Dynamic Malware Analysis

For Dummies

SSTIC08 – 06/06/2007 – http://www.sstic.org

Philippe Lagadec – NATO/NC3A
philippe.lagadec(à)nc3a.nato.int

Scénario type: Gestion d'incident

- **Un fichier suspect (exécutable ou document) a été détecté/capturé.**
- On veut savoir rapidement ce que fait le code malveillant, mais on n'a pas besoin d'une analyse exhaustive.
  - Cas différent d'un éditeur antivirus.
- **Analyse en temps réduit**: quelques heures ou quelques jours maximum.
- L'analyste n'est pas expert en assembleur.

Objectifs de l'analyse

- Le fichier suspect est-il malveillant ?
- Est-ce un fichier connu ou bien une attaque ciblée ?
- Quelles sont sa nature et ses fonctionnalités ?
- Quels sont les risques/conséquences ?
- Comment le détecter et l'éradiquer sur le réseau ? (forensics, gestion d'incident)
- Comment se protéger à l'avenir ?

Méthodes d'analyse

1. Désassemblage / Décompilation
2. Débogage
3. Exploration de fichiers
4. Analyse dynamique « runtime »

Désassemblage / Décompilation

- Outil classique: IDA Pro
- **Avantages:**
  - Vue complète et détaillée du code
  - Outils d'analyse évolués
- **Inconvénients:**
  - Nécessite un haut niveau de compétence et d'expérience (i.e. un gourou du reverse)
  - Peut prendre beaucoup de temps
  - Des techniques de camouflage (packer, VM, etc) peuvent compliquer énormément la tâche.

Débogage

- Outils: Ollydbg, Softice, Syser Debugger, …
- **Avantages:**
  - Vision pas-à-pas de l'exécution
  - Complémentaire du désassemblage si packer
- **Inconvénients**:
  - Aussi besoin de compétences/expérience poussées
  - Méthodes de détection/blocage du débogueur peuvent ralentir l'analyse

Exploration de fichiers

- Parfois aussi appelée « analyse statique »
- Divers outils permettent d'extraire beaucoup d'informations d'un fichier suspect sans l'ouvrir:
  - Éditeur Hexa
  - Recherche de chaînes / regex
  - Explorateurs de formats: PE, OLE2 (MS Office), Zip, …
  - Méta-données des documents
  - File carving (forensics)
  - Antivirus
  - Scanners d'exploits connus: OfficeCat, Fess

Exploration de fichiers

- **Avantages**:
  - Extraction simple de beaucoup d'informations
  - Les auteurs de malware cachent rarement toutes leurs traces.
- **Inconvénients**:
  - Ne fonctionne pas si camouflage

Analyse dynamique « runtime »

- Diverses appellations en anglais:
  - dynamic analysis, runtime analysis, real-time analysis, behavioral analysis
- **Principe**: **exécuter** réellement le code malveillant pour **observer** son comportement et ses effets

Analyse dynamique « runtime »

- **Avantages**:
  - Le code malveillant s'exécute dans un **environnement quasi-naturel**: peu sensible aux protections anti-debug et anti-désassemblage
  - Aperçu rapide du comportement du code
- **Inconvénients**:
  - Ne montre que le code qui s'exécute directement
    - Cible principale: malware simple, virus, ver ou Cheval de Troie
  - Ne montre pas l'intégralité du code

Méthodes d'analyse

	(relativement) **simple, rapide mais partiel**	(Un peu plus) **complexe mais efficace**
Analyse **Statique**	Exploration de fichiers	Désassemblage Décompilation
Analyse **Dynamique**	Analyse runtime	Débogage

Méthode proposée

- **Première étape en gestion d'incident:**
  - **1) Exploration de fichiers**
  - **2) Analyse dynamique**
    - **=> résultats rapides mais partiels**

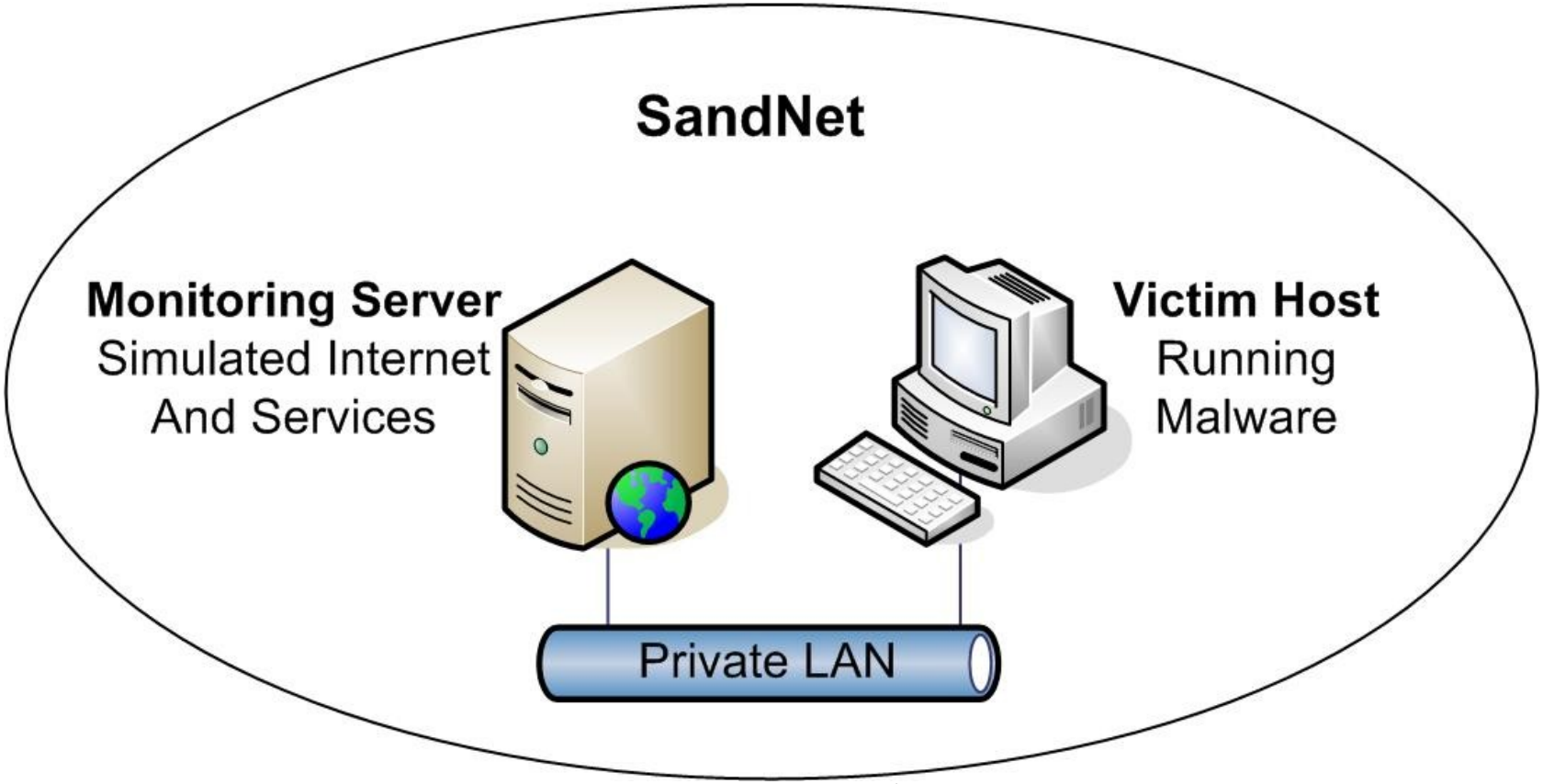
- Si nécessaire, et si le temps le permet:
  - 3) Désassemblage / décompilation
  - 4) Débogage

Un laboratoire d'analyse

- **Objectifs**:
  - Le malware doit s'exécuter naturellement
  - Observation de ses **actions locales + réseau**
  - Réseau offline, non connecté
- **Sandnet**: sandbox + Internet simulé
- Architecture la plus simple: 2 machines
  - Soit **virtuelles**, avec réseau virtuel
  - Soit **physiques**, reliées par un câble croisé

Sandnet

- Sandnet virtuel ou réel:



Machine 1: le faux serveur

- **Simule une connection Internet pour le malware:**
  - Passerelle par défaut
  - Répond pour n'importe quelle adresse IP
  - Faux serveur DNS: renvoie toujours son adresse IP
  - Faux serveurs: HTTP, proxy, SMTP, IRC, …
- **Enregistre toute activité réseau:**
  - Sniffer
  - Logs des serveurs
  - IDS

Machine 2: la victime

- **Le malware doit s'exécuter sans entrave**:
  - Surtout pas d'antivirus ou de pare-feu personnel !
  - OS non patché, correspondant à la cible du malware
  - Compte admin
  - Applications non patchées si besoin (navigateurs, Office, …)
- **Enregistrement des actions locales,** en temps réel et/ou par delta avant/après:
  - Processus, Services
  - Fichiers créés/modifiés/supprimés
  - Base de registre

Sandnet virtuel ou réel ?

	VIRTUEL	REEL
Avantages	Très efficace: Snapshots, Extensible	Réaliste Non vulnérable
Inconvénients	Détectable Vulnérable ?	Lourd: Sauvegardes / restaurations de disque

Sécuriser VMware Workstation

- **Quelques paramètres simples pour inhiber certaines méthodes de détection:**
  - Ne pas installer les VMware Tools
  - Désactiver drag&drop, copier-coller, shared folders
  - Désactiver toute accélération
    - Cf. http://www.trapkit.de/research/vmm/index.html
- **Interface réseau: utiliser une VM Team**
  - Pas de bridge, host-only ou NAT

Méthode: Exploration de fichiers

- **1) Rechercher des patterns intéressants**:
  - Éditeur hexa, Recherche de regex (reScan)
  - Exemples: adresses IP, URLS, NOPs (0x90) pour un shellcode, entête PE dans un document malformé, etc…
- **2) test antivirus / scanner d'exploits**:
  - Vérifier si c'est un malware connu
  - Scanner d'exploits connus: OfficeCat, Fess
- **3) Explorateurs de formats**:
  - EXE: PEiD, pefile, scanbin, Dependency Walker, …
    - => détecter si compression ou packer connu, DLLs employées
  - Office: SSViewer, OleFileIO
    - => Structure OLE2, Méta-données

Méthode: Exploration de fichiers

- **4) File carving (forensics)**:
  - Hachoir-subfile, Scalpel, Foremost
  - Permet d'extraire un EXE inclus dans un document mal formé, par exemple.

Exemple: reScan

```
"IP addresses": r"\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}",
"EXE MZ headers": r"MZ|ZM",
"EXE PE headers": r"PE",
"EXE PE DOS message": r"(?i)This program cannot be run in DOS mode",
".EXE/.COM/.VBS/.JS/.BAT/.CMD/.DLL filename": r"(?
    i)\.EXE|\.COM|\.VBS|\.JS|\.VBE|\.JSE|\.BAT|\.CMD|\.DLL",
"EXE: UPX header": r"(?i)UPX",
"EXE: .text/.data/.rdata section": r"(?i)\.text|\.data|\.rdata",
"EXE: packed with Petite": r"(?i)\.petite",
"EXE: interesting Win32 function names": r"(?i)WriteFile|IsDebuggerPresent|RegSetValue|
    CreateRemoteThread",
"EXE: interesting WinSock function names": r"(?i)WS2_32\.dll|WSASocket|WSASend|WSARecv",
"EXE: possibly compiled with Microsoft Visual C++": r"(?i)Microsoft Visual C\+\+",
"Interesting registry keys": r"(?i)CurrentVersion\\Run|UserInit",
"Interesting file names": r"(?i)\\drivers\\etc\\hosts|cmd\.exe|\\Start Menu\\Programs\\Startup",
"Interesting keywords": r"(?i)password|administrator|smtp|pop|http|ftp|ssh|icq|backdoor|vmware",
"NOP instructions (possible shellcode)": r"\x90{4,}", # this regex matches 4 NOPs or more
"Possible OLE2 header (D0CF)": r"\xD0\xCF\x11\xE0",
"VBA macros": r"(?i)VBA",
```

Exemple: Fess

alert (msg:"**PNG File**"; content:"|89|PNG|0D0A1A0A|"; depth:8; flowbits:noalert; flowbits:set,ispng; reference:url,www.png.org; fid:1;)

alert (msg:"**JPEG File**"; content:"|FF D8 FF|"; depth:3; flowbits:noalert; flowbits:set,isjpeg; reference:url,www.jpeg.org; fid:2;)

alert (msg:"**PE File**"; content:"MZ"; depth:2; byte_jump:4,60; content:"PE"; within:2; flowbits:noalert; flowbits:ispe; fid:3;)

alert (msg:"**PE File with EntryPoint outside of code section**"; content:"MZ"; depth:2; byte_jump:4,60; content:"PE"; within:2; byte_save:4,46,BaseOfData,relative; byte_test:4,>,BaseOfData,38,relative; fid:4;)

alert (msg:"**PNG with PLTE chunk**"; flowbits:isset,ispng; flowbits:set,ispngplte; flowbits:noalert; loop:4,"PLTE",big; start:8; fid:5;)

alert (msg:"**PNG tRNS overflow**"; flowbits:isset,ispng; flowbits:isnotset,ispngplte; loop:4,"tRNS",big; start:8; byte_test:4,>,256,-8,relative,big; fid:6;)

alert (msg:"**Standard WMF**"; byte_test:2,<,3,0,little; content:"|09 00|"; content:"|00 00|"; distance:12; within:2; flowbits:set,wmf; flowbits:noalert; fid:7;)

alert (msg:"**Placeable WMF**"; content:"|D7 CD C6 9A 00 00|"; depth:6; flowbits:set,wmf; flowbits:noalert; fid:8;)

alert (msg:"**SETABORTPROC Escape function in WMF (possible MS06-001 exploit)**"; flowbits:isset,wmf; content:"|26|"; content:"|09 00|"; distance:1; within:2; fid:9;)

alert (msg:"**BMP with invalid bfOffBits (possible MS06-005)**"; content:"BM"; depth:2; content:"|0000000000000000|"; distance:4; within:8; reference:url,www.microsoft.com/technet/security/Bulletin/ms06-005.mspx; fid:10; rev:1;)

alert (msg:"**OSX/Oomp-A**"; content:"|FEEDFACE|"; depth:4; content:"oompa"; content:"kMDItemLastUsedDate|203e3d2024| time.this_month"; reference:url,www.ambrosiasw.com/forums/index.php?showtopic=102379; fid:11; rev:1;)

Méthode: analyse runtime

- **1) Enregistrer l'état initial du poste victime**
  - Snapshot ou image disque saine
  - Liste complète des fichiers du disque
  - Base de registre (complète + clés "run")
  - Liste des processus, drivers et services
- **2) Lancer les outils d'observation**
  - Process monitor (remplace FileMon et RegMon)
  - Process explorer
  - Outils réseau sur le poste d'observation

Méthode: analyse runtime

- **3) Lancer le code malveillant**
  - Avec Sysanalyzer pour enregistrer les appels Win32 et dumper l'image mémoire du malware (utile si packer)
- **4) Observer son comportement**
  - Confirmer s'il s'exécute
  - Adapter le poste d'observation si besoin: simuler adresses IP et serveurs nécessaires
- **5) Enregistrer l'état du poste victime après infection**
  - Idem étape 1
- **6) Comparer les états avant/après**

Vulnérabilités exploitées

- Document malformé: Il est souvent utile de déterminer quelle vulnérabilité est exploitée:
- OfficeCat, Fess ou un antivirus peut fournir ce résultat.
- Autre méthode:
  - 1) Partir d'une version vulnérable
  - 2) Installer progressivement chaque patch, prendre un snapshot
  - 3) Tester chaque fois le malware jusqu'à ce qu'il ne se lance pas.
  - 4) Sinon c'est un zero-day. ;-)
- Il faut souvent corréler les 2 méthodes car un patch corrige plusieurs vulnérabilités.

Résultats

- **Comportement “visible” du malware**:
  - Fichiers créés / modifiés / effacés
  - Clés de registre
  - Processus
  - Connexions réseau
- **Infos utiles pour IDS et forensics**:
  - Signatures réseau
  - Hashs MD5 des fichiers à rechercher

Automatisation

- La méthode décrite s'appuie sur de nombreux petits outils:
  - Processus manuel, fastidieux, erreurs possibles
- Comment l'automatiser ?
  - Scripts
  - Reboot / Snapshots automatisés:
    - Outils/API de virtualisation si virtuel (ex: vmrun pour VMware)
    - Boot PXE si réel
      - cf. http://www.secureworks.com/research/tools/truman.html
- Mais l'automatisation a ses limites: un cerveau humain est toujours le meilleur outil d'analyse…

Automatisation complète

- **S'il est possible d'automatiser tout le processus d'analyse, pourquoi ne pas l'employer sur une passerelle de filtrage ?**
  - Détection de malware inconnu sans signature !
- Problèmes:
  - Beaucoup trop de faux positifs / négatifs: critères d'analyse difficile à formaliser par des règles.
  - Beaucoup de cas particuliers
  - Temps d'analyse non négligeable

Pour aller plus loin

- Il existe de plus en plus de solutions “clés en main” pour automatiser le processus:
  - Soit **gratuites online**: interface web pour soumettre un échantillon
    - Exemples: Anubis, CWSandbox, Norman Sandbox, Joebox
    - Inconvénients: confidentialité, plate-forme contrainte, Win32 only
  - Soit **payantes offline**: environnement d’analyse complet
    - Exemples: CWSandbox, Norman Sandbox
    - Inconvénients: coût annuel, plutôt destiné à des analyses massives
  - **Diverses technologies**:
    - DLL injection+ API hooking, QEMU modifié, OS simulé

Conclusion

- **Il est possible d'analyser rapidement des codes malveillants sans recourir au désassemblage ou au débogage dans un premier temps.**
- **Un petit laboratoire d'analyse dynamique (sandnet) peut être construit à moindres frais.**
- Il serait utile de créer une distribution Linux "prête à l'emploi" et des scripts d'automatisation.
- Pour plus d'info:
  - http://www.decalage.info/sstic08
- …et bon week-end ! ;-)

Outils: analyse statique

- OfficeCat: http://www.snort.org/vrt/tools/officecat.html
- Fess: http://www.secureworks.com/research/tools/fess.html
- PEiD: http://peid.has.it/
- Pefile: http://dkbza.org/pefile.html
- Dependancy Walker: http://www.dependencywalker.com/
- Scanbin: http://jc.bellamy.free.fr/fr/scanbin.html
- Hachoir: http://hachoir.org/
- Foremost: http://foremost.sourceforge.net/
- Scalpel: http://www.digitalforensicssolutions.com/Scalpel/
- Structured Storage Viewer: http://www.mitec.cz/ssv.html
- OleFileIO: http://www.decalage.info/python/olefileio
- reScan: http://www.decalage.info/rescan

Fin

- <? Questions ?>

voir http://actes.sstic.org

et http://www.decalage.info

pour l'article correspondant à cette présentation, des outils et mises à jour.