The Spartan Project

A Modern Ransomware

June 2017

Amir Kotzer
Itamar Raviv
Ransomware – Project Guideline

◊ **Short life span** – Work time is usually ~20 minutes *.

◊ **Purpose oriented** – Encrypting all private files.

◊ **Transparency** - Acts as “legitimate” process.

◊ **Common pattern** – Infection → Persistancy → Weaponize → Encrypt.

*Based on a test VM with 2 CPUs (2 cores each) – About a GB/Minute*
Project Objectives

🔹 **Main task** - Encrypts the victim’s PC, decrypt for ransom.

🔹 **Evasion** - Evade detection tools on the local host.

🔹 **Redundancy** – “Survive” a detection, complete the main task.
Spartan Work Flow

**Spartan Decryptor**
When Encryption is Done, A Decryptor is created.

**Spartan Shield**
Engine Polling for encryption end.

**Spartan Spear Encryptor**
80% Copies itself
Disables:
Windows defender
System restore
Recovery
Security Services
UAC
Deletes Shadow Copies
Icon Creation
Boot Persistency
TurnOff
Defenses

**Spartan Engine**
Injects Engine

**Spartan Client**
USB Infector
Executes

**Spartan Dropper**
Downloads

**Azure Server**
Individual Public Key
Private Key

**USB Dropper**

**TurnOff Defenses**

**Spartan Decryptor**

**Word Macro**
Downloads

**TEMP**

The Spartan Project
Dear Spartan Warrior,

Greetings for joining the Spartan-Project family.

As a Spartan Warrior, all your personal files, documents and images were encrypted and spartanized.

If you wish to leave the Spartans family, you are welcome to pay the exit fine which is 1 BitCoin.

For more details, check out the “SpartanWarrior” directory on your desktop.

The Spartan Project
Infecting the Machine

- A Word document contains a malicious MACRO written as VBA script.
  - Downloads the “SPARTAN Dropper” executable, and executes it.
  - Infection using the Dropper directly.

- All SPARTAN files used by the dropper are stored in Azure storage server.
Spartan Engine (DLL)

- The SPARTAN Engine is the “brain” of the Ransomware.

- In charge of injecting the different SPARTAN components into well-known processes to hide the Ransomware’s activity.

- Cleaning the Ransomware footprints by periodically creating new system processes and re-injecting a DLL to it.

- Monitors the Ransomware progression to determine when the encryption has been finished.
Turning Defenses Off (DLL)

🔹 Disables Windows Defender by changing its Registry key
  🔹 SOFTWARE\Policies\Microsoft\Windows Defender\DisableAntiSpyware

🔹 Disables Windows System Restore (Functionality and Management)
  🔹 SOFTWARE\Policies\Microsoft\Windows NT\SystemRestore\DisableSR
  🔹 SOFTWARE\Policies\Microsoft\Windows NT\SystemRestore\DisableConfig

🔹 Disables Windows Recovery via shell execution (Functionality and Notifications)
  🔹 bcdedit.exe /set {default} recoveryenabled No
  🔹 bcdedit.exe /set {default} bootstatuspolicy ignoreallfailures
Turning Defenses Off (DLL) – cont.

ㅇ Deleting Windows Shadow Copies via shell execution
  ◦ vssadmin.exe Delete Shadows /All /Quiet
  ◦ WMIC.exe ‘shadowcopy delete’

ㅇ Stopping sensitive Windows services
  ◦ Wscsvc - Security Center service.
  ◦ WuauServ - Windows Update service.
  ◦ BITS - Background Intelligent Transfer Service (Silent signatures to Microsoft).
  ◦ WerSvc - Windows Error Reporting Service.
  ◦ EventSystem - COM+ Event System service.
  ◦ EventLog - Windows Event Log service.
Turning Defenses Off (DLL) – cont.

◊ Creates a Spartan icon the system

◊ Disables **UAC** (User Access Control).
  ◊ SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System\EnableLUA

◊ Creates boot persistency.
  ◊ SOFTWARE\Microsoft\Windows\CurrentVersion\Run\Microsoft Startup Service
Spartan Spear (DLL)

◊ Generates a unique computer ID.

◊ Connects to C&C for registration and for the server’s public RSA key.

◊ Maps all the File-System on the local host.
  ◊ C:\, D:\, Removable media.

◊ Creates an “Encryptor Object” per directory.
  ◊ i.e. a unique AES key per directory.

◊ The AES key which used to encrypt a file is stored encrypted at the start of the file with asymmetric RSA encryption.
Spartan Shield (DLL)

◊ Acts as a protection layer between the encryption unit and the SPARTAN Engine.

◊ Creates at least one dummy-dead father process between the Encryption process (which might be detected) to the SPARTAN Engine.

◊ Based on a probability heuristic –
  ◊ Injects a SPARTAN Spear DLL to a known process and exits.
  ◊ Re-inject itself to a known process and exits.
Honeypots Counter Measures

◊ In addition of shutting down the EventLog service, before any file encryption SPARTAN searches for Auditing information on the specific file.

◊ Elevating privileges to SE_SECURITY_NAME.

◊ Create a handle to the file with ACCESS_SYSTEM_SECURITY to retrieve information stored in the SACL (System – ACL).
  ◊ GetSecurityInfo() | SACL_SECURITY_INFORMATION

◊ Checks for file auditing made by the common principles as Administrator ¦ local user.
  ◊ Also called Trustees.
  ◊ GetAuditedPermissionsFromAcl();

◊ If needed, goes deeper into the File’s SACL and searches for any auditing ACE.
Spartan C&C Server

◊ A Multi-Threaded Console App running on Azure VM.

◊ Consists of 3 Modules.
  ◊ Network, Crypto & Database

◊ Connection Via TCP.
  ◊ No sensitive data is being transferred between client and C&C.

◊ Generates a unique RSA key-pair for each Spartan client.

◊ Manual operation for decryption requests.
Spartan Storage Server

- Based on Azure BLOB Storage server.
- Hosts the Spartan payload and files (background, icon, DLLs).
Bonus Summary

- Infection through a Word document.
- Detection avoidance & redundancy methods –
  - Modular structure.
  - Multi process.
  - DLL injections.
  - Using Fake names ("Microsoft Startup Service", Engine is "svchost.exe")
- Paralyzes Windows Defender.
- Disables Windows Restore, Windows Recovery, UAC.
- Deletes Shadow Copies.
- Disable Security Services (EventLog, Security Center, etc.).
Bonus Summary

- Unique AES key per directory.
- Honeypots counter-measures.
- Boot persistency
- Unique SPARTAN icon.

- USB infection – Copies Dropper* to newly inserted USB, looks like legitimate file (a users PDF). [Based on example from MSDN].

---

*Dropper – Downloads all the ransomware’s libraries and executes the “Engine”.
Tests and Analysis

🔹 Testing Environment:
   ◆ VM running Spartan.
   ◆ Azure Server
   ◆ Bare metal Client for USB infection testing.

🔹 Tests:
   ◆ Each module separately:
      ◆ Word macro, USB infection, encryption & decryption, turning off defenses, injection to processes, boot persistency, network modules, server key generation and network components, Server’s DB.

   ◆ Full Cycle:
      ◆ Executing Dropper ➔ Downloading files ➔ Turning defenses off and injecting to processes ➔ Encrypting ➔ Finish message ➔ Decryption.
Tests and Analysis

◇ Notable tests:

◇ Full cycle test (from a clean pc to a clean pc).
  ◇ With(out) Administrator privileges.
  ◇ With(out) reboot during the encryption.

◇ USB infection on bare metal pc (without payload).

◇ CyberReason test.

◇ Other projects testing.
Known Issues

◊ Administrator privileges required for a part of the features.

◊ Antivirus may alert.
  ◊ DLL injections, Word macro [known techniques].

◊ A logger\auditing based detector can find the relation between the Encryptors to the engine and kill the engine.
  ◊ TurnOffDefenses.dll is unprotected.

◊ Internet connection required for infection (downloading files) and key delivery.
Spartan V2

  ✧ All EXE\DLL files are stored encrypted on disk, and are decrypted only on RAM.
  ✧ Much harder detections as source executables are unknown to the AV.
  ✧ The capability was implemented and tested for Spartan V1, but wasn’t merged due to time limits.

✧ Anti-debugging and Anti-virtualization.

✧ Concurrency between Engine and Encryptors.
  ✧ Let the Engine more control over the Encryptors.

✧ Disable all windows notifications.

✧ Shield processes for TurnOffDefenses.dll
Thank You!