Détection de menaces : Fichiers, processus, journaux, réseau

Les méthodes de détection (pré)historiques

Détection des fichiers (exécutables, scripts...) :

- Les antivirus disposent d'une **base de signatures** de fichiers connus comme étant malveillants
- Ils calculent l'empreinte des fichiers analysés
- Vérifient la présence de l'**empreinte** dans la base
- Si l'empreinte est connue > alerte + blocage + suppression

Détection des flux réseau :

- Basée sur des **adresses IP** identifiées comme malveillantes
- Basée sur des **noms de domaines** ou **FQDN** identifiés
- comme malveillants

Ces méthodes doivent-elles être abandonnées aujourd'hui ?

NON !

Elles sont :

- Limitées
- Souvent fiables !

Les méthodes de détection (pré)historiques

Les limites :

- Une modification infime dans un fichier = plus de détection
- Un fichier non encore référencé par l'éditeur = pas de détection
- Les serveurs C2 utilisés pas les attaquants (ou leurs adresses IP) peuvent changer
- Les noms de domaine servent souvent pour une campagne dont la durée est limitée
- Des outils parfaitement légitimes peuvent être utilisés par les attaquants
- Des informations très intéressantes se trouvent dans les journaux, mais ne sont pas regardées

Qui es-tu ? :

- Projet open source
- Initié en 2008 par Victor Manuel Alvarez (Virus Total)
- Vraiment connu depuis 2014
- Objectif : identifier des fichiers malveillants
- À l'aide de règles de détection
- Dont le format est devenu un standard
- Outil intégré dans de nombreuses solutions de détection et d'investigation
- Redéveloppé en Rust (développé en C à l'origine) il devient Yara-X en mai 2024

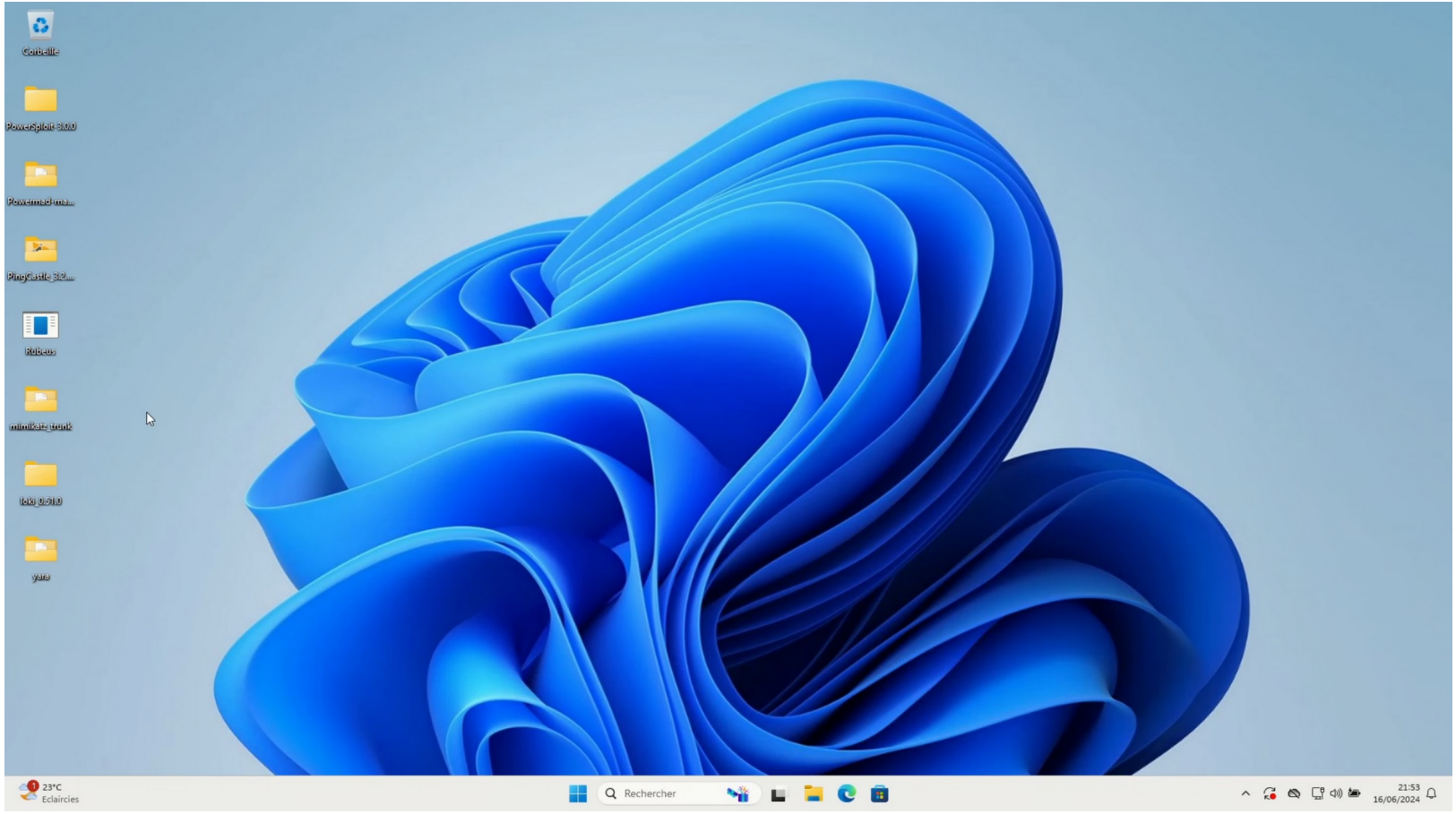
Exemple de règle de détection d'un fichier :


```
rule mimikatz : FILE {
   meta:
      description      = "mimikatz"
      author         = "Benjamin DELPY (gentilkiwi)"
      tool_author      = "Benjamin DELPY (gentilkiwi)"
      modified = "2022-11-16"
      id = "840a5b8c-a311-50bc-a099-6b8ab1492e12"
   strings:
      $exe_x86_1      = { 89 71 04 89 [0-3] 30 8d 04 bd }
      $exe_x86_2      = { 8b 4d e? 8b 45 f4 89 75 e? 89 01 85 ff 74 }

      $exe_x64_1      = { 33 ff 4? 89 37 4? 8b f3 45 85 c? 74}
      $exe_x64_2      = { 4c 8b df 49 [0-3] c1 e3 04 48 [0-3] 8b cb 4c 03 [0-3] d8 }

/*
      $dll_1         = { c7 0? 00 00 01 00 [4-14] c7 0? 01 00 00 00 }
      $dll_2         = { c7 0? 10 02 00 00 ?? 89 4? }
*/

      $sys_x86      = { a0 00 00 00 24 02 00 00 40 00 00 00 [0-4] b8 00 00 00 6c 02 00 00 40 00 00 00 }
      $sys_x64      = { 88 01 00 00 3c 04 00 00 40 00 00 00 [0-4] e8 02 00 00 f8 02 00 00 40 00 00 00 }

   condition:
      (all of ($exe_x86_*)) or (all of ($exe_x64_*))
      // or (all of ($dll_*))
      or (any of ($sys_*))
}
```

Détection d'un fichier :


```
[ALERT]
FILE: C:\Users\c.blanc-rolin\Desktop\mimikatz_trunk\x64\mimikatz.exe SCORE: 475 TYPE: EXE SIZE: 1355264
FIRST_BYTES: 4d5a900003000000040000ffff0000b8000000 / <filter object at 0x000001CAC64C3DF0>
MD5: 29efd64dd3c7fe1e2b022b7ad73a1ba5
SHA1: e3b6ea8c46fa831cec6f235a5cf48b38a4ae8d69
SHA256: 61c0810a23580cf492a6ba4f7654566108331e7a4134c968c2d6a05261b2d8a1 CREATED: Mon Sep 19 16:44:40 2022 MODIFIED: Mon Sep 19 16:44:40 2022 ACCESSED: Sun Jun 16 21:54:17 2024
REASON_1: File Name IOC matched PATTERN: \\(q32|q64|wceaux|w86|q86|quarkpwd[^\\]*|m64|m32|hash32|hash64|64|32|wce32|wce64|w32|w64|wce|p32|p64|ps32|ps64|mimikatz|mimilove|mm32|mm64|pw32|pw64|g32|g64|gs32|gs64|hashdump|dumpsvc)\.exe SUBSCORE: 60 DESC: Cred Dumping
REASON_2: Yara Rule MATCH: mimikatz SUBSCORE: 70
DESCRIPTION: mimikatz REF: - AUTHOR: Benjamin DELPY (gentilkiwi)
MATCHES: $exe_x64_1: '3\xffA\x897L\x8b\xf3E\x85\xc0t', $exe_x64_2: 'L\x8b\xdfI\xc1\xe3H\x8b\xcbL\xd8'
```

yara : Détection d'un fichier



Exemple de règle de détection d'un processus :

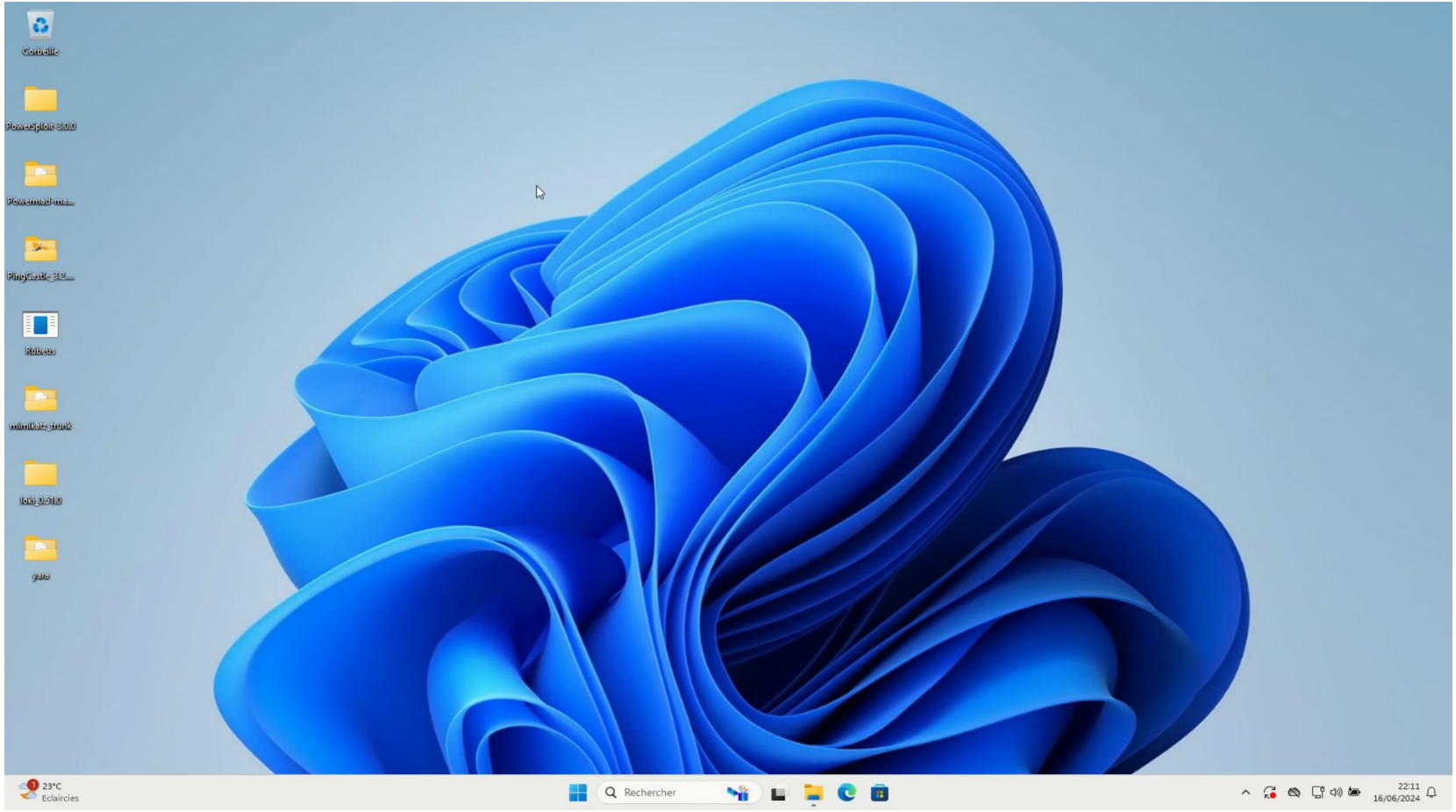
```
rule HKTL_Mimikatz_SkeletonKey_in_memory_Aug20_1_RID3752 : DEMO HKTL S0002 T1003 T1098_004 T1134_005 T1547_008 T1550_002 T1550_003 {
   meta:
      description = "Detects Mimikatz SkeletonKey in Memory"
      author = "Florian Roth"
      reference = "https://twitter.com/sbousseaden/status/1292143504131600384?s=12"
      date = "2020-08-09 17:33:31"
      score = 75
      customer = "demo"
      license = "CC-BY-NC https://creativecommons.org/licenses/by-nc/4.0/"

      tags = "DEMO, HKTL, S0002, T1003, T1098_004, T1134_005, T1547_008, T1550_002, T1550_003"
      minimum_yara = "1.7"

   strings:
      $x1 = { 60 ba 4f ca c7 44 24 34 dc 46 6c 7a c7 44 24 38
              03 3c 17 81 c7 44 24 3c 94 c0 3d f6 }
   condition:
      1 of them
}
```

Détection d'un processus :

Sélection Administrateur : Windows PowerShell

```
Windows PowerShell
Copyright (C) Microsoft Corporation. Tous droits réservés.

Installez la dernière version de PowerShell pour de nouvelles fonctionnalités et améliorations ! https://aka.ms/PSWindow
s

PS C:\Windows\system32> cd C:\Users\c.blanc-rolin\Desktop\yara\
PS C:\Users\c.blanc-rolin\Desktop\yara> .\yapscan.exe scan -C .\yara_rules.yar 8604
Filters: segment state must be one of ["commit"] AND segment must be smaller than 240 MB


Scanning process "mimikatz.exe" (8604) by user "BUILTIN\Administrateurs"...
 MATCH: Rule "HKTL_Mimikatz_SkeletonKey_in_memory_Aug20_1_RID3752" matches segment 0x00007FF752011000.
 MATCH: Rule "Powerkatz_DLL_Generic_RID2F2F" matches segment 0x00007FF7520E1000.
Scanning 8604: 100 %
Some processes matched the provided rules, see above.
PS C:\Users\c.blanc-rolin\Desktop\yara>
```

yara : Détection d'un processus



Qui es-tu ? :

- Projet open source
- Lancé en 2017 par Florian Roth (Nextron Systems)
- Objectif : détecter des actions malveillantes / suspectes à partir des journaux d'évènements
- À l'aide de règles de détection
- Dont le format est devenu un standard
- Outil intégré dans de nombreuses solutions de détection et d'investigation
- Importante communauté rédactrice de règles (initialement espérée par son créateur)

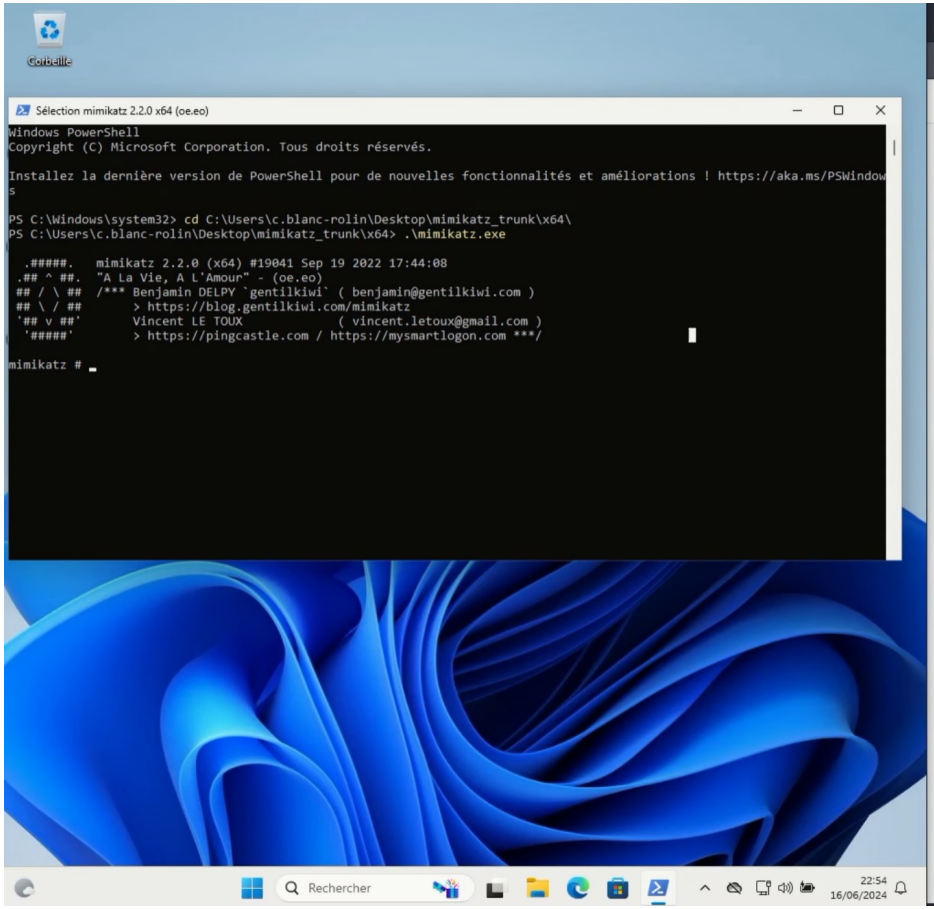
Exemple de règle de détection d'une action malveillante (DCSync depuis un compte utilisateur) :

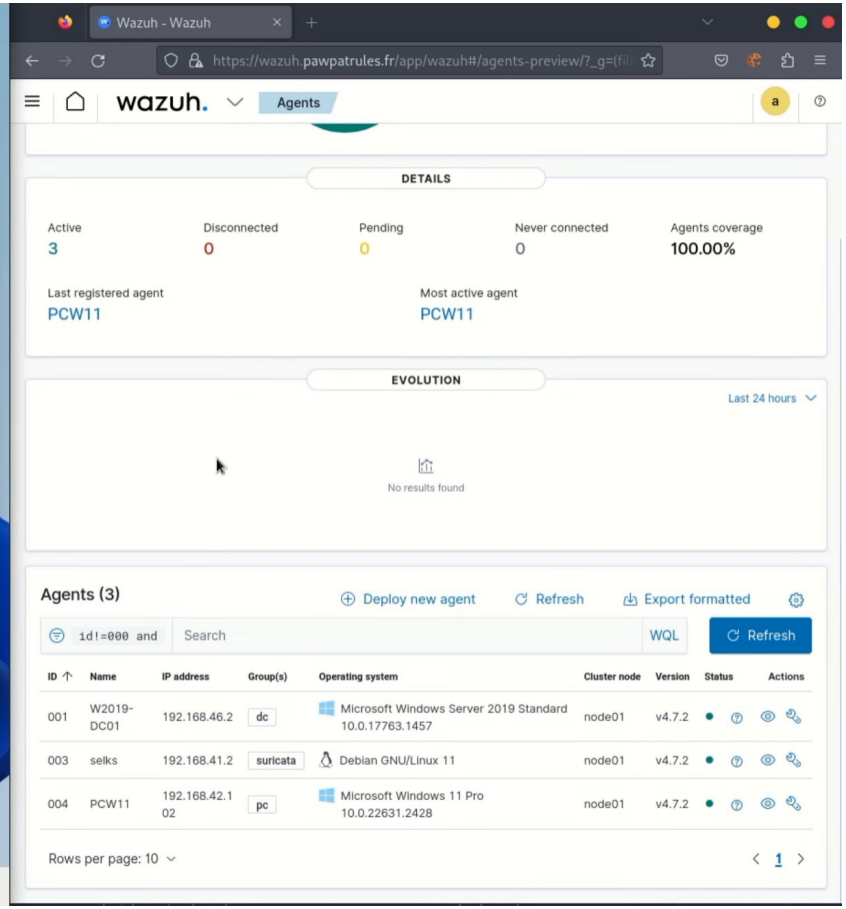
```yaml
title: Active Directory Replication from Non Machine Account
id: 17d619c1-e020-4347-957e-1d1207455c93
status: test
description: Detects potential abuse of Active Directory Replication Service (ADRS) from a non machine account to request credentials.
author: Roberto Rodriguez @Cyb3rWard0g
references:
  - https://threathunterplaybook.com/notebooks/windows/06_credential_access/WIN-180815210510.html
date: 2019/07/26
modified: 2021/11/27
logsource:
  product: windows
  service: security
detection:
  selection:
    EventID: 4662
    AccessMask: '0x100'
    Properties|contains:
      - '1131f6aa-9c07-11d1-f79f-00c04fc2dcd2'
      - '1131f6ad-9c07-11d1-f79f-00c04fc2dcd2'
      - '89e95b76-444d-4c62-991a-0facbeda640c'
  filter:
    - SubjectUserName|endswith: '$'
    - SubjectUserName|startswith: 'MSOL_'     #https://docs.microsoft.com/en-us/azure/active-directory/hybrid/reference-connect-accounts-
  condition: selection and not filter
fields:
  - ComputerName
  - SubjectDomainName
  - SubjectUserName
falsepositives:
  - Unknown
level: critical
tags:
  - attack.credential_access
  - attack.t1003.006
```

```
- <Event xmlns="http://schemas.microsoft.com/win/2004/08/events/event">
  - <System>
      <Provider Name="Microsoft-Windows-Security-Auditing" Guid="{54849625-5478-4994-a5ba-3e3b0328c30d}" />
      <EventID>4662</EventID>
      <Version>0</Version>
      <Level>0</Level>
      <Task>14080</Task>
      <Opcode>0</Opcode>
      <Keywords>0x8020000000000000</Keywords>
      <TimeCreated SystemTime="2024-06-16T20:55:28.482460600Z" />
      <EventRecordID>20336</EventRecordID>
      <Correlation />
      <Execution ProcessID="596" ThreadID="712" />
      <Channel>Security</Channel>
      <Computer>W2019-DC01.woundride.local</Computer>
      <Security />
    </System>
  - <EventData>
      <Data Name="SubjectUserSid">S-1-5-21-3332904308-1614487934-3407257785-1104</Data>
      <Data Name="SubjectUserName">c.blanc-rolin</Data>
      <Data Name="SubjectDomainName">WOUNDRIDE</Data>
      <Data Name="SubjectLogonId">0x9d5294</Data>
      <Data Name="ObjectServer">DS</Data>
      <Data Name="ObjectType">%{19195a5b-6da0-11d0-afd3-00c04fd930c9}</Data>
      <Data Name="ObjectName">%{3e64635d-222e-43fd-8f57-9b817764f67a}</Data>
      <Data Name="OperationType">Object Access</Data>
      <Data Name="HandleId">0x0</Data>
      <Data Name="AccessList">%%7688</Data>
      <Data Name="AccessMask">0x100</Data>
      <Data Name="Properties">%%7688 {1131f6ad-9c07-11d1-f79f-00c04fc2dcd2} {19195a5b-6da0-11d0-afd3-00c04fd930c9}</Data>
      <Data Name="AdditionalInfo">-</Data>
      <Data Name="AdditionalInfo2" />
    </EventData>
  </Event>
```

Sigma : Détection d'une action malveillante

SIEM Detection Format





SURICATA

Qui es-tu ? :

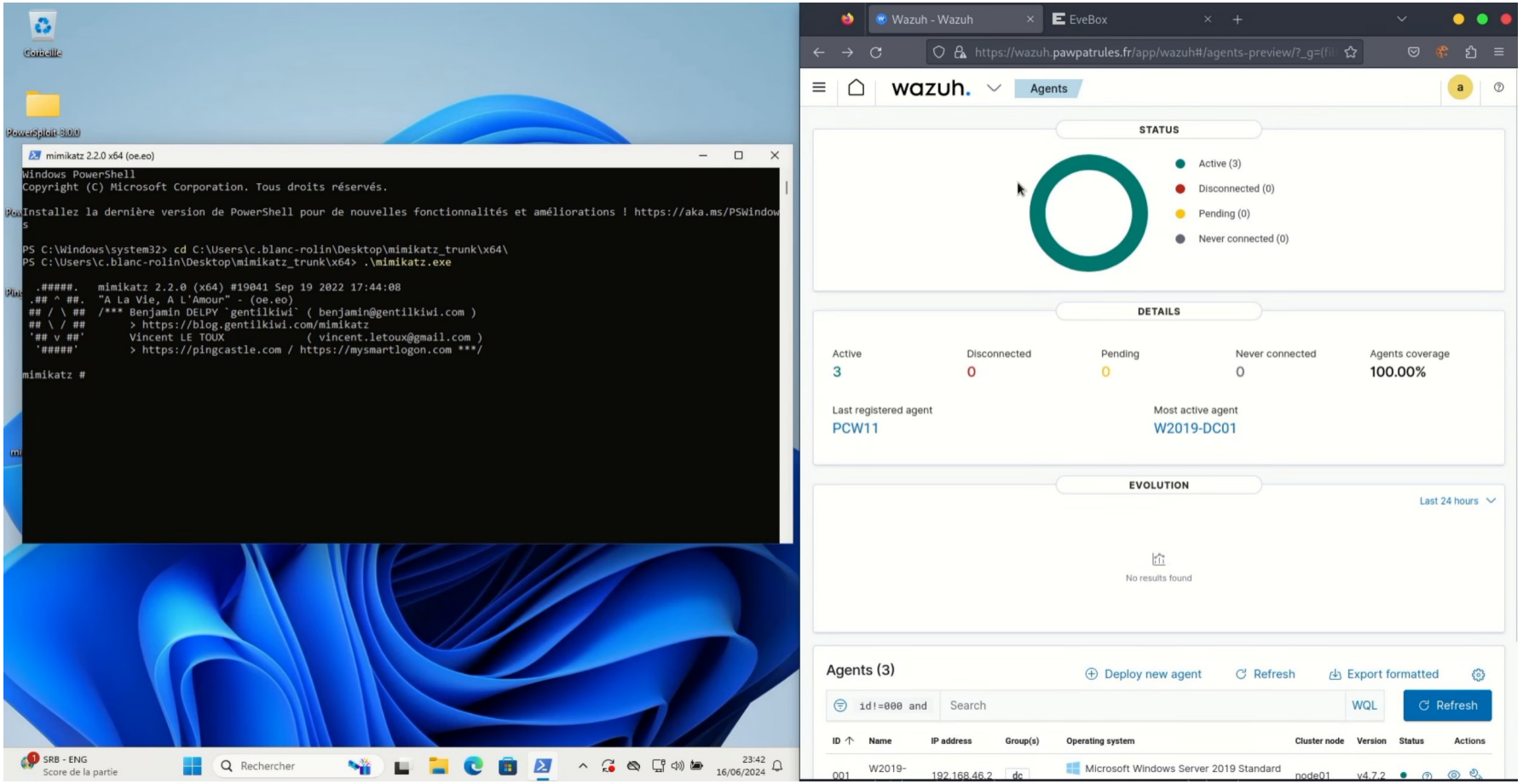
- Projet open source encadré par la fondation OISF (Open Information Security Foundation)
- Réponse à l'arrêt de l'acceptation des contributions de la communauté au sein du projet Snort
- Initié par Victor Julien, William Metcalf (développeurs et anciens contributeurs du projet Snort) et Matt Jonkman (fondateur d'Emerging Threat)
- Version 1 lancée en 2010
- Objectif : détecter des menaces à partir du trafic réseau
- À l'aide de règles de détection
- Dont le format est devenu un standard
- Outil intégré dans de nombreuses solutions de détection, dont les sondes qualifiées par l'ANSSI

Exemple de règle de détection d'une action malveillante (DCSync à l'aide Mimikatz) :

```
alert tcp-pkt any any -> $HOME_NET any (msg:"🐾 - 🔔 DRSUAPI DsGetDomainControllerInfo - Possible Mimikatz
DCSync attack 🥷 - T1003.006 - Check if source is a legit 🪟 Domain Controler"; flow:to_server, stateless;
content:"|05 00 00|"; depth:3; content:"|03 00 00 00 50 00 00 00 00 00 10 00|"; fast_pattern;
reference:url,https://attack.mitre.org/techniques/T1003/006/; reference:url,https://github.com/gentilkiwi/
mimikatz; reference:url,https://adsecurity.org/?p=1729; metadata:created_at 2024_05_30, updated_at 2024_05_30,
signature_severity Major, attack_target Server_Endpoint, affected_product Windows_Server_32_64_Bit,
mitre_tactic_id TA0006, mitre_tactic_name Credential_Access, mitre_technique_id T1003_006, mitre_technique_name
OS_Credential_Dumping_DCSync; sid:3321275; rev:1; classtype:attempted-recon;)
```

PAWPATRULES.FR

SURICATA : Détection d'une action malveillante



Les « plus » du langage Suricata :

- Possibilité de spécifier le protocole réseau utilisé (exemple : tls)
- Ainsi que les mots clés associés (exemple : tls.cert_subject)
- Possibilité d'utiliser des expressions régulières pour détecter du contenu
- Exemple d'une règle de détection d'un certificat présenté par un C2 Qbot / Pikabot :

```
alert tls $EXTERNAL_NET any -> $HOME_NET any (msg:"🐾 - 🚨 Suspicious TLS Certificate - Possible 😈 QakBot / Qbot / Pikabot 🪟 flow with C2 Server"; flow:to_client,
stateless; tls.cert_subject; content:"C="; pcre:"/C=[A-Z]{2}/"; content:"OU="; pcre:"/OU=[a-zA-Z\t.]{3,35}/"; content:"CN="; pcre:"/CN=[a-z]{1,35}.[a-z]{2,5}/";
tls.cert_issuer; content:"C="; pcre:"/C=[A-Z]{2}/"; content:"ST="; fast_pattern; pcre:"/ST=[A-Z]{2}/"; content:"L="; pcre:"/L=[a-zA-Z\t.]{3,35}/"; content:"O="; pcre:"/
O=[a-zA-Z\t.]{3,35}/"; content:"CN="; pcre:"/CN=[a-z]{1,35}.[a-z]{2,5}/"; reference: url,https://malpedia.caad.fkie.fraunhofer.de/details/win.qakbot; reference:
url,https://malpedia.caad.fkie.fraunhofer.de/details/win.pikabot; metadata:attack_target Client_and_Server, signature_severity Major, affected_product
Windows_XP_Vista_7_8_10_11_Server_32_64_Bit, mitre_tactic_id TA0011, mitre_tactic_name Command_and_Control, mitre_technique_id T1071.001, mitre_technique_name
Application_Layer_Protocol_Web_Protocols, former_category MALWARE, malware_family QakBot,created_at 2024_01_11, updated_at 2024_03_07; sid:3301116; rev:8;
classtype:command-and-control;)
```

PAWPATRULES.FR

Nouveauté dans la V8 (en développement, version beta 1 en cours) :

- Sortie officielle de la version 8 prévue pour juillet 2025
- Possibilité d'intégrer une volumétrie de données en entrée ou en sortie (exemple : upload de 100Mo vers une IP publique via le protocole SSH)

```
alert ssh any any -> $EXTERNAL_NET any (msg:"🐾 - 🚨 Over 100MB uploaded via SSH / SFTP to public IP address - Possible data
exfiltration 🏴‍☠️"; requires: version >= 8; flow:to_server, established; threshold: type both, track by_src,count 1, seconds
60; flow.bytes_toserver:>=100000000; metadata:created_at 2024_02_18, updated_at 2024_06_04; sid:3301137; rev:4;
classtype:policy-violation;)
```

PAWPATRULES.FR

Ressources et outils utilisés

- Yara :
  https://virustotal.github.io/yara/
- Sigma :
  https://sigmahq.io/
- Suricata :
  https://suricata.io/
- Jeux de règles pour Suricata :
  https://rules.evebox.org/
- Yapscan (FKIE-CAD) :
  https://github.com/fkie-cad/yapscan
- Loki (Florian Roth) :
  https://github.com/Neo23x0/Loki
- Wazuh :
  https://wazuh.com/
- Selks / Clear NDR (Stamus Networks) :
  https://www.stamus-networks.com/clear-ndr-community
- EveBox (Jason Ish) :
  https://evebox.org/
- PawPatrules (Charles BLANC-ROLIN) :
  https://pawpatrules.fr/