black hat®
ASIA 2021
May 6-7, 2021
ARSENAL

FalconEye: Windows Process Injection Techniques Catch Them All

Rajiv Kulkarni, Sushant Paithane, Rex Guo

Rajiv Kulkarni

Senior Staff Security Researcher @
Palo Alto Networks
(Wildfire Linux Analyzer)

Security Researcher @ Tetration
Analytics (Cisco Acquisition)

Senior Security Analyst @ Adobe
(Incident Response and Sec Eng)

https://www.linkedin.com/in/rajivkulkarni/

Sushant Paithane

Security Engineer @ Confluera
(Windows Security)

Engineering Manager @ FireEye
(MVX Sandbox Detection)

Staff Engineer @ FireEye
(MVX Sandbox Detection)

Rex Guo

Principle Security Engineer @ Confluera

Engineering Manager @ Tetration Analytics (Cisco acquisition)

Security Researcher @ Intel

Ph.D. @ NYU

https://www.linkedin.com/in/xiaofeiguo/

Agenda

- Motivation
- Classification of process injections
- FalconEye approach
- FalconEye architecture
- Demo 1: Reflective DLL
- Demo 2: Stateless Atom Bombing
- Demo 3: Stateful (PROPagate)
- Demo 4: Instrumentation Callback Injection

Process Injection

- Objectives of process injections
  - Defense evasion
  - Privilege escalation

- Red teams and APTs frequently uses process injections
  - DLL injection (39 threat actors, e.g., BlackEnergy, Emotet, Maze, Powersploit, Cobalt Strike, Poison Ivy, etc.)
  - Extra Window Memory Injection - SetWindowLongEx (threat actor: EPIC, power loader)
  - Asynchronous procedure call (Attor, Carberp, IcedID, InvisiMole, Pillowmint, TURNEDUP)
  - Thread Execution Hijacking (Gazer, Trojan.Karagany)
  - Atombombing (Dridex)

https://attack.mitre.org/techniques/T1055/

Why this Research?

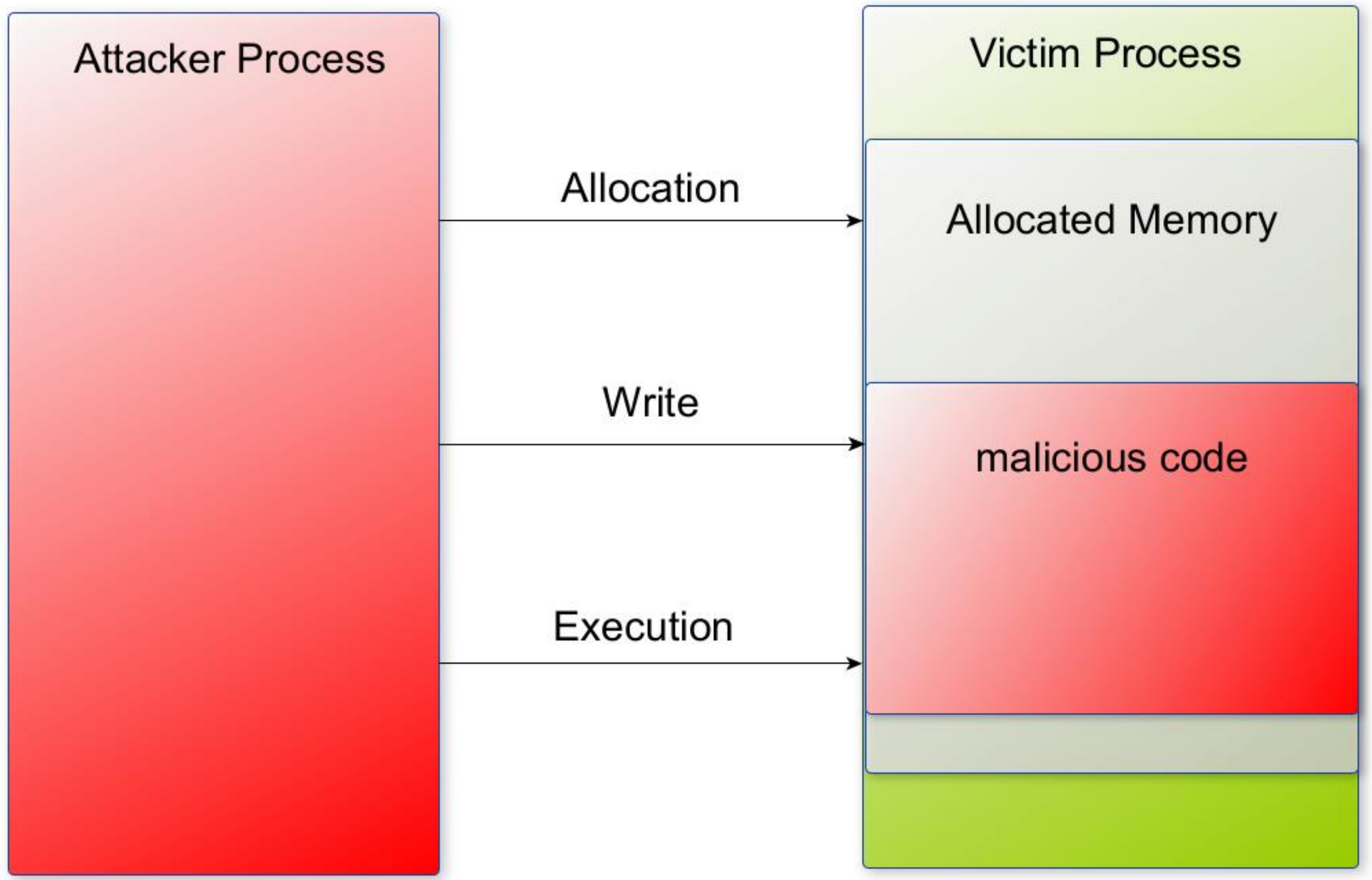
- Prior research
  - Pinjectra’s on process injection, etc.

- No comprehensive real-time detection techniques for the publicly known PIs
  - Memhunter (periodical scanning)
  - Volatility/rekall (memory forensics)

- Native windows capabilities are not sufficient for detecting PIs
  - Event Tracing for Windows (ETW)
  - Windows event log
  - Sysmon

https://github.com/SafeBreach-Labs/pinjectra

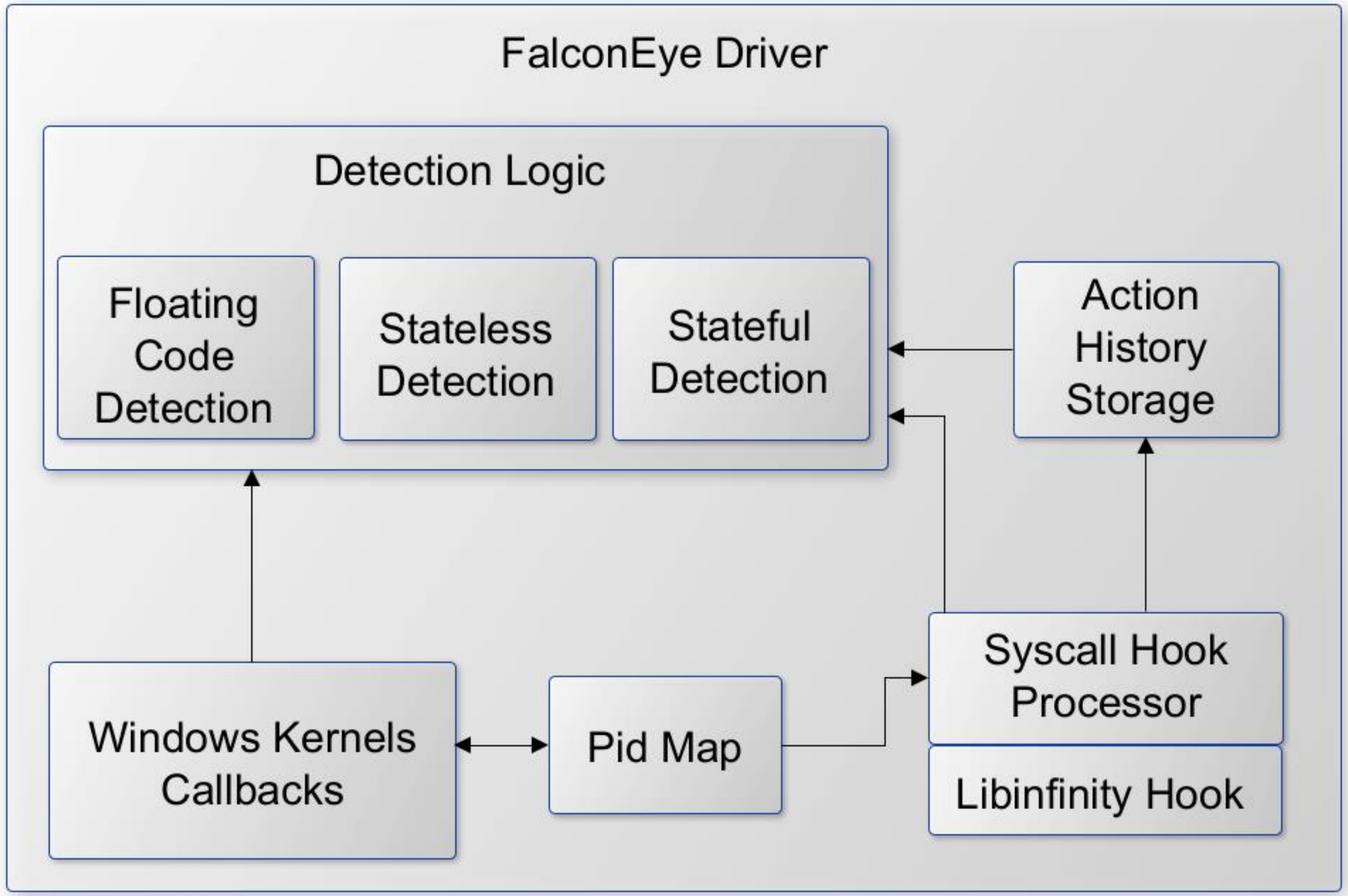
Common Process Injections Methodology



FalconEye - Approach

- Realtime detection as opposed to scan based triggers
  - Memhunter
  - Volatility
- Generic behavior detection
  - Overall attacker process behavior rather than sequence of syscalls
- Modular Implementation
  - Easily add detections for newly discovered techniques
- Low FP rate
  - Easily add detections for newly discovered techniques

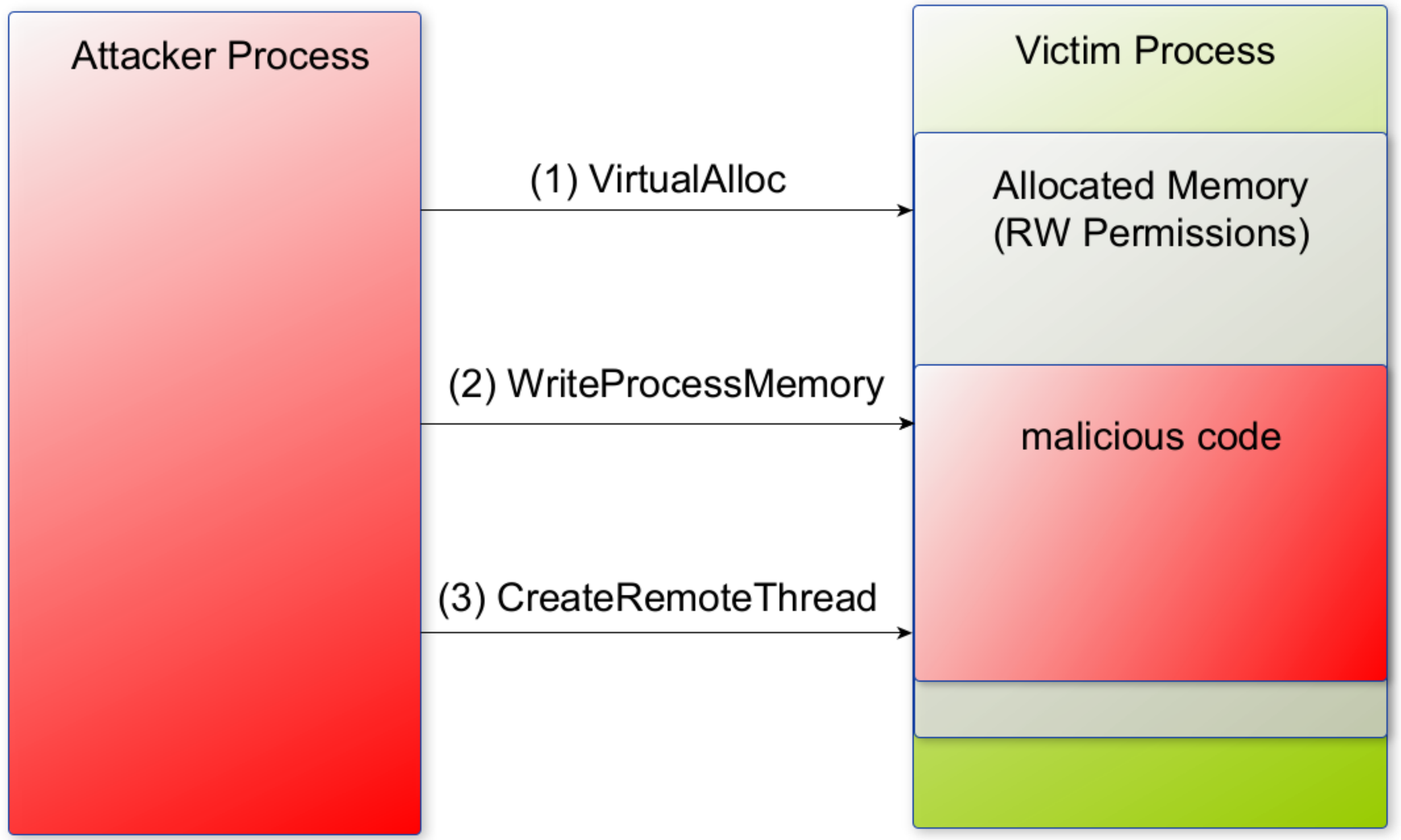
Software Architecture



https://github.com/everdox/InfinityHook

Demo 1: Reflective DLL Injection

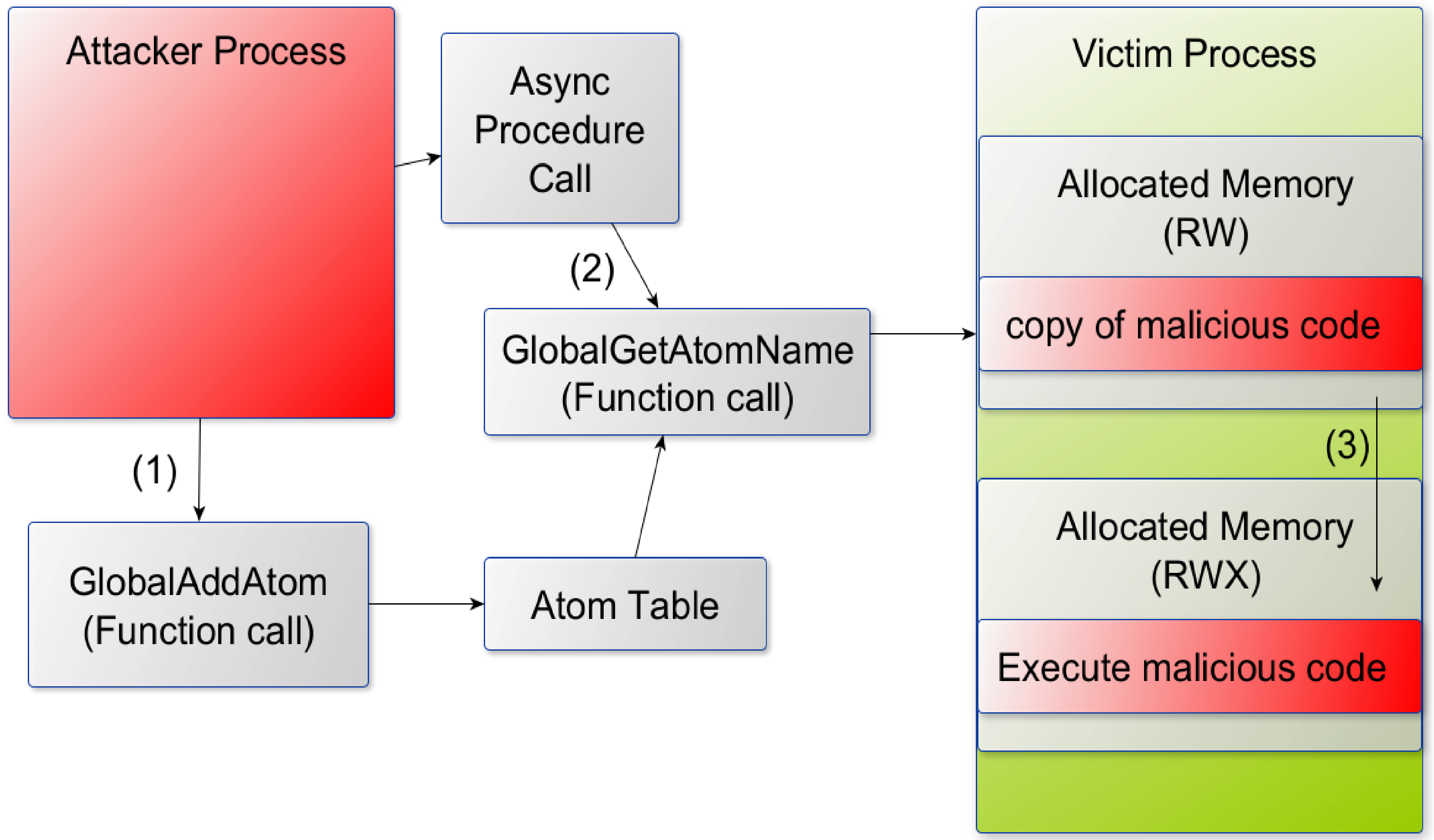
- It bypasses ImageLoad callback and inspecting loaded module list
- Still requires the attacker process to create a PE file in the victim process memory
- Finally the execution is triggered via CreateRemoteThread
- Detection: Using thread started in floating memory (not backed by binary image)



https://github.com/stephenfewer/ReflectiveDLLInjection

Demo 2: Atom Bombing

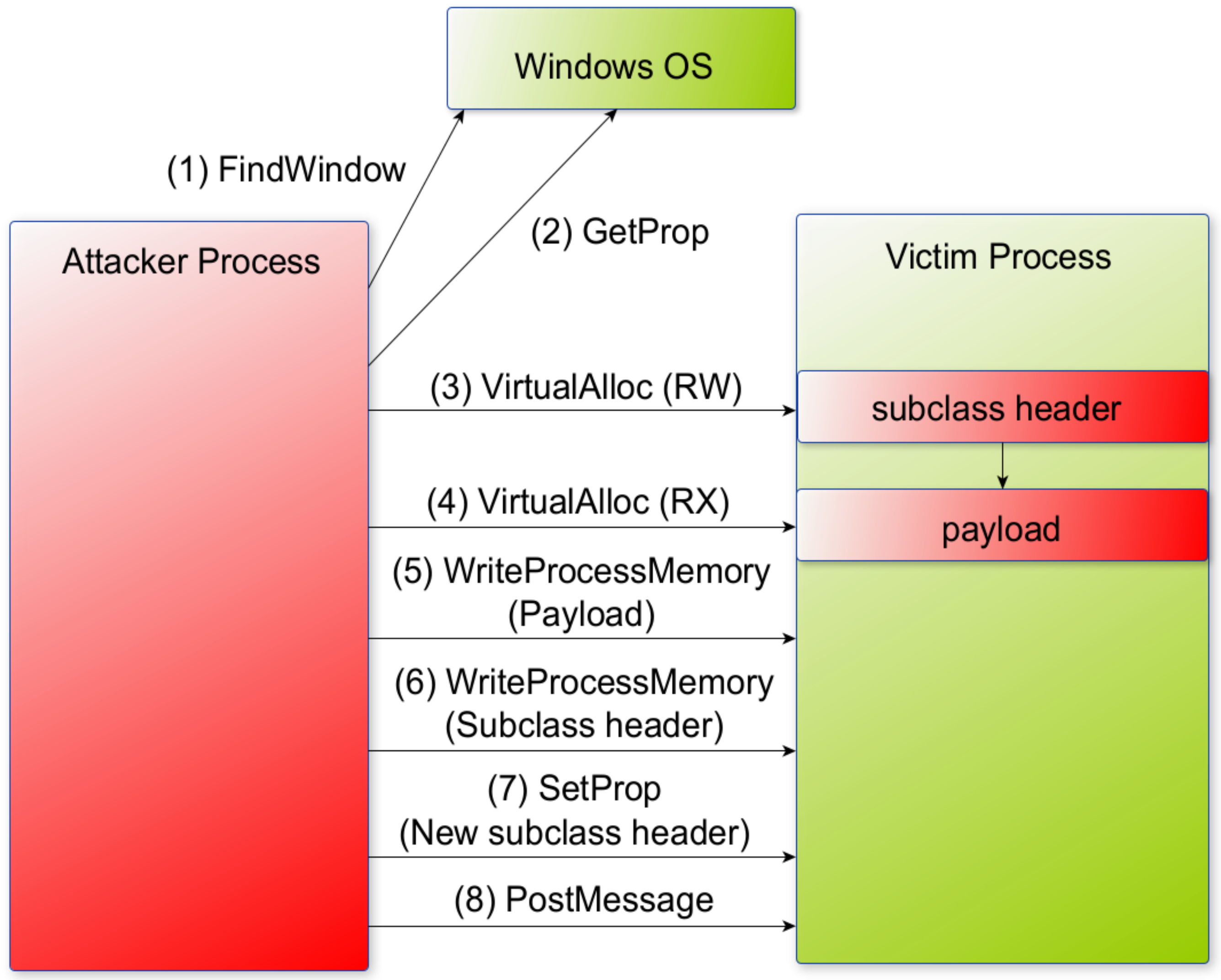
- Innovative technique that achieves Allocation and Write indirectly via Global Atoms
- Uses repeated QueueUserAPC to manipulate victim process memory
- Finally leverages a code cave to write ROP chain and triggers it via SetThreadContext
- Detection: Using ApcRoutine pointing to GetGlobalAtom



https://www.enisa.europa.eu/publications/info-notes/atombombing-2013-a-new-code-injection-attack

Demo 3: PROPagate

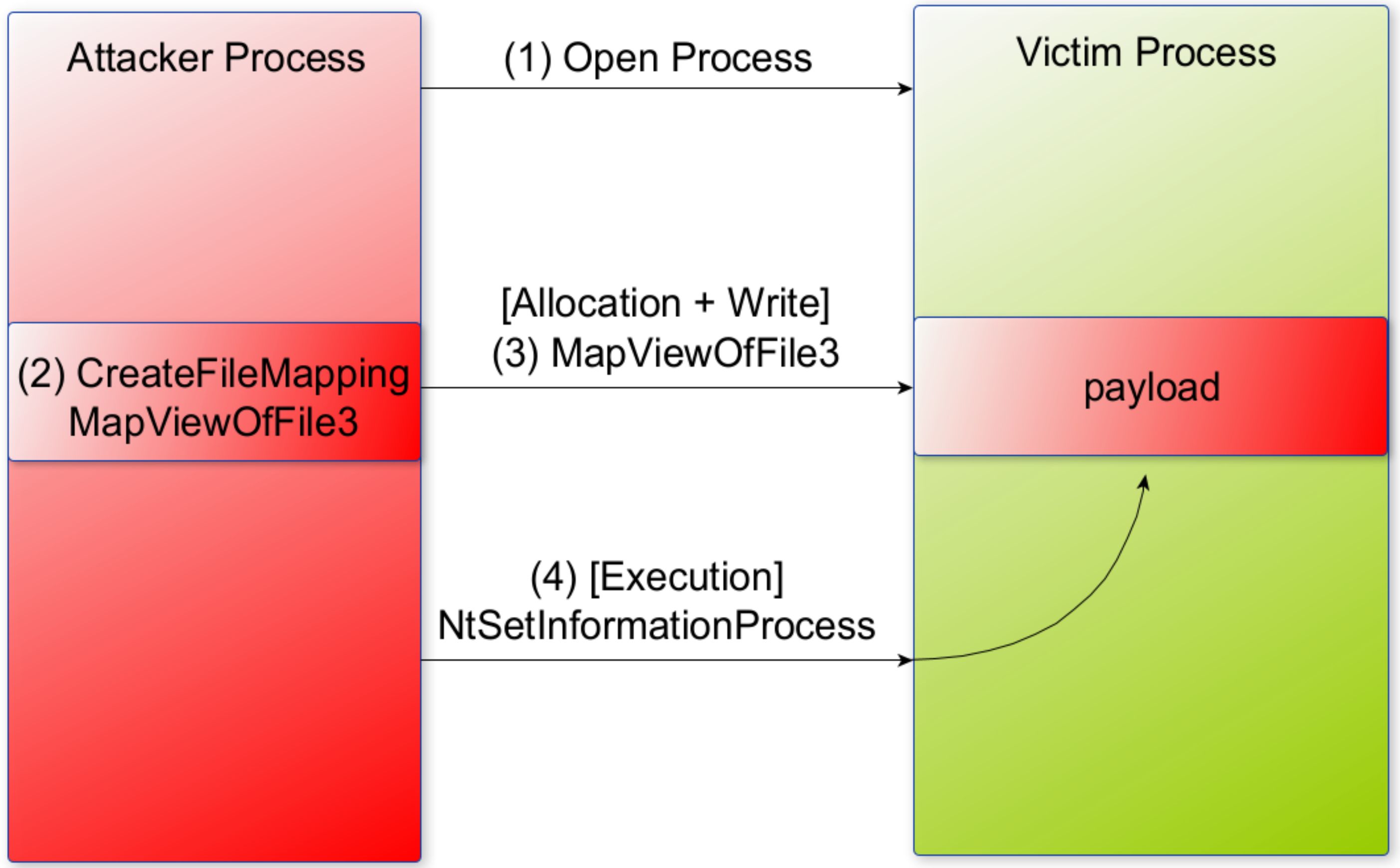
- Class of PI techniques that overwrite function pointers to point to a payload
- PROPagate overwrites the function pointer in a subclassed window via SetProp
- FalconEye stores WPM calls (Stateful)
- Detection: (1) triggered when FalconEye encounters SetProp family of functions.
- Detection: (2) stored WPM state is stitched. This enables the payload lookup



https://modexp.wordpress.com/2018/08/23/process-injection-propagate/

Demo 4: Instrumentation callback injection

- Relatively new technique. Mid-2020
- Got coverage without explicitly designing FalconEye for this technique
- Attacker needs to allocate memory in victim and point it to the callback. Then the malicious callback is launched as a new thread
- Detection: catches new thread execution and uses floating code detection



https://splintercod3.blogspot.com/p/weaponizing-mapping-injection-with.html

FalconEye – Future

- Leverage MS documented instrumentation as much as possible
- Refine the algorithm for new PI techniques in the future
- Optimize performance

https://github.com/rajiv2790/FalconEye