APT28, sarA Is watching you!

Retex sur l'adoption de l'IA pour du reverse de malware

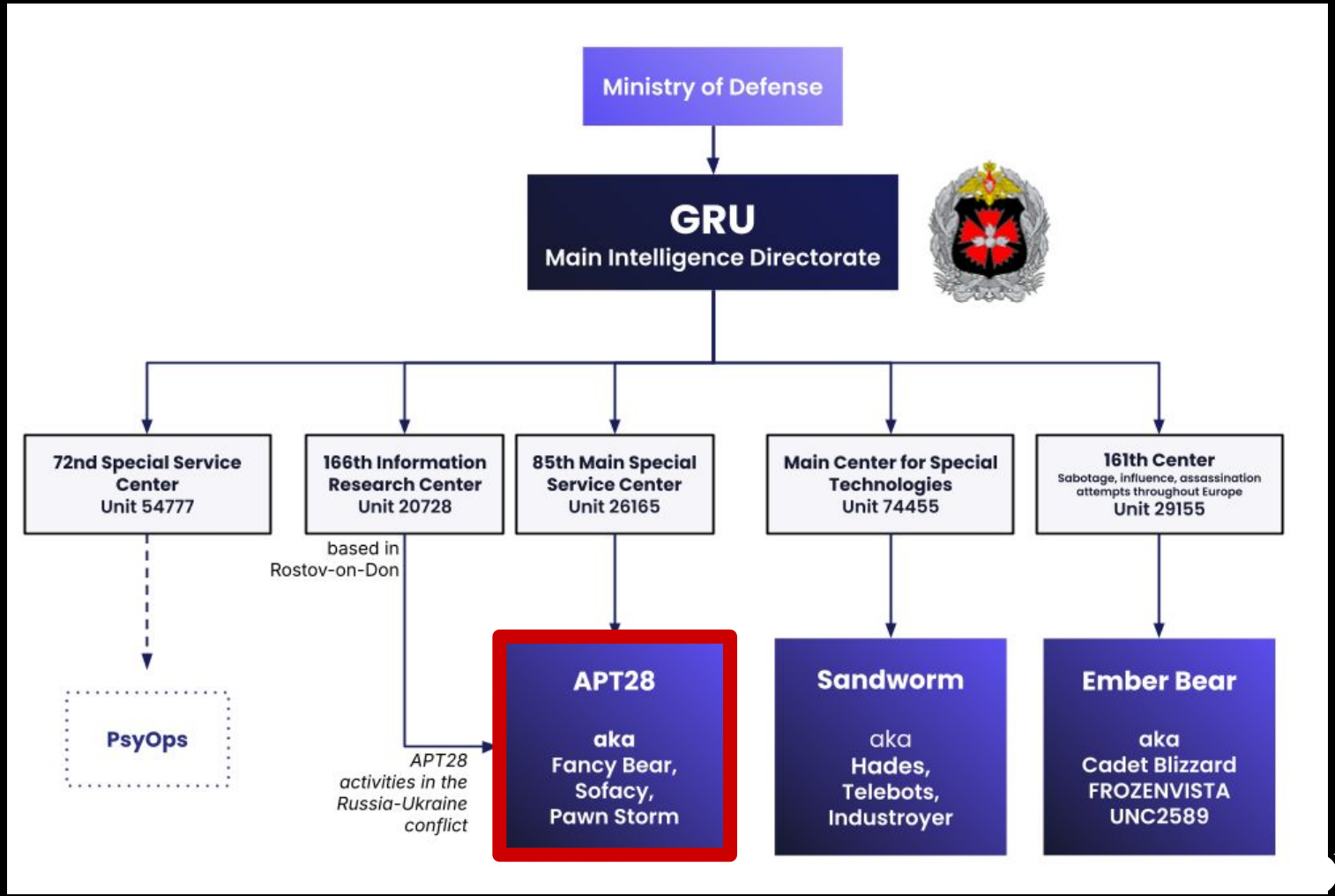
Robin Kwiatkowski
Charles Meslay

TDR by sekoia
THREAT DETECTION & RESEARCH

APT28

(Fancy Bear, Sofacy, Sednit, etc.)

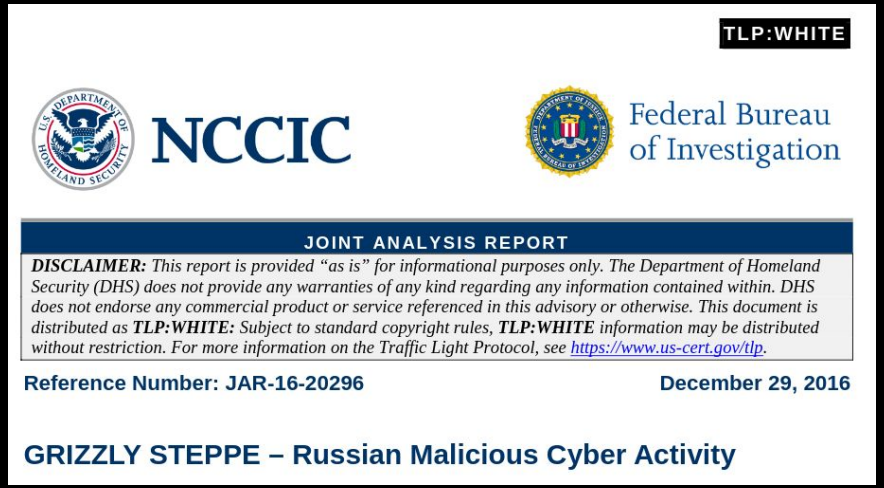
- **Actif depuis 2004**
- **Attribué au GRU**
  - renseignement militaire russe


Ministry of Defense
GRU
Main Intelligence Directorate
72nd Special Service Center
Unit 54777
166th Information Research Center
Unit 20728
based in Rostov-on-Don
85th Main Special Service Center
Unit 26165
Main Center for Special Technologies
Unit 74455
161th Center
Sabotage, influence, assassination attempts throughout Europe
Unit 29155
PsyOps
APT28 activities in the Russia-Ukraine conflict
APT28
aka
Fancy Bear,
Sofacy,
Pawn Storm
Sandworm
aka
Hades,
Telebots,
Industroyer
Ember Bear
aka
Cadet Blizzard
FROZENVISTA
UNC2589

APT28

(Fancy Bear, Sofacy, Sednit, etc.)

- **Actif depuis 2004**
- **Attribué au GRU**
  - renseignement militaire russe
- **Campagnes de "*hack & leak*"**
  - Objectif : Discréditer les systèmes démocratiques



TLP:WHITE

NCCIC

Federal Bureau of Investigation

JOINT ANALYSIS REPORT

DISCLAIMER: *This report is provided "as is" for informational purposes only. The Department of Homeland Security (DHS) does not provide any warranties of any kind regarding any information contained within. DHS does not endorse any commercial product or service referenced in this advisory or otherwise. This document is distributed as* ***TLP:WHITE:*** *Subject to standard copyright rules,* ***TLP:WHITE*** *information may be distributed without restriction. For more information on the Traffic Light Protocol, see https://www.us-cert.gov/tlp.*

Reference Number: JAR-16-20296 **December 29, 2016**

GRIZZLY STEPPE – Russian Malicious Cyber Activity

In spring 2016, APT28 compromised the same political party, again via targeted spearphishing. This time, the spearphishing email tricked recipients into changing their passwords through a fake webmail domain hosted on APT28 operational infrastructure. Using the harvested credentials, APT28 was able to gain access and steal content, likely leading to the exfiltration of information from multiple senior party members. The U.S. Government assesses that information was leaked to the press and publicly disclosed.

5

Russie – Attribution de cyberattaques contre la France au service de renseignement militaire russe (APT28)

Russie | Diplomatie numérique

Déclaration

Le : 29 avril 2025

La France condamne avec la plus grande fermeté le recours par le service de renseignement militaire russe (GRU) au mode opératoire d'attaque APT28, à l'origine de plusieurs cyber-attaques contre des intérêts français.

Depuis 2021, ce mode opératoire d'attaque (MOA) a été utilisé dans le ciblage ou la compromission d'une dizaine d'entités françaises. Ces entités sont des acteurs de la vie des Français : services publics, entreprises privées, ainsi qu'une organisation sportive liée à l'organisation des Jeux olympiques et paralympiques 2024. Par le passé, ce mode opératoire a également été utilisé par le GRU dans le sabotage de la chaîne de télévision TV5Monde en 2015, ainsi que dans la tentative de déstabilisation du processus électoral français en 2017.

APT28 est aussi employé pour exercer une pression constante sur les infrastructures ukrainiennes dans le contexte de la guerre d'agression menée par la Russie contre l'Ukraine, notamment lorsqu'il est opéré par l'unité 20728 du GRU. De nombreux partenaires européens ont également été visés par APT28 au cours des dernières années. À ce titre, l'UE a imposé des sanctions aux personnes et entités responsables des attaques menées à l'aide de ce mode opératoire.

6

APT28

(Fancy Bear, Sofacy, Sednit, etc.)

- **Publication de nombreux rapports**
  - Exemple : Rapport Mueller aux USA
- **À partir de 2018/2019 : Baisse de visibilité d'APT28**
  - Toujours présents mais moins visibles

APT28

Ciblage actuel

- **2022 Invasion à grande échelle de l'Ukraine**
- **Regain de visibilité : support à la guerre**
- **Ciblage : Ukraine, OTAN et France**

8

APT28

Exemples de campagnes

- **Phishing société civile**
  - Ciblage de la plateforme UKR.NET

- **Support opérationnel : ciblage les caméras connectées**
  - Suivi du transport de matériel
  - Ukraine et pays limitrophes

9

APT28

Quelques TTPs

- **Compromission d'équipements réseaux *// edge devices***
  - Routeurs, pare-feu, etc.
  - Utilisés comme relais pour réaliser les attaques
- **Mettre à jour / Appliquer les patchs de sécurité**

APT28

Codes malveillants

› Évolution dans les codes malveillants

- › 2004 - 2018 : “*malware signature*” : X-agent

11

APT28

Codes malveillants

› Évolution dans les codes malveillants

~~› 2004 - 2018 : “*malware signature*” : X-agent~~

› 2018 - 2022 : ???

12

APT28

Codes malveillants

› Évolution dans les codes malveillants

~~› 2004 - 2018 : “*malware signature*” : X-agent~~

~~› 2018 – 2022 : ???~~

› 2022 - 2024 : Remplacés par des codes “jetables”

› Masepie, Steelhook, Oceanmap, etc.

13

APT28

Codes malveillants

› Évolution dans les codes malveillants

~~› 2004 - 2018 : “*malware signature*” : X-agent~~

~~› 2018 – 2022 : ???~~

~~› 2022 - 2024 : Remplacés par des codes “jetables”~~
~~› Masepic, Steelhook, Oceanmap, etc.~~

› Depuis 2024, retour à des implants plus sophistiqués

14

APT28

Activités actuelles

- **Nouveaux “implants signature” :**
  - Deux backdoors :
    - BeardShell (C++)
    - Covenant (.NET, Open Source, mais repris par APT28)
  - Un keylogger : Slimagent

15

APT28

Implants

› Analyse des malwares :

https://blog.sekoia.io/apt28-operation-phantom-net-voxel

https://www.welivesecurity.com/en/eset-research/sednit-reloaded-back-trenches

16

Retex de l’analyse de ces malwares

17

APT28 - BeardShell

- **Difficultés de reverse**
  - Provient de l'analyse du C++
  - Temps nécessaire pour recréer les structures
  - Long mais pas de problématiques particulières

18

APT28 – Covenant

```
public void CCDLKA785C()
{
    try
    {
        string text = L2WSH3CNPJ.W8Y70HMX0S("SmEpdTdyJQ==").Replace(Environment.NewLine, L2WSH3CNPJ.W8Y70HMX0S("Ow=="));
        string text2 = L2WSH3CNPJ.W8Y70HMX0S("SmEpdTdyJQ==").Replace(Environment.NewLine, L2WSH3CNPJ.W8Y70HMX0S("Ow=="));
        string text3 = L2WSH3CNPJ.W8Y70HMX0S("CTR1PH8iaXRuNw==");
        string text4 = L2WSH3CNPJ.W8Y70HMX0S("dD02MgMMO242C0ccGz8mAisSFxpbOho0DQ8JCDoldDIjITlzby1uJgljYGQ=");
        string text5 = PJ8YF3UG0C.EID5TXAGCS(text4);
        PJ8YF3UG0C.B5OOGU8HM3 b5OOGU8HM = new PJ8YF3UG0C.B5OOGU8HM3(text4);
        PJ8YF3UG0C.EROYOYIZ6E eroyoyiz6E = new PJ8YF3UG0C.EROYOYIZ6E();
        ICryptoTransform cryptoTransform = b5OOGU8HM.RI89Q0UVYZ();
        HMACSHA256 hmacsha = new HMACSHA256(b5OOGU8HM.KELYRW0EEX());
        Thread.Sleep(TimeSpan.FromSeconds(2.0));
        RSACryptoServiceProvider rsacryptoServiceProvider = new RSACryptoServiceProvider(2048, new CspParameters());
        byte[] bytes = Encoding.UTF8.GetBytes(rsacryptoServiceProvider.ToXmlString(false));
        byte[] array = b5OOGU8HM.JWI0CQADMU(cryptoTransform, bytes, 0, bytes.Length);
        byte[] array2 = hmacsha.ComputeHash(array);
        string text6 = L2WSH3CNPJ.W8Y70HMX0S("HzM9Nw==");
```


19

APT28 - Covenant

Base64 / Chiffrement

```
public void CCDLKA785C()
{
    try
    {
        string text = L2WSH3CNPJ.W8Y70HMX0S("SmEpdTdyJQ==").Replace(Environment.NewLine, L2WSH3CNPJ.W8Y70HMX0S("Ow=="));
        string text2 = L2WSH3CNPJ.W8Y70HMX0S("SmEpdTdyJQ==").Replace(Environment.NewLine, L2WSH3CNPJ.W8Y70HMX0S("Ow=="));
        string text3 = L2WSH3CNPJ.W8Y70HMX0S("CTR1PH8iaXRuNw==");
        string text4 = L2WSH3CNPJ.W8Y70HMX0S("dD02MgMMO242C0ccGz8mAisSFxpbOho0DQ8JCDoldDIjITlzby1uJgljYGQ=");
        string text5 = PJ8YF3UG0C.EID5TXAGC
        PJ8YF3UG0C.B5OOGU8HM3 b5OOGU8HM = new PJ8YF3UG0C.B5OOGU8HM3(text4);
        PJ8YF3UG0C.EROYOYIZ6E eroyoyiz6E = new PJ8YF3UG0C.EROYOYIZ6E();
        ICryptoTransform cryptoTransform = b5OOGU8HM.RI89Q0UVYZ();
        HMACSHA256 hmacsha = new HMACSHA256(b5OOGU8HM.KELYRW0EEX());
        Thread.Sleep(TimeSpan.FromSeconds(2.0));
        RSACryptoServiceProvider rsacryptoServiceProvider = new RSACryptoServiceProvider(2048, new CspParameters());
        byte[] bytes = Encoding.UTF8.GetBytes(rsacryptoServiceProvider.ToXmlString(false));
        byte[] array = b5OOGU8HM.JWI0CQADMU(cryptoTransform, bytes, 0, bytes.Length);
        byte[] array2 = hmacsha.ComputeHash(array);
        string text6 = L2WSH3CNPJ.W8Y70HMX0S("HzM9Nw==");
```


20

APT28 – Covenant

```
public void CCDLKA785C()
{
    try
    {
        string text = L2WSH3CNPJ.W8Y70HMX0S("SmEpdTdyJQ==").Replace(Environment.NewLine, L2WSH3CNPJ.W8Y70HMX0S("Ow=="));
        string text2 = L2WSH3CNPJ.W8Y70HMX0S("SmEpdTdyJQ==").Replace(Environment.NewLine, L2WSH3CNPJ.W8Y70HMX0S("Ow=="));
        string text3 = L2WSH3CNPJ.W8Y70HMX0S("CTR1PH8iaXRuNw==");
        string text4 = L2WSH3CNPJ.W8Y70HMX0S("dD02MgMMO242C0ccGz8mAisSFxpbOho0DQ8JCDoldDIjITlzby1uJgljYGQ=");
        string text5 = PJ8YF3UG0C.EID5TXAGCS(text4);
        PJ8YF3UG0C.B5OOGU8HM3 b5OOGU8HM = new PJ8YF3UG0C.B5OOGU8HM3(text4);
        PJ8YF3UG0C.EROYOYIZ6E eroyoyiz6E = new PJ8YF3UG0C.EROYOYIZ6E();
        ICryptoTransform cryptoTransform = b5OOGU8HM.RI89Q0UVYZ();
        HMACSHA256 hmacsha = new HMACSHA256(b5OOGU8HM.KELYRW0EEX());
        Thread.Sleep(TimeSpan.FromSeconds(2.0));
        RSACryptoServiceProvider rsacryptoServiceProvider = new RSACryptoServiceProvider(2048, new CspParameters());
        byte[] bytes = Encoding.UTF8.GetBytes(rsacryptoServiceProvider.ToXmlString(false));
        byte[] array = b5OOGU8HM.JWI0CQADMU(cryptoTransform, bytes, 0, bytes.Length);
        byte[] array2 = hmacsha.ComputeHash(array);
        string text6 = L2WSH3CNPJ.W8Y70HMX0S("HzM9Nw==");
```

Analyse des fonctions & renommage

21

APT28 - Covenant

- **“Automatisation” réalisée :**
  - Script python (dnlib / pythonnet)
  - Renommer chaque fonction manuellement

22

APT28 - Covenant

- **“Automatisation” réalisée :**
  - Script python (dnlib / pythonnet)
  - ~~Renommer chaque fonction manuellement~~
  - Copier-coller dans ChatGPT

- **Aucune difficulté technique à reverser ce code avec l’IA**

23

APT28 - Covenant

- **“Automatisation” réalisée :**
  - Script python (dnlib / pythonnet)
  - ~~Renommer chaque fonction manuellement~~
  - Copier-coller dans ChatGPT

- **Aucune difficulté technique à reverser ce code avec l’IA**

- **Difficulté émotionnelle : l’humain (moi) est automatisé par l’IA**

- **De nombreux samples. Besoin d’une meilleure automatisation**

24

APT28 - Slimagent

- **C/C++**

- **Fonctionnalités**
  - Keylogging
  - Screenshot à interval régulier (5secs)

- **Données chiffrées puis sauvegardées**
  - Fichier Desktop.svc

APT28 - Slimagent

- **Cryptographie :**
  - Desktop.svc : chiffrement à base d'AES et de RSA
  - Simple mais…

26

APT28 – Slimagent

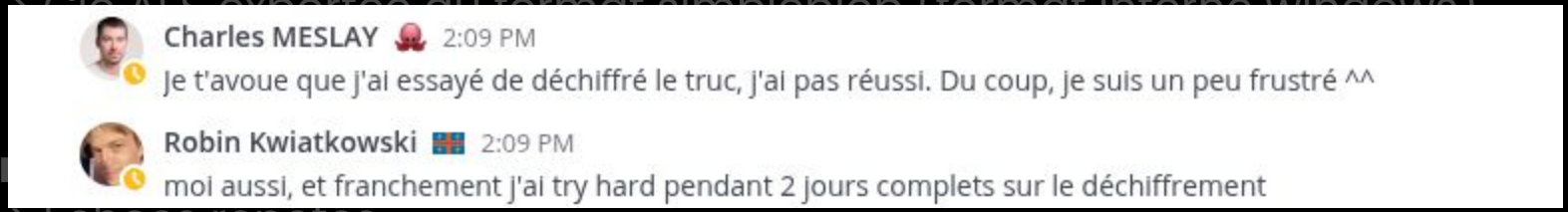
- **Cryptographie :**
  - Desktop.svc : chiffrement à base d'AES et de RSA
  - Simple mais…

- **On n'a jamais réussi à scripter le déchiffrement**
  - Échecs répétés
  - Problématique liée au format de la clé RSA exportée

27

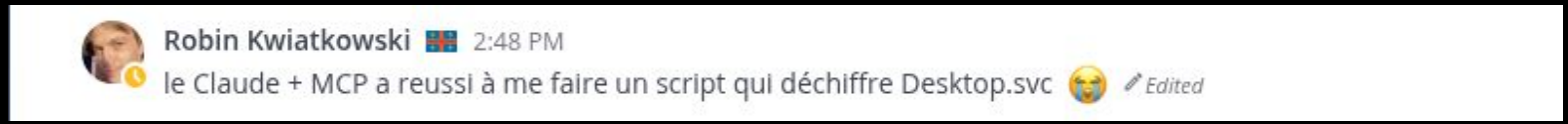
APT28 – Slimagent

› **Cryptographie :**
  › Contient une clé publique RSA hardcodée
  › Clé AES exportée au format simpleblob (format interne windows)



› **Ter**
  › Échecs répétés
  › Problématique de formats “non concordant”

Puis, deux mois plus tard...


Robin Kwiatkowski 2:48 PM
le Claude + MCP a reussi à me faire un script qui déchiffre Desktop.svc
Edited

Robin Kwi
le Claude


Robin Kwiatkowski 3:17 PM
il est meilleur que moi et il coûte 20€/mois
Charles MESLAY 3:17 PM
"que nous"
1

Robin Kwi
il est meill

Charles M
"que nous

1

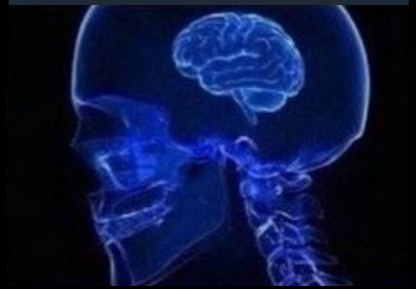
“On n’a pas le choix, il faut vraiment intégrer l’IA à nos analyses”

34

Évolution du job de reverser 2022-SSTIC26

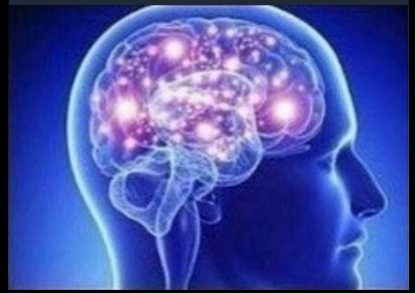
Étape 1

De Ghidra, copier-coller dans un chatbot pour analyser les fonctions une par une.



Étape 2

Créer une extension ou un plugin Ghidra pour faire click-droit : analyser + renommer.



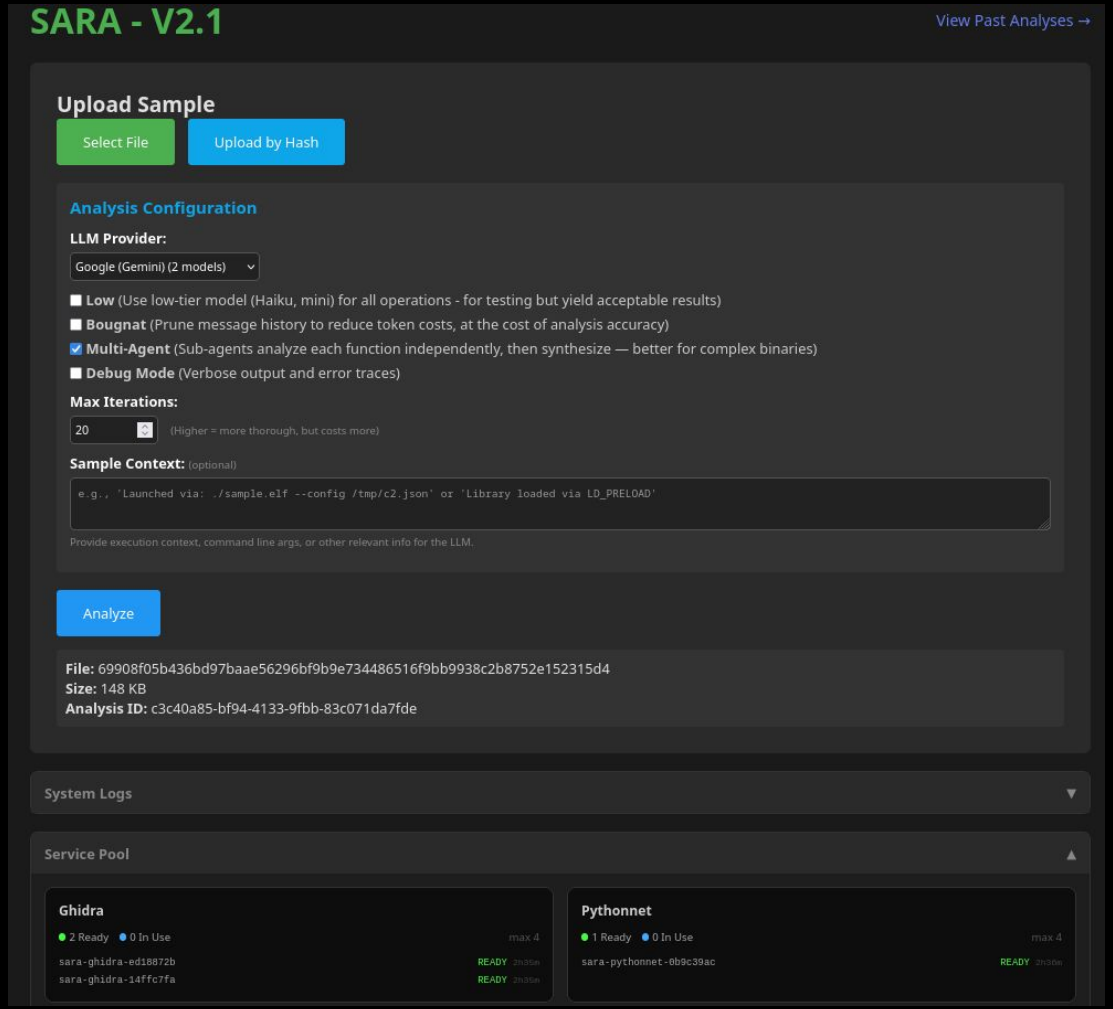
Étape 3

Connecter un chatbot à Ghidra via un serveur MCP.

Étape 4

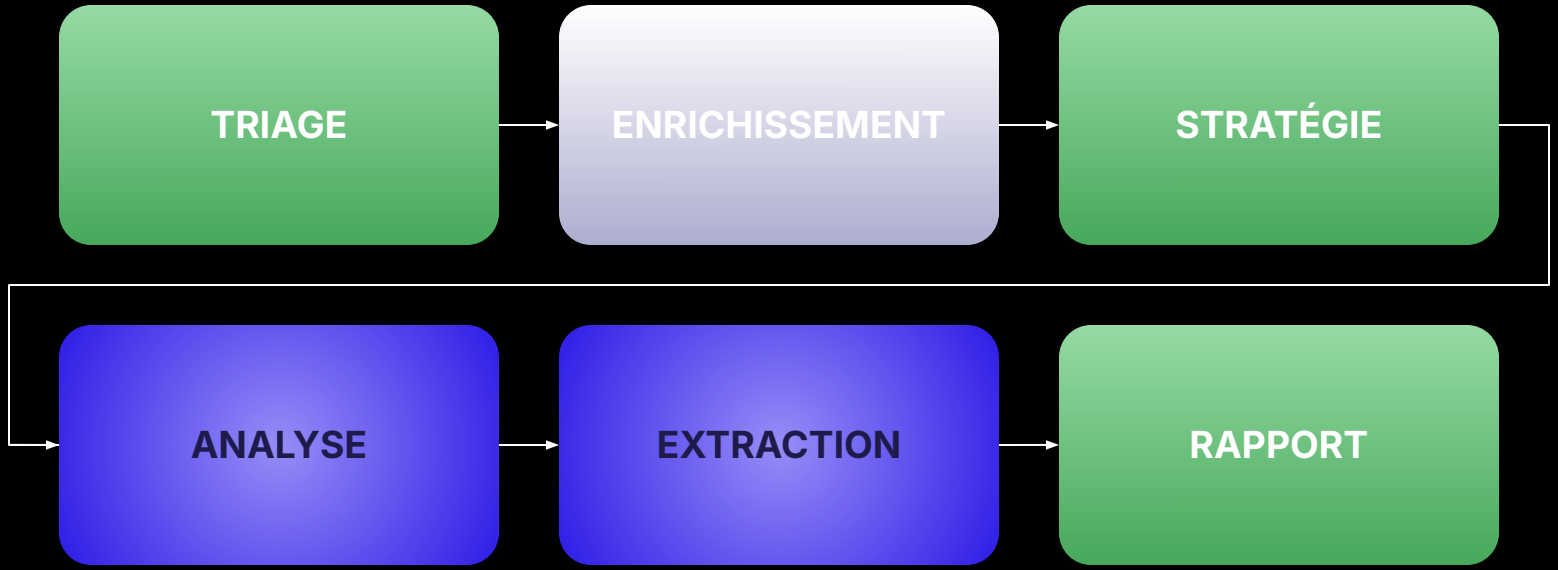
Se passer du chatbot pour automatiser.

35



Self Autonomous Reverse Analyst

Workflow LangGraph



Triage & Stratégie

Triage Node

Objectif :

Déterminer le type de fichier et choisir le pipeline d'analyse (Ghidra, pythonnet, etc.). Sélectionne les modèles.

Sources de données :

- Binaire (SHA-256) & python-magic
- Parsing PE-header (DLL detection)
- LiteLLM proxy / ModelSelector

Appel LLM (LOW tier) :

"Sélectionne le pipeline le plus spécialisé. Format JSON : {"pipeline": "..."}"

Strategy Node

Objectif :

Choisir un fichier de stratégie comme prompt système pour le nœud d'Analyse.

Sources de données :

- Fichiers config/strategies/{pipeline}/
- Champs d'état : type, packed, context
- Modèle LOW tier choisi au triage

Appel LLM (LOW tier) :

"Sélectionne la stratégie pertinente. Format JSON : {"strategy": "..."}"

38

```
[4:30:18 PM] ================================================================================
[4:30:18 PM] NODE: File Triage
[4:30:18 PM] ================================================================================
[4:30:18 PM] File: e386476ca9a54022baa1ec8ee14594ba6567e28899b244741efdf4ef9394c773
[4:30:18 PM] Type: PE32+ executable for MS Windows 6.00 (DLL), x86-64, 7 sections (MIME: application/
vnd.microsoft.portable-executable)
[4:30:18 PM] Hash: e386476ca9a54022baa1ec8ee14594ba6567e28899b244741efdf4ef9394c773
[4:30:18 PM] Is DLL: True
[4:30:18 PM] LLM-Based Pipeline Selection:
[4:30:18 PM]    Available pipelines: ghidra, pythonnet, manual_review
[4:30:18 PM]     Models selected:
[4:30:18 PM]       Low tier:  gemini/gemini-3.1-flash
[4:30:18 PM]       High tier: gemini/gemini-3.1-pro
[4:30:18 PM]    LLM RAW RESPONSE: '{"pipeline": "GHIDRA"}
This file is a native PE32+ DLL (x86-64), which is a compiled binary format. GHIDRA is the appropriate tool for
decompiling and analyzing native machine code, whereas PYTHONNET is specifically for .NET/MSIL assemblies, which this
file is not identified as.'
[4:30:18 PM]    [OK] LLM selected: ghidra
[4:30:18 PM] [OK] Pipeline: ghidra
[4:30:18 PM] [OK] Confidence: 0.95
```


39

Serveurs MCP

MCP ?

Model Context Protocol : standard ouvert proposé par Anthropic (2024) pour connecter les LLM aux outils externes.

Dans **SARA**, il sert de pont avec **Ghidra/Pythonnet** :

- Expose des outils (ex: decompile_function).
- Traduit les requêtes LLM en opérations natives.
- Permet au LLM de lister dynamiquement les capacités selon le pipeline.

ghidra-mcp-headless

Infrastructure Docker : Ghidra-headless + Serveur MCP

28 outils disponibles :

- 7 **Gestion** (chargement du sample, gestion des artefacts)
- 21 **Analyse** (décompilation, références)

Remarque:
L'outil Find_Strings est souvent sur-utilisé par les LLM, en cherchant des strings qui potentiellement existe dans les malwares i.e : cmd.exe, http, https, ...

40

Noeud d'Analyse

Mode agent unique

Analyse séquentielle : fonctionnement similaire à la connexion d'un chatbot avec un serveur MCP d'analyse.

✓ Point fort

Efficace sur des petits samples.

× Limitations

- Coût en tokens quadratique.
- Atteinte rapide de la fenêtre de contexte maximum.

Mode multi-agent

Un agent **coordinateur** orchestre des agents de **classification**. Les agents de **classification** reçoivent le code et le context, l'orchestrateur ne voit jamais de code.

✓ Point fort

Coût en token linéaire.

× Limitations

- Pas de contexte global complet
- nécessite un noeud d'extraction pour extraire des données qui subissent plusieurs transformations (déchiffrement, deobfuscation, ...)

41

Mode multi-agent

Agent Coordinateur

Espace de décision (LLM)

- Quelles fonctions regarder ensuite ?
- Quel outil de navigation est le moins coûteux ?
- Cette branche vaut-elle la peine d'être suivie ?
- L'analyse est-elle terminée ?

Tâches Déterministes

- Chargement du binaire & graphe d'appels
- Mapping des hits CAPA
- Gestion de l'état (pending vs done)
- Mise à jour de la structure de données

Agent Classificateur

Espace de décision (LLM)

- Verdict : intéressant, neutre ou standard_lib ?
- Nom descriptif & résumé d'une ligne
- Extraction d'IOCs visibles
- Besoin d'extraction de données ?

Tâches Déterministes

- Filtrage des fonctions courtes (< 50 chars)
- Sélection des appelants et appelés à inclure
- Parallelisation des workers (3-worker pool)

42

Choix des modèles

LiteLLM Proxy

Abstraction du format des messages

Provider testés

- Anthropic (Opus - haiku)
- OpenAI (gpt - mini)
- Google (Pro - flash)

Provider intéressant pour la suite

- Xiaomi (Mimo)
- Z.ai (GLM)

Agent unique

Modèle HIGH tier

- Choix des tools a utiliser
- Choix d'exploration
- Analyse du code

Modèle low tier

- choix de la stratégie
- choix de la pipeline

Multi-agent

Modèle HIGH tier

- phase de synthèse
- création du rapport
- Extraction des données

Modèle low tier

- choix de la stratégie
- choix de la pipeline
- Analyse des fonctions par le classifier
- Coordinateur

43

Coûts, tokens & résultats

Agent unique

[Total] $2.3911

[Fonction traitées] 67 en 30 iterations
[LOW] Low tier: $0.0106 (23498 in, 3123 out)
[HIGH] High tier: $2.3806 (1145900 in, 7398 out)
[prix par fonction] $0.036

Multi agent

[Total] $0.5375

[Fonction traitées] 145
[LOW] Low tier: $0.4711 (1041904 in, 148436 out)
[HIGH] High tier: $0.0664 (10066 in, 3849 out
[prix par fonction] $0.0038

44

Coûts, tokens & résultats

1) Summary

- The analyzed binary is a sophisticated Windows DLL acting as an infostealer or spyware.
- It features extensive user surveillance capabilities, including keylogging, clipboard monitoring, active window tracking, and screen capture.
- Stolen data is staged locally in temporary files and encrypted using AES-256 via the Windows CryptoAPI.
- The malware employs anti-analysis techniques, including dynamic API resolution and anti-debugging via software breakpoints (`int 3`).
- Execution is gated by a hardcoded mutex (`hey4kmr8oj46n45n3p`) to ensure a single instance runs.

2) Execution flow

1. `entry`: Initializes security cookies and dispatches to the DLL main handler.
    - calls `dllmain_dispatch` for process attachment logic.
2. `dllmain_dispatch`: Handles DLL initialization.
    - calls `FUN_1800013c0` to load `eapphost.dll` and spawn the main thread.
3. `FUN_1800013c0`: Dynamically resolves APIs and starts the primary malware routine.

45

Coûts, tokens & résultats

3) Capabilities

- Keylogging (proof: FUN_180003790 uses `GetKeyboardState` and `ToUnicodeEx` to log keystrokes)
- Clipboard monitoring (proof: FUN_180004080 uses `OpenClipboard` and `GetClipboardData` to read CF_UNICODETEXT)
- Screen capture (proof: FUN_1800029b0 uses `EnumDisplayMonitors` and `CryptBinaryToStringA` to capture and encode screen data)
- Anti-debugging (proof: FUN_1800013c0 implements `int 3` software breakpoint checks)
- Mutex-based single instance (proof: FUN_180001230 uses `CreateMutexA` with string `hey4kmr8oj46n45n3p`)
- Dynamic API resolution (proof: FUN_1800013c0 resolves function pointers at runtime)

4) Key functions

- entry, (entry): Program entry point, initializes security cookies.
- dllmain_dispatch, (dllmain_dispatch): Handles DLL process attachment and thread spawning.
- FUN_1800013c0, (unknown): Dynamically resolves APIs and initiates malware routine.
- FUN_180001230, (unknown): Performs mutex check and configuration initialization.
- FUN_1800022d0, (unknown): Orchestrates worker threads for data staging and monitoring.

46

Coûts, tokens & résultats

- `filepath: %SystemRoot%\System32\eapphost.dll` — source: init_eapphost_and_spawn_thread, dllmain_dispatch_with_eapphost_injection
- `string: <img src="` — source: capture_screen_to_base64_html
- `string: "/><br>` — source: capture_screen_to_base64_html
- `string: RoInitialize` — source: resolve_and_call_RoInitialize
- `string: api-ms-` — source: resolve_api_function_with_caching
- `string: ext-ms-` — source: resolve_api_function_with_caching
- `filepath: %TEMP%\Desktop_` — source: construct_temp_file_path
- `string: .svc` — source: construct_temp_file_path
- `string: %d-%m-%Y_%H-%M-%S` — source: construct_temp_file_path
- `algorithm: SHA-256 (0x800c)` — source: hash_and_concatenate_data
- `string: image/jpeg` — source: capture_screen_to_jpeg
- `config: 0x180052c80` — source: FUN_180001230 via static_read
- `strings: ['OnSessionChange', 'DllCanUnloadNow', 'DllGetClassObject', 'DllRegisterServer', 'DllUnregisterServer', 'InitializeEapHost', 'StopServiceOnLowPower', 'UninitializeEapHost']` — source: FUN_1800013c0 via pointer_dereference
- `strings: ['<font color=blue>', '<font color=red>', '<font color=green>', '<font color=black>Clipboard :: ', '</font><br>']`

47

Limites et usages

Limites

Junk code / faux appels API / obfuscation

Sara a du mal à différencier le code utiles ou non. Échoue lamentablement en cas de control flow flattening.

Langages

2 pipelines sont disponibles pour le moment, Ghidra et Pythonnet. Les langages mals supportés par ghidra donnent de très mauvais résultats (Mach-O)
Les langages objet comme le C++ donnent aussi de mauvais résultat (pas de création de structures)

Format de fichier

Ne fonctionne que sur des PE ou ELF, pas sur les lnk, script, docx, ...

Pas de récursivité

Si un stage 2 est extrait, il n'est pas automatiquement re-injecté pour une analyze.

Usages

Tri

Permet assez bien de trier des fichiers, savoir si ils sont malveillant ou bénin

Comparaison

analyser deux binaires pour confirmer s'ils appartiennent à une même famille rapidement

Pré-analyze

Avant chaque analyse manuelle, l'analyse de sara permet de pré-mâcher le travail, pour n'avoir qu'à valider le rapport, ou creuser certain aspect

48

Conclusion

- **Reste un travail de recherche**
  - Publication du MCP PythonNET
  - https://github.com/SEKOIA-IO/RePythonNET-MCP
- **Gain de temps important**
- **Permet aux juniors de s'initier au reverse**
- **Roadmap :**
  - Analyse dynamique
  - De nouvelles pipelines (Mach-O / jadx)
  - Intégration à nos autres pipelines

49

Merci !

Avez-vous des questions ?

Nos investigations

Retrouvez nos dernières analyses sur notre blog :

blog.sekoia.io

Contactez-nous

Pour échanger avec l'équipe :

tdr@sekoia.io