Effective, Scalable Threat Detection & Response

Andrew Case / @attrc

Volexity

VOLEXITY

My Background

- Open source tool development (Volatility, Registry Decoder)
- Co-Author “The Art of Memory Forensics”
- Led numerous insider threat investigations
- Perform incident response on large networks targeted by sophisticated threat actors

VOLEXITY

Traditional Detection & Response is Dead

- Traditional investigations only required analysis of a small number of systems
- Anti-forensics tools and techniques were easy to identify and work around
- Data sources were constant and straightforward to analyze

Modern Detection and Response

- Modern IR involves tens, hundreds, or thousands of systems
  - Spread across networks and the world
  - Involving multiple platforms (Windows, Linux, Mac) and devices (laptops, desktops, servers, mobile, "smart" everything)
- Modern attacker toolsets are built with scale and anti-forensics capabilities as main features

VOLEXITY

Issues on the Endpoint – Anti-Forensics

- AV and security agents look for disk-based artifacts and activity
  - Skilled attackers operate only in memory
- Responders run "Live IR" toolkits to look for signs of evil
  - Modern malware specifically tampers with the APIs that "Live IR" toolkits depend on
  - Responders and threat hunting teams see only what the malware wants them to see

Issues on the Endpoint – Scale

- Attackers have many avenues to scale
  - PowerShell
  - Group policy
  - Remotely scheduled tasks
  - Automated build/update/configuration servers
  - SSH automation (Linux/Mac)
- Defenders are on their own

PowerShell Empire [1]

Origins

Empire is a pure PowerShell post-exploitation agent built on cryptologically-secure communications and a flexible architecture. Empire implements the ability to run PowerShell agents without needing powershell.exe, rapidly deployable post-exploitation modules ranging from key loggers to Mimikatz, and adaptable communications to evade network detection, all wrapped up in a usability-focused framework. It premiered at BSidesLV in 2015.

Metasploit/Meterpreter [2]

Meterpreter is an advanced, dynamically extensible payload that uses *in-memory* DLL injection stagers and is extended over the network at runtime. It communicates over the stager socket and provides a comprehensive client-side Ruby API. It features command history, tab completion, channels, and more.

Issues on the Network

- Focused on detection of known malicious websites and domains
- Full network capture is generally not feasible or manageable on production networks
  - Too much data
  - After capture, still needs to be parsed and processed
  - Ubiquitous encryption

VOLEXITY

Detection and Response Questions

- Which users visited malicious-website[.]com in the last hour?
  - Day?
  - Week?
  - Month?
  - Year?
- Which users resolved bad-domain[.]org?
- Which users downloaded Flash-Exploit[.]swf?

VOLEXITY

More Questions

- Which internal systems did an infected system connect to?
- Which systems did Bob's compromised account access?
  - File shares? Internal web applications? Vendor portals?
- Malware[.]exe persists through a registry key, which other systems also have that key present?
- A keylogger records keystrokes to C:\operations.log and hides the file from the live system - can we still find it?

Effective Detection & Response Requirements

- Visibility
  - Ability to monitor actions across entire environment
- Historical data
  - Investigators can't incorporate what isn't there
- Acquisition & analysis processes that defeat anti-forensics

Endpoint Requirements – Event Logging

- Minimum:
  - Logins/logoffs
  - User account and groups creation/deletion/modification
- Desirable:
  - Process creation/termination
  - Use of sensitive privileges
- Must be centralized
  - Avoids anti-forensics clearing of logs
  - Allows for historical querying and hunting of suspicious activity

VOLEXITY

Endpoint Requirements – Memory Forensics [4]

- Direct examination of RAM, not output of live APIs
  - Not susceptible to traditional malware interference
  - MUCH more information available
- Scalability
  - No disk images
  - Can focus only on portions of memory with needed data structures
- Find system state anomalies, not malware-specific patterns
- Memory samples also contain relevant disk artifacts

Endpoint-Based Detection

- Automating event log analysis
  - Why did Bob in accounting log into the domain controller?
  - Why was a network admin logging into systems at midnight?
  - Why did Jane in HR suddenly have domain admin privileges?
- Gathering data artifacts through memory forensics
  - Map your existing “Live IR” artifacts in memory artifacts
  - Speed + avoid anti-forensics

VOLEXITY

Endpoint-Based Response

- Leveraging event logs
  - Review all logs around relevant timeframe
  - Which user accounts involved?
  - Which tools were run?
- Leveraging memory forensics
  - Find on-disk & memory-only malware
  - Find PowerShell and cmd.exe activity missed by agents and disk forensics
  - Find historical information no longer tracked by APIs

VOLEXITY

Scaling Endpoint-Based Response

- Focus the IR workflow to where we know the attackers were active
- While attackers were active:
  - Which systems were compromised accounts used on?
  - Do any systems have suspicious account creation?
- Use IOCs found through memory forensics to quickly sweep the network for other infected hosts

Threat Hunting on the Endpoint

- Review of event logs easily directs threat hunting
- Deep memory forensics of critical systems is a necessity to combat advanced threats
- Not convinced of the value of threat hunting?
  - Watch [3]

VOLEXITY

Network Detection and Response Requirements

- Passive DNS
  - Record all replies and requests sent through network
- HTTP request and responses
- Netflow
- Full PCAP is usually not feasible and/or desired

VOLEXITY

Network-Based Detection

- Leveraging Passive DNS
  - Resolution of newly registered domains
  - Resolution of domains from dynamic DNS providers
- Leveraging HTTP
  - Unusual user agents
  - Direct file downloads
- Leveraging Netflow
  - Why did Bob in accounting connect to a file server on the other side of the world?

VOLEXITY

Network-Based Response

- Bob's system was infected with malware from abc[.]xyz
- Which other systems resolved this domain?
- What are all the IP addresses that this domain ever resolved to?
  - Which other hosts contacted these IP addresses?
- Which internal and external resources did Bob's system contact after being infected?

Threat Hunting on the Network

- Reapply detection stage examples
- Perform deep review of historical data
- Leverage statistical analysis:
  - Of 5,000 employees, which domains were resolved by less than 1%?
  - Of 5,000 employees, why did only 1 talk to a particular IP address

Effective Detection & Response Requirements

- Visibility
  - We have full visibility of endpoints and the network
- Historical data
  - Event logs + capture of most relevant network data
- Acquisition & analysis processes that defeat anti-forensics
  - Accomplished through memory forensics and centralized logging

Scalable & Effective Results

- Ongoing threat detection that actually finds threats
- Once detected, highly focused incident response can begin
- Containment and remediation can focus on systems that were truly affected without missing attacker activity

VOLEXITY

Tools of the Trade

- Log aggregation / analysis - graylog
  - https://www.graylog.org/
- Memory forensics – Volatility
  - https://github.com/volatilityfoundation/volatility
- Network capture – Bro or Suricata
  - https://www.bro.org/ | | https://suricata-ids.org/
- The Almighty ELK Stack
  - http://www.jeeatwork.com/?p=224

VOLEXITY

Questions/Comments?

- Contact Information:
  - andrew@dfir.org (0xB2446B45)
  - @attrc

- References

1. https://www.powershellempire.com/
2. https://www.offensive-security.com/metasploit-unleashed/about-meterpreter/
3. http://bit.ly/2ek5MsN
4. http://2014.video.sector.ca/video/110388398

VOLEXITY