. Ainsi, la moitié seulement des terminaux seraient analysés. Ce qui peut être un frein à sa mise en place et/ou à son intérêt.

Cependant, nous voyons aussi grâce à ce genre de produit, une possibilité de mutualiser les indicateurs techniques au sein de la communauté EN/ESR. Par exemple, il pourrait être opportun que le COSSIM ou une autre entité porte un projet de partage de ce type d'informations afin de pouvoir mutualiser et partager les IOC plus facilement, nous pensons notamment à un produit tel que MISP [24].

Bibliographie

[1] G.Arcas, DFIR : Hier, aujourd'hui et demain, MISC Hors-Série N°23, 2021.

[2] Computer Security Incident Handling Guide, 2012, NIST.
https://csrc.nist.gov/publications/detail/sp/800-61/rev-2/final

[3] Good Practice Guide for Incident Management, 2010, ENISA.
https://www.enisa.europa.eu/publications/good-practice-guide-for-incident-management/at_download/fullReport

[4] ISO/IEC 27035-1, Information technology - Security techniques - Information security incident management – Part 1 Principles of incident management, ISO/IEC, 2016.

[5] ISO/IEC 27035-2, Information technology - Security techniques - Information security incident management – Part 2 Guidelines to plan and prepare for incident response, ISO/IEC, 2016.

[6] Clusif, Gestion des incidents de sécurité du système d'information, 2011.
https://clusif.fr/wp-content/uploads/2015/09/clusif-2011-gestion-des-incidents.pdf

[7] M.Cohen & al, Distributed forensics and incident response in the enterprise, 11th Annual Digital Forensics Research Conference, 2011.
https://www.sciencedirect.com/science/article/pii/S1742287611000363

[8] A.Moser & al, Hunting in the enterprise: Forensic triage and incident response, Digital Investigation: The International Journal of Digital Forensics & Incident Response, 2013.
https://dl.acm.org/doi/10.1016/j.diin.2013.03.003

[9] E.Ladent, Arrêtez de grogner avec GRR Rapid Response, MISC N°87, 2016.

[10] C.Garcia & al, Linux Volatility Create Profile, 2021.
https://docs.velociraptor.app/exchange/artifacts/pages/volatility_profile

[11] C.Garcia & al, Linux Memory Acquisition, 2021.
https://docs.velociraptor.app/exchange/artifacts/pages/linuxmemoryacquisition

[12] Volatility Foundation, 2021.
https://www.volatilityfoundation.org/

[13] M.Ligh & al, The Art of Memory Forensics: Detecting Malware and Threats in Windows, Linux, and Mac Memory, Wiley, 2014.

[14] Plugins disponibles pour Volatility, 2021.
https://github.com/volatilityfoundation/volatility

[15] A.Menelet & al, Retour d'expérience : investigation numérique dans un environnement Windows, MISC N°111, Septembre 2020.

[16] F.Baguelin, Volatilisons Linux : partie 2, MISC N°76, Novembre 2014.

[17] C.Maurugeon, Investigation numérique sous macOS / HFS+, MISC N°107, Janvier 2020.

[18] J.Sylve & al, Acquisition and analysis of volatile memory from android devices, Digital Investigation, Février 2012, Pages 175-184.
https://www.sciencedirect.com/science/article/abs/pii/S1742287611000879

[19] Outil Orochi, CERT LDO.
https://github.com/LDO-CERT/orochi

[20] Autopsy, 2021.
https://www.sleuthkit.org/autopsy/

[21] W.Bonnaventure, L'analyse de disques durs, MISC HS N°23, Février 2021.

[22] B.Carrier, Introduction to Autopsy, 2019.
https://www.osdfcon.org/presentations/2019/Brian-Carrier_Introduction-to-Autopsy.pdf

[23] G.Johansen, Digital Forensics and Incident Response, Packt Publishing, 2017.

[24] MISP, Open Source Threat Intelligence Platform, 2021.
https://www.misp-project.org/

Annexe

Exemple 1 : ISO 27035 : Phases de gestion des incidents de sécurité de l'information (ISO/IEC)

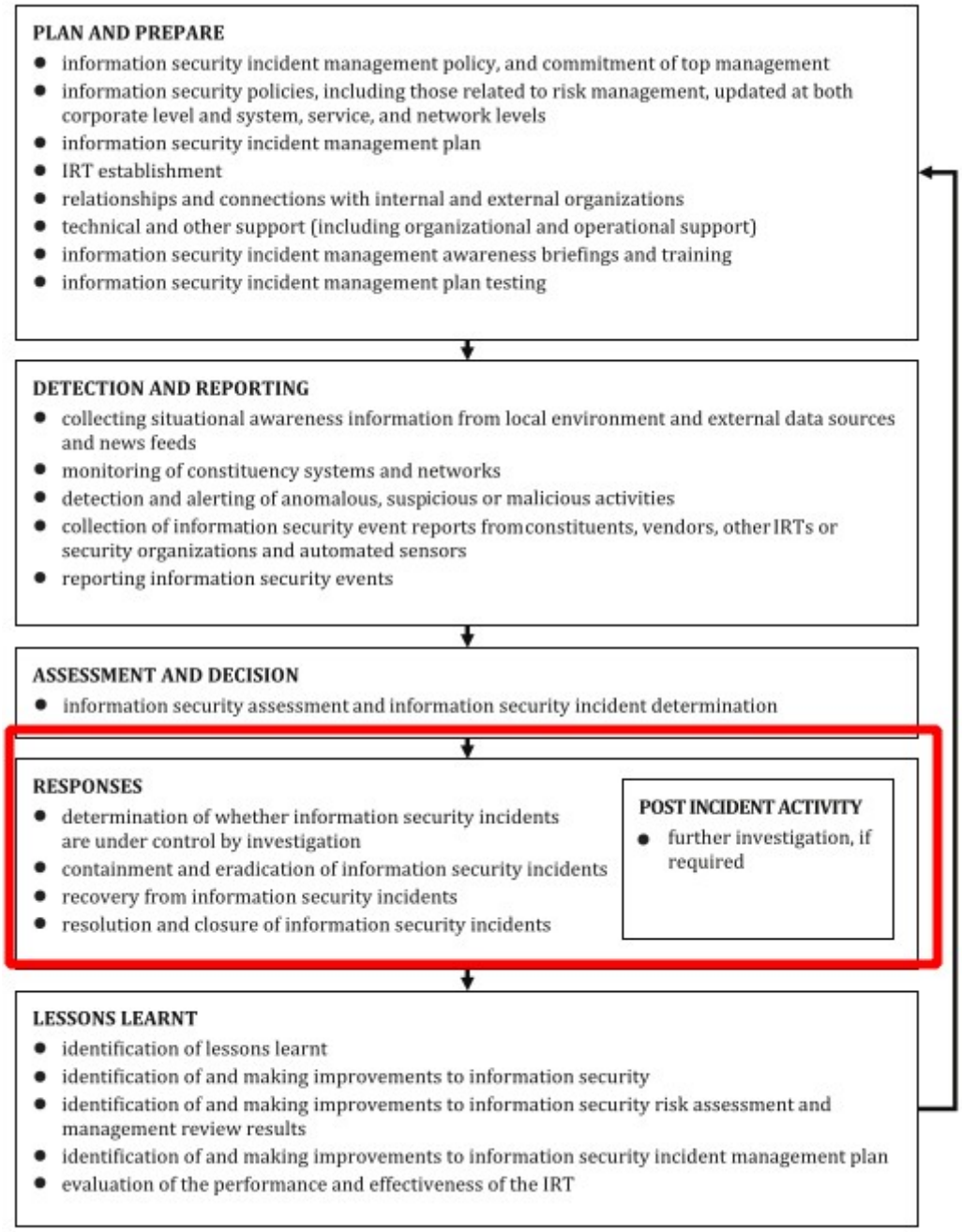

Figure 11: ISO 27035 : Phases de gestion des incidents de sécurité de l'information (© ISO/IEC)

Exemple 2 : RI et IN liaisons vertueuses (© MISC HS N°23, p35)

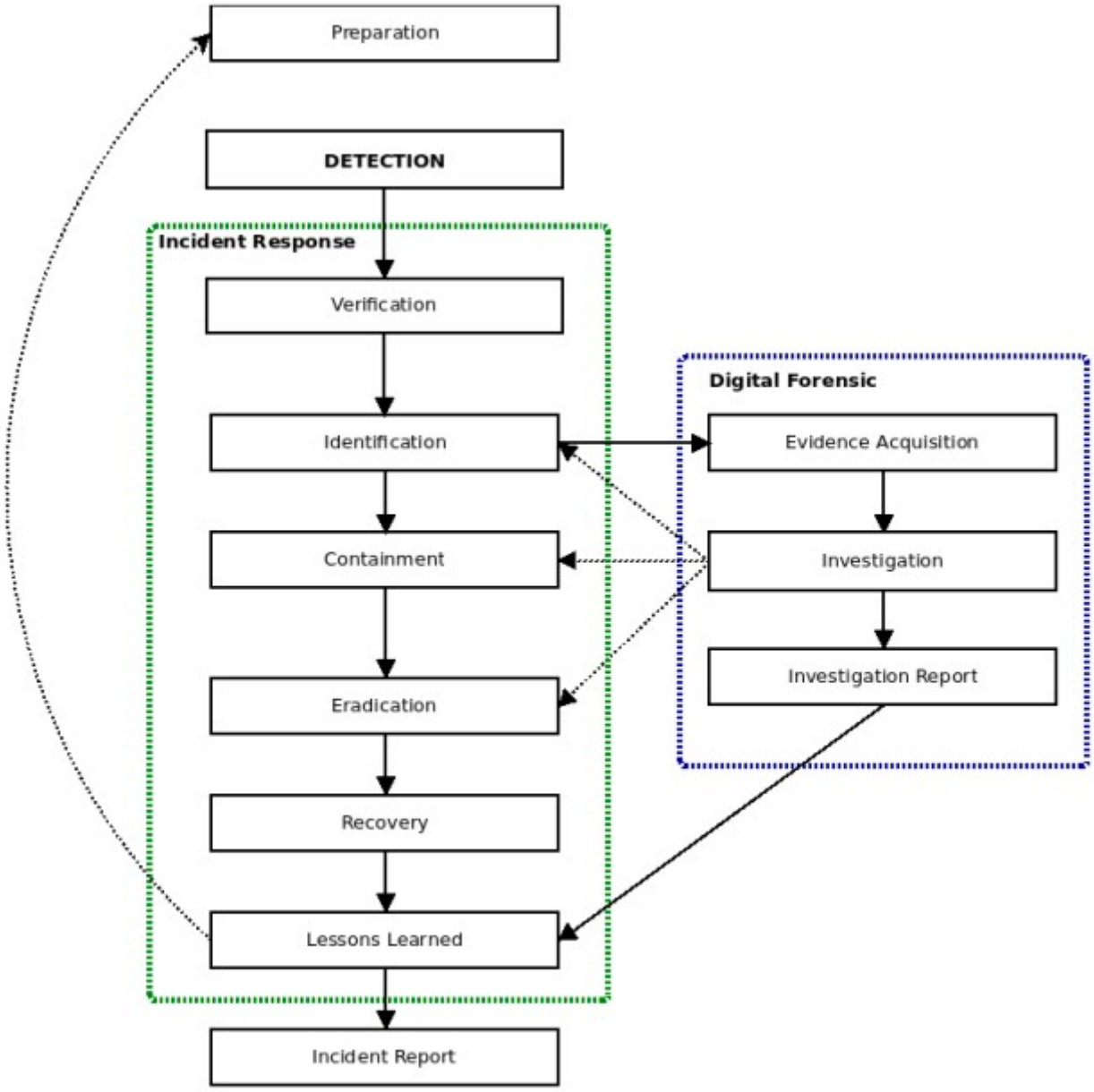

Figure 12: Réponse à Incident & Investigation Numérique - relations vertueuses

Exemple 3 : *Velociraptor* – menu contextuel

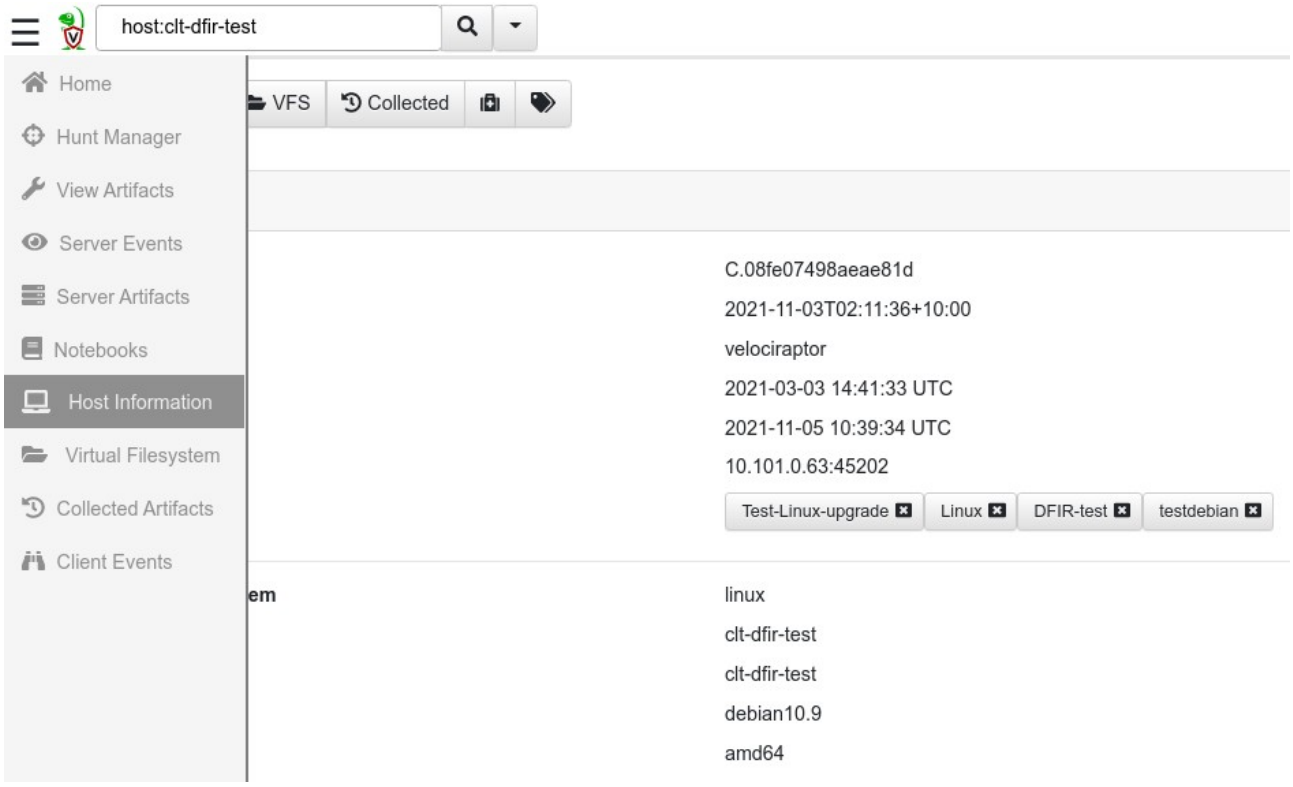

Figure 13: Velociraptor - menu contextuel

Exemple 4 : *Velociraptor* – Intérêt des *labels*

Figure 14: Velociraptor - Clients avec les labels

Exemple 5 : *Velociraptor* – Création d'un *artefact* pour créer un profil Volatility pour clients Linux

```
name: Custom.URCA.Linux.Volatility.Create.Profile
author: URCA (Corentin Garcia / Emmanuel Mesnard)
description: |
  This artifact is used to create the profile to the environnements Debian / Ubuntu.



required_permissions:
  - EXECVE

tools:
  - name: Volatility
    url: https://github.com/volatilityfoundation/volatility/archive/master.zip

parameters:
    - name: Zipname
      type: string
      default: Ubuntu

precondition: SELECT OS From info() where OS = 'linux'

sources:
  - queries:
    - LET dirtmp = tempdir(remove_last=true)


        LET vola = SELECT * FROM execve(argv=['bash', '-c', 'mv /tmp/master.zip /tmp/volatility-
master.zip ; cd /tmp/ ; apt install -y dwarfdump zip unzip ; unzip -o /tmp/volatility-master.zip -d
/tmp/  ; cd /tmp/volatility-master/tools/linux/ ; make clean ; make ; zip /tmp/' + Zipname +
'.zip /tmp/volatility-master/tools/linux/module.dwarf /boot/System.map-$(uname -r)'])


      SELECT * FROM foreach(
          row={
            SELECT FullPath FROM Artifact.Generic.Utils.FetchBinary(ToolName="Volatility")
          },
          query={
            SELECT * FROM chain(
                  a={SELECT *, if(condition=Complete, then=upload(file="/tmp/" + Zipname + ".zip",
name=Zipname + ".zip")) As Upload FROM vola},
                        b={SELECT * FROM execve(argv=['bash', '-c', 'mv /tmp/volatility-master
/tmp/volatility-master.zip /tmp/' + Zipname + '.zip ' + dirtmp])})
          })
```

Exemple 6 : *Velociraptor* – Création d'un *artefact* pour capturer la mémoire pour clients Linux

```
name: Custom.URCA.Linux.Memory.Acquisition
author: URCA (Corentin Garcia / Emmanuel Mesnard)
description: |
  Acquires a full memory image. We download LiME and use it to acquire a full memory image.
  NOTE: This artifact usually transfers a lot of data. You should increase the default timeout to allow it to complete. ( Example : 2Gb of memory time takes about 50s )


required_permissions:
- EXECVE

tools:
  - name: LiME
    url: https://github.com/504ensicsLabs/LiME/archive/refs/heads/master.zip
  - name: Volatility
    url: https://github.com/volatilityfoundation/volatility/archive/master.zip
parameters:
    - name: StartVolatility
      type: bool
      default: Y
    - name: Zipname
      type: string
      description: Name of zip profile for Volatility
      default: Ubuntu
    - name: Dumpname
      type: string
      description: Name of dump memory
      default: dump

precondition: SELECT OS From info() where OS = 'linux'
sources:
  - queries:
      - LET Volatility = SELECT * FROM Artifact.Custom.URCA.Linux.Volatility.Create.Profile(Zipname=Zipname)
      SELECT * FROM foreach(
          row={
            SELECT FullPath FROM Artifact.Generic.Utils.FetchBinary(ToolName="LiME")
          },
          query={
            SELECT * FROM chain(
            a={SELECT FullPath, Stdout, Stderr,
                   if(condition=Complete,
                         then=upload(file="/tmp/" + Dumpname + ".raw", name=Dumpname + ".raw")) As Upload FROM execve(argv=['bash', '-c', 'mv /tmp/master.zip /tmp/lime-master.zip ; unzip -o /tmp/lime-master.zip -d /tmp/ ; cd /tmp/LiME-master/src/ ; apt-get -y install build-essential linux-headers-$(uname -r) ; make ; insmod /tmp/LiME-master/src/lime-*.ko "path=/tmp/"' + Dumpname + '".raw format=lime"'])},
            b={SELECT * FROM execve(argv=['bash', '-c', 'rm -rf /tmp/' + Dumpname + '.raw'])},
            c={SELECT * FROM if(
                condition=StartVolatility,
                then=Volatility,
                else=scope()
            )}
            )
          })
```

Exemple 7 : *Velociraptor* – exemple par défaut d'*artefacts*

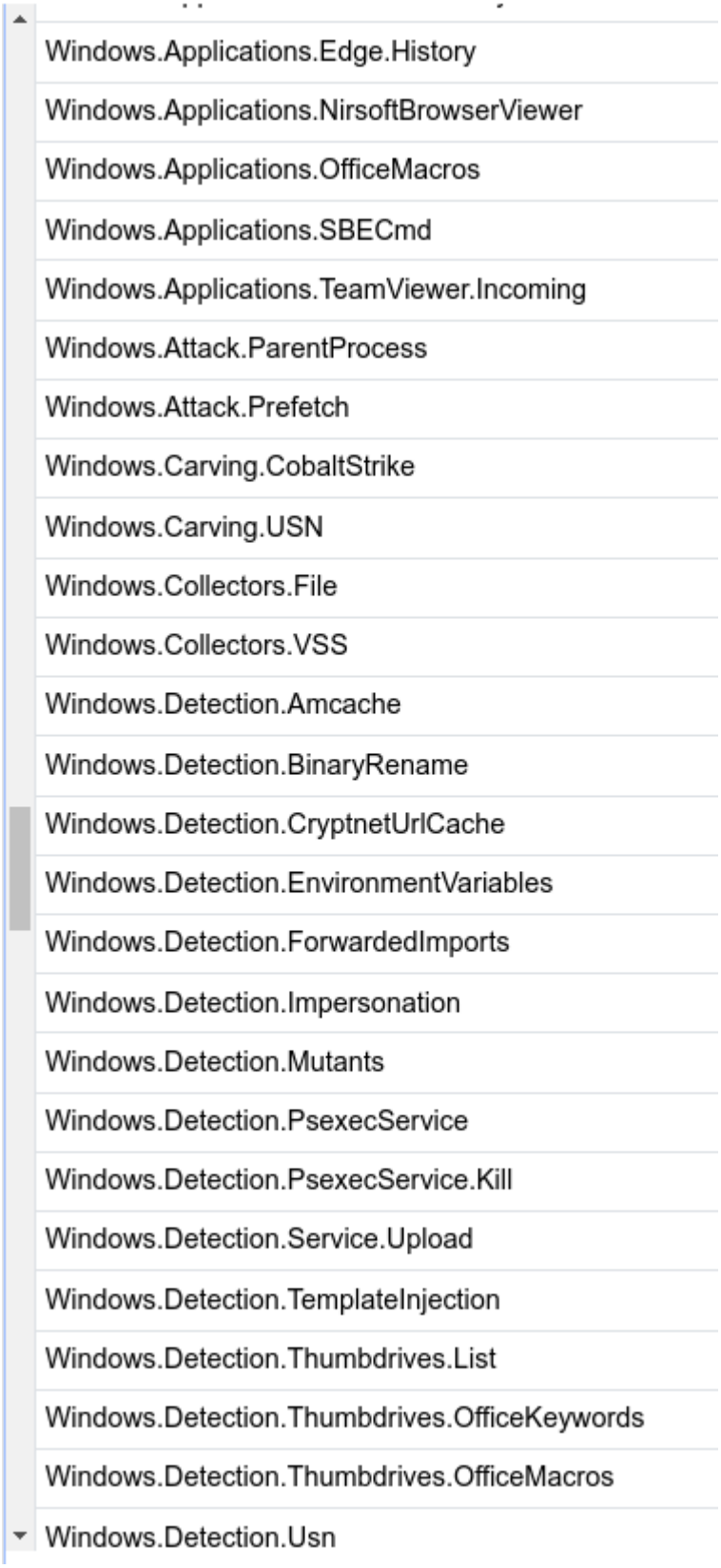

Figure 15: Velociraptor - exemple par défaut d'artefacts

Example 8 : *Velociraptor* – Analyse des fichiers d'un dossier pour récupérer les condensats

	path	md5	sha1	sha256	size	timestamp
	Filtre	Filtre	Filtre	Filtre	Filtre	Filtre
1	/tmp/.X0-lock	4595c2ffa6865ebfd1780e4088d1b7...	ccead61f4dd9539c6b1320...	75d2429069292e254347...	11	1638892504
2	/tmp/.xfsm-ICE-7LN1D1	1149c47eaf793a4e1d18246fe5086...	73ccbaff1d4271ca7b1a72...	6b21ba06c566d1afaca11...	430	1638892504
3	/tmp/hashdb.sqlite	600e18a55ece8da83a70cb1a04c50...	8d8ddd48b0df08b23019b...	4e4453326ba30b3455d8...	28672	1638892504
4	/home/forensic/.ICEauthority	843cfae0e6159acd77659a47e321a...	4801c5463ef0b623204 1a4...	bbd17c493197f911eaa3f...	2736	1638892504
5	/home/forensic/.Xauthority	03dbf4bd7374e1f442c2e271d9841...	64bf316ce1fe80d10dd467...	381ba8e088e775418b3f...	58	1638892504
6	/home/forensic/.bash_history	fd4b0d1aef29a97ad08c6b322f5cb7...	19d64c9a02bc8322b3f297...	386ded8b06b544318b8f...	637	1638892504
7	/home/forensic/.bash_logout	22bfb8c1dd94b5f3813a2b25da674...	dc216ac4a4c2328157319...	26882b79471c25f945c97...	220	1638892504
8	/home/forensic/.bashrc	2130bff0f5992e9f095da13337d5e4a4e	abe05a77de4f6be0ddaba3...	968095122e5e8c115a7e...	3525	1638892504
9	/home/forensic/.dmrc	dbad5c4177782e0cdf057d0210bb6...	d0970b24183951202a37e...	aaf3853303e76b221e2c4...	35	1638892504
10	/home/forensic/.profile	f4e81ade7d6f9fb342541152d08e7a...	2b9ee6d446f8f9ffccaab42...	28b4a453b68dde64f814...	807	1638892504
11	/home/forensic/.recently-used	37610450d7de56777a173d969def1...	c93f095a7c5377e53f83ef0...	f1dbad1258d95dbd7c8e...	339	1638892504
12	/home/forensic/.vboxclient-clipboard.pid	89f6a0bff21f3bb3f1fa73bf8391f25a	8bc9bfbec0532c9c2d9b42...	74ca9c82081ee39c47f73...	4	1638892504
13	/home/forensic/.vboxclient-display.pid	b249765015629c2463d17c05c65e...	488c1ac191788594ad658...	d2f9933100f3388210476...	4	1638892504
14	/home/forensic/.vboxclient-draganddrop.pid	e2bb17e005a998642a09bc27a568...	7d87f1b03df38ae328858b...	180cc1607cdf0fc42b95df...	4	1638892504
15	/home/forensic/.vboxclient-seamless.pid	d9210b5882984388aa89c6d4807a...	eb0de4c9ceecbca816f618...	17176d48a9e2493da3d4...	4	1638892504
16	/home/forensic/.xfce4-session.verbose-log	5cb90b3e9434419280 15d52772cca...	79d9258260 5c872c41a53...	94fab783bc4635404c917...	17315	1638892504
17	/home/forensic/.xfce4-session.verbose-log.last	b5a9e6cf4b8626268 9379eead71fb...	929e8ef2cf3d9621861bd2...	705dc24df4081ea73495...	19896	1638892504
18	/home/forensic/.xsession-errors	69727a8a6297092b3c99928b9d09...	1c72fda3ceae2954aa11c4...	dfb82571e93368fd9e72b...	2745	1638892504
19	/home/forensic/.xsession-errors.old	5a1081d4115 2d47eb011678b8839...	b925777fc0da84b53c7f3a...	bc1e9b74af43ef8805567...	2909	1638892504
20	/home/forensic/Documents/.~lock.article_jres_2021.odt#	f9e608e692dc5b8e17f8fed7349405...	7a0357e38adcd97ce1e69...	098cdb9520ff4dba092b1...	85	1638892504
21	/home/forensic/Documents/article_jres_2021.odt	fabe7e9400b1e72d3e4a5f47c047a...	ba518c65b0addefa60a799...	cc6980b285bc892dff697...	8220	1638892504
22	/home/forensic/Téléchargements/VBoxLinuxAdditions.run	f7dea3958d87b53ac420cfef7cfeb3f0	c8e65c40e316c5a4a93afd...	06f481c6d9627bb6a115...	6307277	1638892505
23	/home/forensic/Téléchargements/linpmem-2.1.post4	4a5a922b0c0c2b38131fb4831cc4e...	d099504cd8eaa948285fcc...	622b01d1739e8befa960...	7272696	1638892505
24	/home/forensic/Téléchargements/velociraptor-v0.5.4-linux-amd64	cd524f8cbc317ad8b3269f983be36...	c1ac2bcbc19820afb21f9d...	7658eae7acab869cbef97...	42210368	1638892506

Figure 16: Velociraptor - Condensats des fichiers de dossiers en base SQLite