MITRE | ATT&CK™ EVALUATIONS

APT29 / Cozy Bear / The Dukes Emulation Plan
Last Updated: January 2021

Disclaimer

The views, opinions and/or findings contained in this report are those of The MITRE Corporation and should not be construed as an official government position, policy, or decision, unless designated by other documentation. The emulation content is only to be executed with appropriate, prior, and explicit authorization for the purposes of assessing security posture and/or research.

Purpose

To advance the practice of offensive security testing through threat-informed operations, we present this adversary emulation plan of APT29/The Dukes/Cozy Bear/YTTRIUM (hereinafter referred to as just APT29). This documentation is supported by publicly available reporting and/or contributions via our open call to share cyber threat intelligence. This emulation plan includes descriptions of APT29 tradecraft as well as links to resources developed to execute an emulation of those adversary behaviors. To ground the content of this emulation plan in a common taxonomy, it is based on the MITRE ATT&CK™ model (https://attack.mitre.org).

This emulation plan can be utilized by anyone wanting to execute an offensive security assessment "in the spirit of" known APT29 TTPs, as well as a reference to the broader community of researchers, analysts, and even defenders seeking to understand documented APT29 tradecraft.

Acknowledgements

We would like to formally thank the people that contributed to the content, review, and format of this document. This includes the MITRE ATT&CK and MITRE ATT&CK Evaluations teams, the organizations and people that provided public intelligence and resources, as well as the following organizations that participated in the community cyber threat intelligence contribution process:

- Kaspersky
- Microsoft
- SentinelOne

Adversary Overview

APT29 is threat group that has been attributed to the Russian government and has operated since at least 2008. [1] [14] This group has been attributed to major breaches targeting U.S. governments/organizations such as the Democratic National Committee, as well as various international ministries and agencies. [15] [16] APT29 has also been known to "cast a wide net" in terms of targeting, seemingly making this group a universal threat.

In terms of operational tradecraft, APT29 is distinguished by their commitment to stealth and sophisticated implementations of techniques via an arsenal of custom malware. APT29 typically accomplishes goals via custom compiled binaries and alternate (at least at the time) execution methods such as PowerShell and WMI. APT29 has also been known to employ various operational cadences (smash-and-grab vs. slow-and-deliberate) depending on the perceived intelligence value and/or infection method of victims.

Emulation Plan Structure

This emulation plan was derived from the methodology used to execute Round 2 of ATT&CK Evaluations (https://attackevals.mitre.org/), which aims to evaluate cybersecurity products using an open methodology based on our ATT&CK framework.

APT29 has been known to conduct large-scale, simultaneous, targeted phishing campaigns and broader less targeted campaigns using TOR (The Onion Router) exit nodes and Torrents to get initial access. [1] In their broader campaigns, APT29 has conducted smash-and-grab espionage with rapid collection and exfiltration. If the victim appeared to be of value, they switched to a different toolset and stealthier tactics for a persistent, long-term intelligence-gathering campaign. [1]

In their smaller more targeted campaigns, APT29 has utilized a different toolset incrementally modified to attempt to evade published intelligence about their operations. [1] The Evaluations methodology, as well as this emulation plan, are split into two distinct scenarios (Day 1 and Day 2) to reflect these differing operational cadences (*see graphic to right*).

For more information about the methodology used to execute Round 2 of ATT&CK Evaluations, please visit https://attackevals.mitre.org/adversary-emulation.html.
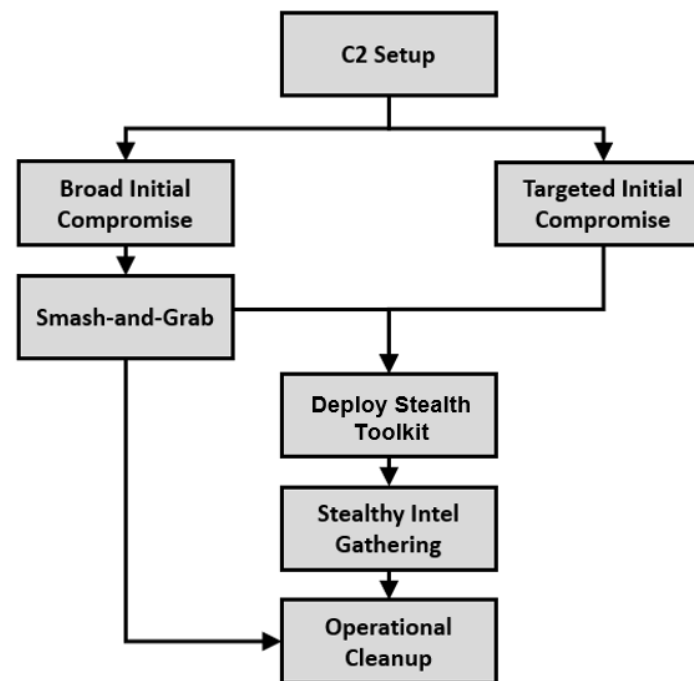
APT29 Operational Cadences
(Day 1 along the left, Day 2 along the right)

Day 1

Overview

The narrative we created for our Day 1 scenario [based on CosmicDuke (ATT&CK S0050), MiniDuke (ATT&CK S0051), SeaDuke/SeaDaddy (ATT&CK S0053), CozyDuke/CozyCar (ATT&CK S0046), and HAMMERTOSS (ATT&CK S0037)] begins with a legitimate user clicking on a malicious payload delivered via a "spray and pray" broad spearphishing campaign. The attacker immediately kicks off a "smash-and-grab" rapid espionage mission, gathering and exfiltrating data. After initial exfiltration, the attacker realizes the value of victim and subsequently deploys a stealthier toolkit, changing TTPs and eventually moving laterally through the rest of the environment. The scenario ends with the execution of previously established persistence mechanisms are executed.

This content to execute this scenario was tested and developed using Pupy (https://github.com/n1nj4sec/pupy), Meterpreter (https://github.com/rapid7/metasploit-framework), and other custom/modified scripts and payloads. Pupy and Meterpreter were chosen based on their available functionality and similarities to the adversary's malware within the context of this scenario, but alternative red team tooling could be used to accurately execute these and other APT29 behaviors. More information, including the required resources, setup instructions, and step by step instructions on how to execute the Day 1 scenario, is available at https://github.com/mitre-attack/attack-arsenal.

Scope

The in-scope techniques for the Day 1 scenario are displayed to the right:

Initial Access (11 items)	Execution (34 items)	Persistence (62 items)	Privilege Escalation (32 items)	Defense Evasion (69 items)	Credential Access (21 items)	Discovery (23 items)	Lateral Movement (18 items)	Collection (13 items)	Command And Control (16 items)	Exfiltration (9 items)	Impact (16 items)
Drive-by Compromise	AppleScript	.bash_profile and .bashrc	Access Token Manipulation	Access Token Manipulation	Account Manipulation	Account Discovery	AppleScript	Audio Capture	Commonly Used Port	Automated Exfiltration	Account Access Removal
Exploit Public-Facing Application	CMSTP	Accessibility Features	Accessibility Features	Binary Padding	Bash History	Application Window Discovery	Application Deployment Software	Automated Collection	Communication Through Removable Media	Data Compressed	Data Destruction
External Remote Services	Command-Line Interface	Account Manipulation	AppCert DLLs	BITS Jobs	Brute Force	Browser Bookmark Discovery	Component Object Model and Distributed COM	Clipboard Data	Connection Proxy	Data Encrypted	Data Encrypted for Impact
Hardware Additions	Compiled HTML File	AppCert DLLs	AppInit DLLs	Bypass User Account Control	Credential Dumping	Domain Trust Discovery	Exploitation of Remote Services	Data from Information Repositories	Custom Command and Control Protocol	Data Transfer Size Limits	Defacement
Replication Through Removable Media	Component Object Model and Distributed COM	AppInit DLLs	Application Shimming	Clear Command History	Credentials in Files	File and Directory Discovery	Internal Spearphishing	Data from Local System	Custom Cryptographic Protocol	Exfiltration Over Alternative Protocol	Disk Content Wipe
Spearphishing Attachment	Control Panel Items	Application Shimming	Bypass User Account Control	CMSTP	Credentials in Registry	Network Service Scanning	Logon Scripts	Data from Network Shared Drive	Data Encoding	Exfiltration Over Command and Control Channel	Disk Structure Wipe
Spearphishing Link	Dynamic Data Exchange	Authentication Package	DLL Search Order Hijacking	Code Signing	Exploitation for Credential Access	Network Share Discovery	Pass the Hash	Data from Removable Media	Data Obfuscation	Exfiltration Over Other Network Medium	Endpoint Denial of Service
Spearphishing via Service	Execution through API	BITS Jobs	Dylib Hijacking	Compile After Delivery	Forced Authentication	Network Sniffing	Pass the Ticket	Data Staged	Domain Fronting	Exfiltration Over Physical Medium	Firmware Corruption
Supply Chain Compromise	Execution through Module Load	Bootkit	Elevated Execution with Prompt	Compiled HTML File	Hooking	Password Policy Discovery	Remote Desktop Protocol	Email Collection	Domain Generation Algorithms	Scheduled Transfer	Inhibit System Recovery
Trusted Relationship	Exploitation for Client Execution	Browser Extensions	Emond	Component Firmware	Input Capture	Peripheral Device Discovery	Remote File Copy	Input Capture	Fallback Channels		Network Denial of Service
Valid Accounts	Graphical User Interface	Change Default File Association	Exploitation for Privilege Escalation	Component Object Model Hijacking	Input Prompt	Permission Groups Discovery	Remote Services	Man in the Browser	Multi-hop Proxy		Resource Hijacking
	InstallUtil	Component Firmware	Extra Window Memory Injection	Connection Proxy	Kerberoasting	Process Discovery	Replication Through Removable Media	Screen Capture	Multi-Stage Channels		Runtime Data Manipulation
	Launchctl	Component Object Model Hijacking	File System Permissions Weakness	Control Panel Items	Keychain	Query Registry	Shared Webroot	Video Capture	Multiband Communication		Service Stop
	Local Job Scheduling	Create Account	Hooking	DCShadow	LLMNR/NBT-NS Poisoning and Relay	Remote System Discovery	SSH Hijacking		Multilayer Encryption		Stored Data Manipulation
	LSASS Driver	DLL Search Order Hijacking	Image File Execution Options Injection	Deobfuscate/Decode Files or Information	Network Sniffing	Security Software Discovery	Taint Shared Content		Port Knocking		System Shutdown/Reboot
	Mshta	Dylib Hijacking	Launch Daemon	Disabling Security Tools	Password Filter DLL	Software Discovery	Third-party Software		Remote Access Tools		Transmitted Data Manipulation
	PowerShell	Emond	New Service	DLL Search Order Hijacking	Private Keys	System Information Discovery	Windows Admin Shares		Remote File Copy		
	Regsvcs/Regasm	External Remote Services	Parent PID Spoofing	DLL Side-Loading	Securityd Memory	System Network Configuration Discovery	Windows Remote Management		Standard Application Layer Protocol		
	Regsvr32	File System Permissions Weakness	Path Interception	Disabling Security Tools	Steal Web Session Cookie	System Network Connections Discovery			Standard Cryptographic Protocol		
	Rundll32	Hidden Files and Directories	Plist Modification	Execution Guardrails	Two-Factor Authentication Interception	System Owner/User Discovery			Standard Non-Application Layer Protocol		
	Scheduled Task	Hooking	Port Monitors	Exploitation for Defense Evasion		System Service Discovery			Uncommonly Used Port		
	Scripting	Hypervisor	PowerShell Profile	Extra Window Memory Injection		System Time Discovery			Web Service		
	Service Execution	Image File Execution Options Injection	Process Injection	File and Directory Permissions Modification		Virtualization/Sandbox Evasion					
	Signed Binary Proxy Execution	Kernel Modules and Extensions	Scheduled Task	File Deletion							
	Signed Script Proxy Execution	Launch Agent	Service Registry Permissions Weakness	File System Logical Offsets							
	Source	Launch Daemon	Setuid and Setgid	Gatekeeper Bypass							
	Space after Filename	Launchctl	SID-History Injection	Group Policy Modification							
	Third-party Software	LC_LOAD_DYLIB Addition	Startup Items	Hidden Files and Directories							
	Trap	Local Job Scheduling	Sudo	Hidden Users							
	Trusted Developer Utilities	Login Item	Sudo Caching	Hidden Window							
	User Execution	Logon Scripts	Valid Accounts	HISTCONTROL							
	Windows Management Instrumentation	LSASS Driver	Web Shell	Image File Execution Options Injection							
	Windows Remote Management	Modify Existing Service		Indicator Blocking							
	XSL Script Processing	Netsh Helper DLL		Indicator Removal from Tools							
		New Service		Indicator Removal on Host							
		Office Application Startup		Indirect Command Execution							
		Path Interception		Install Root Certificate							
		Plist Modification		InstallUtil							
		Port Knocking		Launchctl							
		Port Monitors		LC_MAIN Hijacking							
		PowerShell Profile		Masquerading							
		Rc.common		Modify Registry							
		Re-opened Applications		Mshta							
		Redundant Access		Network Share Connection Removal							
		Registry Run Keys / Startup Folder		NTFS File Attributes							
		Scheduled Task		Obfuscated Files or Information							
		Screensaver		Parent PID Spoofing							
		Security Support Provider		Plist Modification							
		Server Software Component		Port Knocking							
		Service Registry Permissions Weakness		Process Doppelgänging							
		Setuid and Setgid		Process Hollowing							
		Shortcut Modification		Process Injection							
		SIP and Trust Provider Hijacking		Redundant Access							
		Startup Items		Regsvcs/Regasm							
		System Firmware		Regsvr32							
		Systemd Service		Rootkit							
		Time Providers		Rundll32							
		Trap		Scripting							
		Valid Accounts		Signed Binary Proxy Execution							
		Web Shell		Signed Script Proxy Execution							
		Windows Management Instrumentation Event Subscription		SIP and Trust Provider Hijacking							
		Winlogon Helper DLL		Software Packing							
				Space after Filename							
				Template Injection							
				Timestomp							
				Trusted Developer Utilities							
				Valid Accounts							
				Virtualization/Sandbox Evasion							
				Web Service							
				XSL Script Processing							

Breakdown

Step	High Level Overview of Emulation and Techniques Evaluated	Cited Intelligence	Open Invitation Contributor(s)	Emulation Content
1	The scenario begins with an initial breach, where a legitimate user clicks **(T1204)** an executable payload (screensaver executable) masquerading as a benign word document **(T1036)**. Once executed, the payload creates a C2 connection over port 1234 **(T1065)** using the RC4 cryptographic cipher . The attacker then uses the active C2 connection to spawn interactive cmd.exe **(T1059)** and powershell.exe **(T1086)** shells.	CosmicDuke's infection payloads have started by tricking victims into opening a Windows executable whose filename is manipulated to look like an image file using the Right-to-Left Override (RLO) feature. CosmicDuke has also used RC4 to decrypt incoming data and encrypt outgoing data.[2] SeaDuke and CozyDuke have used the RC4 cipher to encrypt data.[4] [7] [13] [16] CozyDuke can be used to spawn a command line shell. [16]	Kaspersky	The Day 1 README.md file describes how to either use the precompiled cod.3aka3.scr or generate a custom payload (via payload_configs.md), as well as additional commands to complete the step.
2	The attacker runs a one-liner command to search for filesystem for document and media files **(T1083, T1119)**, collecting **(T1005)** and compressing **(T1002)** content into a single file **(T1074)**. The file is then exfiltrated over the existing C2 connection **(T1041)**.	CosmicDuke's information stealing functionality included stealing user files with file extensions that match a predefined list.[1] [2]	Kaspersky	The Day 1 README.md file contains the commands to complete the step.
3	The attacker now uploads a new payload **(T1105)** to the victim. The payload is a legitimately formed image file with a concealed PowerShell script **(T1027)**. The attacker then elevates privileges via a user account control (UAC) bypass **(T1122, T1088)**, which executes the newly added payload. A new C2 connection is established over port 443 **(T1043)** using the HTTPS protocol **(T1071, T1032)**. Finally, the attacker removes artifacts of the privilege escalation from the Registry **(T1112)**.	CosmicDuke has occasionally embedded other malware components that are written to disk and executed. [1] MiniDuke has transferred additional backdoors onto a system via GIF files.[3] SeaDaddy/SeaDuke may support HTTPS/SSL network communications. [4] [13] APT29 has removed tools and forensic artifacts to hide activity, including the usage of Sdelete (ATT&CK S0195). APT29 has also bypassed UAC to elevate privileges.[5] HAMMERTOSS has embedded pictures with commands using steganography.[6]	Kaspersky Microsoft	The Day 1 README.md file describes how to either use the prebuilt monkey.png or generate a custom payload (via payload_configs.md), as well as additional commands to complete the step.

Step	High Level Overview of Emulation and Techniques Evaluated	Cited Intelligence	Open Invitation Contributor(s)	Emulation Content
4	The attacker uploads additional tools (T1105) through the new, elevated access before spawning an interactive powershell.exe shell (T1086). The additional tools are decompressed (T1140) and positioned on the target for usage. The attacker then enumerates running processes (T1057) to discover/terminate the initial access from Step 1 before deleting various files (T1107) associated with that access. Finally, the attacker launches a PowerShell script that performs a wide variety of reconnaissance commands (T1083, T1033, T1082, T1016, T1057, T1063, T1069), some of which are done by accessing the Windows API (T1106).	CozyDuke has been instructed to download and execute other executables, which in some cases included common hacking tools such as PSExec (ATT&CK S0029). [1] MiniDuke can download and execute new malware and lateral movement tools. [3] APT29 has removed tools and forensic artifacts to hide activity. [5] [7] [13] CozyDuke can be used to spawn a command line shell. [16]	Microsoft Kaspersky SentinelOne	The Day 1 README.md file contains the commands to complete the step, including executing the Invoke-Discovery function within readme.txt.
5	The attacker establishes two distinct means of persistent access to the victim by creating a new service (T1050) and creating a malicious payload in the Windows Startup folder (T1060).	CosmicDuke has installed a Windows service to achieve persistence on a system. [2] SeaDuke has the ability to persist using a .lnk file stored in the Startup directory. [4] APT29 has used several persistence mechanisms, including .LNK files. [5]	Kaspersky	The Day 1 README.md file describes how to generate custom hostui.exe and javamtsup.exe payloads (via payload_configs.md), as well as additional commands to complete the step, including executing the Invoke-Persistence function within readme.txt.
6	The attacker accesses credentials stored in a local web browser (T1081, T1003) using a tool renamed to masquerade as a legitimate utility (T1036). The attacker then harvests private keys (T1145) and password hashes (T1003).	CosmicDuke's information stealing functionality has included exporting user's cryptographic certificates, including private keys, and collecting user credentials, including passwords from web browsers (ex: Google Chrome). CozyDuke has contained modules that can steal NTLM hashes as well as capture screenshots. [1] [2]	Kaspersky SentinelOne	The Day 1 README.md file contains the commands to complete the step, including executing the Get-PrivateKeys function within readme.txt.
7	The attacker collects screenshots (T1113), data from the user's clipboard (T1115), and keystrokes (T1056). The attacker then collects files (T1005), which are compressed (T1002) and encrypted (T1022), before being exfiltrated to an attacker-controlled WebDAV share (T1048).	CosmicDuke's information stealing functionality has included keylogging, taking screenshots, and stealing clipboard contents. Collected data can be exfiltrated using WebDAV. [1] [2] CozyDuke can be used to take screenshots of a full desktop window and encrypt collected data. [16]	Kaspersky	The Day 1 README.md file contains the commands to complete the step, including executing the Invoke-ScreenCapture, Get-Clipboard, Get-Keystrokes, and Invoke-Exfil functions within psversion.txt.

Step	High Level Overview of Emulation and Techniques Evaluated	Cited Intelligence	Open Invitation Contributor(s)	Emulation Content
8	The attacker uses Lightweight Directory Access Protocol (LDAP) queries to enumerate other hosts in the domain **(T1018)** before creating a remote PowerShell session to a secondary victim **(T1028)**. Through this connection, the attacker enumerates running processes **(T1057)**. Next, the attacker uploads a new UPX-packed payload **(T1045, T1105)** to the secondary victim. This new payload is executed on the secondary victim via the PSExec utility **(T1077, T1035)** using the previously stolen credentials **(T1078)**.	SeaDuke has been written in Python and has been delivered through the CozyDuke toolkit. [1] [13] SeaDuke/SeaDaddy samples have been UPX-packed. [4] [5] [12] APT29 has UPX-packed and used SMB to transfer files. [5] APT29 has used UPX-packed, Python-compiled backdoors. [7]	Microsoft SentinelOne	The Day 1 README.md file describes how to generate a custom python.exe payload (via payload_configs.md), as well as additional commands to complete the step, including executing the Ad-Search and Invoke-SeaDukeStage functions within psversion.txt.
9	The attacker uploads additional utilities to the secondary victim **(T1105)** before running a PowerShell one-liner command **(T1086)** to search for filesystem for document and media files **(T1083, T1119)**. Files of interested are collected **(T1005)** then encrypted **(T1022)** and compressed **(T1002)** into a single file **(T1074)**. The file this then exfiltrated over the existing C2 connection **(T1041)**. Finally, the attacker deletes various files **(T1107)** associated with that access.	CosmicDuke's information stealing functionality has included stealing user files with file extensions that match a predefined list and exfiltrating collected data via HTTPS. SeaDuke can execute command such as uploading and downloading files. [1] [2] MiniDuke can download and execute new malware and lateral movement tools. [3] SeaDuke has contained commands to download and Base-64-encode files. [4] APT29 has removed tools and forensic artifacts to hide activity, including the usage of Sdelete (ATT&CK S0195). [5] [7] [13] SeaDaddy has used RAR to archive collected data. [7] CozyDuke can be used to take screenshots of a full desktop window and encrypt collected data. [16]	Kaspersky Microsoft SentinelOne	The Day 1 README.md file contains the commands to complete the step.

| 10 | The original victim is rebooted and the legitimate user logs in, emulating ordinary usage and a passage of time. This activity triggers the previously established persistence mechanisms, namely the execution of the new service **(T1035)** and payload in the Windows Startup folder **(T1060)**. The payload in the Startup folder executes a follow-on payload using a stolen token **(T1106, T1134)**. | CosmicDuke has installed persistence services that duplicate and uses the process token of explorer.exe to start the malware. [2] | Kaspersky | The Day 1 README.md file contains the commands to complete the step. |

Day 2

Overview

The narrative we created for our Day 2 scenario [based on PowerDuke (ATT&CK S0139), POSHSPY (ATT&CK S0150), CloudDuke (ATT&CK S0054), and more recent (2016+) TTPs)] begins with a legitimate user clicking on a malicious payload delivered via a targeted spearphishing campaign. The attacker employs a low and slow, methodical approach to owning the initial target, establishing persistence, gathering credential materials, then finally enumerating and owning the entire domain. Data exfiltration is dumped to a public cloud storage. The scenario ends with a simulated time-lapse where previously established persistence mechanisms are executed.

This content to execute this scenario was tested and developed using PoshC2 (https://github.com/nettitude/PoshC2) and other custom/modified scripts and payloads. PoshC2 was chosen based on its available functionality and similarities to the adversary's malware within the context of this scenario, but alternative red team tooling could be used to accurately execute these and other APT29 behaviors. More information, including the required resources, setup instructions, and step by step instructions on how to execute the Day 2 scenario, is available at https://github.com/mitre-attack/attack-arsenal.

Scope

The in-scope techniques for the Day 2 scenario are displayed to the right:

Initial Access (11 items)	Execution (34 items)	Persistence (62 items)	Privilege Escalation (32 items)	Defense Evasion (69 items)	Credential Access (21 items)	Discovery (23 items)	Lateral Movement (18 items)	Collection (13 items)	Command And Control (13 items)	Exfiltration (9 items)	Impact (16 items)
Drive-by Compromise	AppleScript	.bash_profile and .bashrc	Access Token Manipulation	Access Token Manipulation	Account Manipulation	Account Discovery	AppleScript	Audio Capture	Commonly Used Port	Automated Exfiltration	Account Access Removal
Exploit Public-Facing	CMSTP	Accessibility Features	Accessibility Features	Binary Padding	Bash History	Application Window	Application Deployment	Automated Collection	Communication Through	Data Compressed	Data Destruction
External Remote Services	Command-Line Interface	Account Manipulation	Account Manipulation	BITS Jobs	Brute Force	Browser Bookmark	Component Object Model	Clipboard Data	Connection Proxy	Data Encrypted	Data Encrypted for Impact
Hardware Additions	Compiled HTML File	AppCert DLLs	AppCert DLLs	Bypass User Account	Credential Dumping	Domain Trust Discovery	Exploitation of Remote	Data from Information	Custom Command and	Data Transfer Size Limits	Defacement
Replication Through	Component Object Model	AppInit DLLs	AppInit DLLs	Bypass User Account	Credentials from Web	File and Directory	Internal Spearphishing	Data from Local System	Custom Cryptographic	Exfiltration Over Alternative	Disk Content Wipe
Spearphishing Attachment	Control Panel Items	Application Shimming	Application Shimming	Clear Command History	Credentials in Files	Network Service Scanning	Logon Scripts	Data from Network Shared	Data Encoding	Exfiltration Over Command	Disk Structure Wipe
Spearphishing Link	Dynamic Data Exchange	Authentication Package	Bypass User Account	Code Signing	Credentials in Registry	Network Share Discovery	Pass the Hash	Data from Removable	Data Obfuscation	Exfiltration Over Other	Endpoint Denial of Service
Spearphishing via Service	Execution through API	BITS Jobs	DLL Search Order	Compile After Delivery	Exploitation for Credentials	Network Sniffing	Pass the Ticket	Data Staged	Domain Fronting	Exfiltration Over Physical	Firmware Corruption
Supply Chain Compromise	Execution through Module	Bootkit	Elevated Execution with	Compiled HTML File	Forced Authentication	Password Policy Discovery	Remote Desktop Protocol	Email Collection	Domain Generation	Scheduled Transfer	Inhibit System Recovery
Trusted Relationship	Exploitation for Client	Browser Extensions	Emond	Component Firmware	Hooking	Peripheral Device	Remote File Copy	Input Capture	Fallback Channels		Network Denial of Service
Valid Accounts	Graphical User Interface	Change Default File	Exploitation for Privilege	Component Object Model	Input Capture	Permission Groups	Remote Services	Man in the Browser	Multi-hop Proxy		Resource Hijacking
	Launchctl	Component Firmware	Extra Window Memory	Connection Proxy	Input Prompt	Process Discovery	Remote System Discovery	Screen Capture	Multi-Stage Channels		Runtime Data Manipulation
	Local Job Scheduling	Component Object Model	File System Permissions	Control Panel Items	Kerberoasting	Query Registry	Replication Through	Video Capture	Multiband Communication		Service Stop
	LSASS Driver	Create Account	Hooking	DCShadow	Keychain	Remote System Discovery	Shared Webroot		Multilayer Encryption		Stored Data Manipulation
	Mshta	DLL Search Order	Image File Execution	Deobfuscate/Decode Files	LLMNR/NBT-NS Poisoning	Security Software	SSH Hijacking		Port Knocking		System Shutdown/Reboot
	PowerShell	Dylib Hijacking	Launch Daemon	Disabling Security Tools	Network Sniffing	Software Discovery	Taint Shared Content		Remote Access Tools		Transmitted Data
	Regsvcs/Regasm	Emond	New Service	DLL Search Order	Password Filter DLL	System Information	Windows Admin Shares		Remote File Copy		
	Regsvr32	External Remote Services	Parent PID Spoofing	DLL Side-Loading	Private Keys	System Network	Windows Remote		Standard Application Layer		
	Rundll32	File System Permissions	Plist Modification	Execution Guardrails	Securityd Memory	System Network Discovery			Standard Cryptographic		
	Scheduled Task	Hidden Files and	Port Monitors	Exploitation for Defense	Steal Web Session Cookie	System Owner/User			Standard Non-Application		
	Scripting	Hooking	PowerShell Profile	Extra Window Memory	Two-Factor Authentication	System Service Discovery			Uncommonly Used Port		
	Service Execution	Hypervisor	Process Injection	File Deletion		System Time Discovery			Web Service		
	Signed Binary Proxy	Image File Execution	Scheduled Task	File System Logical Offsets		Virtualization/Sandbox					
	Signed Script Proxy	Kernel Modules and	Service Registry	File Permissions							
	Source	Launch Agent	Setuid and Setgid	Gatekeeper Bypass							
	Space after Filename	Launch Daemon	SID-History Injection	Group Policy Modification							
	Third-party Software	Launchctl	Startup Items	Hidden Files and							
	Trap	LC_LOAD_DYLIB Addition	Sudo	Hidden Users							
	Trusted Developer Utilities	Local Job Scheduling	Sudo Caching	Hidden Window							
	User Execution	Login Item	Valid Accounts	HISTCONTROL							
	Windows Management	Logon Scripts	Web Shell	Indicator Blocking							
	Windows Remote	LSASS Driver		Indicator Removal from							
	XSL Script Processing	Modify Existing Service		Indicator Removal on Host							
		Netsh Helper DLL		Indirect Command							
		New Service		Install Root Certificate							
		Office Application Startup		InstallUtil							
		Path Interception		Launchctl							
		Plist Modification		LC_MAIN Hijacking							
		Port Knocking		Masquerading							
		Port Monitors		Modify Registry							
		PowerShell Profile		Mshta							
		Rc.common		Network Share Connection							
		Re-opened Applications		NTFS File Attributes							
		Redundant Access		Obfuscated Files or							
		Registry Run Keys /		Parent PID Spoofing							
		Scheduled Task		Plist Modification							
		Screensaver		Port Knocking							
		Security Support Provider		Process Doppelgänging							
		Server Software		Process Hollowing							
		Service Registry		Process Injection							
		Setuid and Setgid		Redundant Access							
		Shortcut Modification		Regsvcs/Regasm							
		SIP and Trust Provider		Regsvr32							
		Startup Items		Rootkit							
		System Firmware		Rundll32							
		System Service		Scripting							
		Time Providers		Signed Binary Proxy							
		Trap		Signed Script Proxy							
		Valid Accounts		SIP and Trust Provider							
		Web Shell		Software Packing							
		Windows Management		Space after Filename							
		Winlogon Helper DLL		Template Injection							
				Timestomp							
				Trusted Developer Utilities							
				Valid Accounts							
				Virtualization/Sandbox							
				Web Service							
				XSL Script Processing							

Breakdown

Step	High Level Overview of Emulation and Techniques Evaluated	Cited Intelligence	Open Invitation Contributor(s)	Emulation Content
11	The scenario begins with initial breach, where a legitimate user clicks (T1204) a link file payload, which executes an alternate data stream (ADS) hidden on another dummy file (T1096) delivered as part of the spearphishing campaign. The ADS performs a series of enumeration commands to ensure it is not executing in a virtualized analysis environment (T1497, T1082, T1120, T1033, T1016, T1057, T1083) before establishing persistence via a Windows Registry Run key entry (T1060) pointing to an embedded DLL payload that was decoded and dropped to disk (T1140). The ADS then executes a PowerShell stager (T1086) which creates a C2 connection over port 443 (T1043) using the HTTPS protocol (T1071, T1032).	APT29 has used several persistence mechanisms, including, Registry run keys. [5] [11] APT29 phishing campaigns have contained weaponized Windows shortcut files that executed an obfuscated PowerShell command from within the file and dropped a DLL to the victim's system. [8] [11] [17] PowerDuke has performed anti-VM checks designed to avoid executing in virtualized environments. PowerDuke payloads have also contained a component hidden in an ADS and connected to C2 over port 443. [11] Note: The anti-analysis commands and logic were derived from a VirusTotal submission. [9]	Microsoft	The Day 2 README.md file describes how to configure the schemas.ps1, 2016_United_States_presidential_election_-_Wikipedia.html and 37486-the-shocking-truth-about-election-rigging-in-america.rtf.lnk payloads, as well as additional commands to complete the step.
12	The attacker modifies the time attributes of the DLL payload (T1099) used in the previously established persistence mechanism to match that of a random file found in the victim's System32 directory (T1083). The attacker then enumerates registered AV products (T1063) and software installed by the user documented in the Windows Registry (T1012).	POSHSPY can modify standard information timestamps of downloaded executables to match a randomly selected file from the System32 directory. PowerDuke also has had undescribed commands named "detectav" and "software." [10]	Kaspersky SentinelOne	The Day 2 README.md file contains the commands to complete the step, including executing the timestomp function within timestomp.ps1 and the detectav and software functions within stepTwelve.ps1.
13	The attacker performs local enumeration using various Windows API calls, specifically gathering the local computer name (T1082), domain name (T1063), current user context (T1016), and running processes (T1057).	PowerDuke can get the NetBIOS name, the computer's domain name, user's name, and process list via select Windows API calls. [11]		The Day 2 README.md file contains the commands to complete the step, including executing the comp, domain, user, and pslist functions within stepThirteen.ps1.

Step	High Level Overview of Emulation and Techniques Evaluated	Cited Intelligence	Open Invitation Contributor(s)	Emulation Content
14	The attacker elevates privileges via a user account control (UAC) bypass **(T1122, T1088)**. The attacker then uses the new elevated access to create and execute code within a custom WMI class **(T1047)** that downloads **(T1105)** and executes Mimikatz to dump plain-text credentials **(T1003)**, which are parsed, encoded, and stored in the WMI class **(T1027)**. After tracking that the WMI execution has completed **(T1057)**, the attacker reads the plaintext credentials stored within the WMI class **(T1140)**.	APT29 has embedded and encoded PowerShell scripts in WMI class properties. [5] [10] APT29 has bypassed UAC to elevate privileges. [5] APT29 has used WMI to store and run Invoke-Mimikatz (ATT&CK S0002) on remote hosts. [7] [12] POSHSPY has used WMI to both store and persist PowerShell backdoor code. POSHSPY can also download and execute additional PowerShell code and Windows binaries. [7] [10] [12]	Microsoft SentinelOne	The Day 2 README.md file describes how to configure the stepFourteen_bypassUAC.ps1 and stepFourteen_credDump.ps1 payloads, as well as additional commands to complete the step, including executing the bypass function within stepFourteen_bypassUAC.ps1 and the wmidump function within stepFourteen_credDump.ps1.
15	The attacker establishes a secondary means of persistent access to the victim by creating a WMI event subscription **(T1084)** to execute a PowerShell payload whenever the current user **(T1033)** logs in.	APT29 has used several persistence mechanisms, including WMI backdoors that execute PowerShell components. [5] [10]	Microsoft SentinelOne	The Day 2 README.md file describes how to configure the stepFifteen_wmi.ps1 payload, as well as additional commands to complete the step, including executing the wmi function within stepFifteen_wmi.ps1.
16	The attacker enumerates the environment's domain controller **(T1018)** and the domain's security identifier (SID) **(T1033)** via the Windows API **(T1106)**. Next, the attacker uses the previously dumped credentials **(T1078)** to create a remote PowerShell session to the domain controller **(T1028)**. Through this connection, the attacker copies the Mimikatz binary used in Step 14 to the domain controller **(T1105)** then dumps the hash of the KRBTGT account **(T1003)**.	PowerDuke can get the current user's SID via select Windows API calls. [11]	Microsoft SentinelOne	The Day 2 README.md file contains the commands to complete the step, including executing the Get-NetDomainController function within powerView.ps1, the siduser function within stepSixteen_SID.ps1, and the Invoke-WinRMSession function within Invoke-WinRMSession.ps1.
17	The attacker harvests emails stored in the local email client **(T1114)** before collecting **(T1005)** and staging **(T1074)** a file of interest. The staged file is compressed **(T1002)** as well as prepended with the magic bytes of the GIF file type **(T1027)**.	APT29 has used the legit Microsoft DLL and PowerShell to interact with Exchange Web Services (EWS) for email theft. [7] POSHSPY can appended a file signature header to all encrypted data prior to upload or download. [10]	Kaspersky Microsoft	The Day 2 README.md file contains the commands to complete the step, including executing the psemail function within stepSeventeen_email.ps1 and the zip function within stepSeventeen_zip.ps1.
18	The attacker maps a local drive to an online web service account **(T1102)** then exfiltrates the previous staged data to this repository **(T1048)**.	CloudDuke can use a Microsoft OneDrive to exchange stolen data with its operators. [1] [5]	Kaspersky Microsoft SentinelOne	The Day 2 README.md file contains the commands to complete the step.

Step	High Level Overview of Emulation and Techniques Evaluated	Cited Intelligence	Open Invitation Contributor(s)	Emulation Content
19	The attacker deletes various files **(T1107)** associated with that access by reflectively loading and executing the Sdelete binary **(T1055)** within powershell.exe.	APT29 has removed tools and forensic artifacts to hide activity, including the usage of Sdelete (ATT&CK S0195). [5] PowerDuke can write random data across then delete a file. [11]	Microsoft SentinelOne	The Day 2 README.md file contains the commands to complete the step, including executing the wipe function within wipe.ps1.
20	The original victim is rebooted and the legitimate user logs in, emulating ordinary usage and a passage of time. This activity triggers the previously established persistence mechanisms, namely the execution of the DLL payload **(T1085)**, referenced by the Windows Registry Run key, and the WMI event subscription **(T1084)**, which executes a new PowerShell stager **(T1086)**. The attacker uses the renewed access to generate a Kerberos Golden Ticket **(T1097)**, using materials from the earlier breach, which is used to establish a remote PowerShell session to a new victim **(T1028)**. Through this connection, the attacker creates a new account within the domain **(T1136)**.	APT29 have used Kerberos ticket attacks for lateral movement and has created accounts to log in. [5] [7]	Microsoft SentinelOne	The Day 2 README.md file contains the commands to complete the step, including executing the Invoke-Mimikatz function within Invoke-Mimikatz.ps1.

References

[1] THE DUKES: 7 years of Russian cyberespionage - https://www.f-secure.com/documents/996508/1030745/dukes_whitepaper.pdf

[2] COSMICDUKE: Cosmu with a twist of MiniDuke - https://www.f-secure.com/documents/996508/1030745/cosmicduke_whitepaper.pdf

[3] The MiniDuke Mystery: PDF 0-day Government Spy Assembler 0x29A Micro Backdoor - https://securelist.com/the-miniduke-mystery-pdf-0-day-government-spy-assembler-0x29a-micro-backdoor/31112/

[4] Unit 42 Technical Analysis: Seaduke - https://unit42.paloaltonetworks.com/unit-42-technical-analysis-seaduke/

[5] No Easy Breach DerbyCon 2016 - https://www.slideshare.net/MatthewDunwoody1/no-easy-breach-derby-con-2016

[6] HAMMERTOSS: Stealthy Tactics Define a Russian Cyber Threat Group - https://www2.fireeye.com/rs/848-DID-242/images/rpt-apt29-hammertoss.pdf

[7] State of the Hack S2E01: #NoEasyBreach REVISITED - https://www.fireeye.com/blog/products-and-services/2019/02/state-of-the-hack-no-easy-breach-revisited.html

[8] Not So Cozy: An Uncomfortable Examination of a Suspected APT29 Phishing Campaign - https://www.fireeye.com/blog/threat-research/2018/11/not-so-cozy-an-uncomfortable-examination-of-a-suspected-apt29-phishing-campaign.html

[9] VirusTotal Submission 2f39dee2ee608e39917cc022d9aae399959e967a2dd70d83b81785a98bd9ed36 - https://www.virustotal.com/gui/file/2f39dee2ee608e39917cc022d9aae399959e967a2dd70d83b81785a98bd9ed36/detection

[10] Dissecting One of APT29's Fileless WMI and PowerShell Backdoors (POSHSPY) - https://www.fireeye.com/blog/threat-research/2017/03/dissecting_one_ofap.html

[11] PowerDuke: Widespread Post-Election Spear Phishing Campaigns Targeting Think Tanks and NGOs - https://www.volexity.com/blog/2016/11/09/powerduke-post-election-spear-phishing-campaigns-targeting-think-tanks-and-ngos/

[12] CrowdStrike's work with the Democratic National Committee: Setting the record straight - https://www.crowdstrike.com/blog/bears-midst-intrusion-democratic-national-committee/

[13] "Forkmeiamfamous": Seaduke, latest weapon in the Duke armory - https://www.symantec.com/connect/blogs/forkmeiamfamous-seaduke-latest-weapon-duke-armory

[14] GRIZZLY STEPPE – Russian Malicious Cyber Activity - https://www.us-cert.gov/sites/default/files/publications/JAR_16-20296A_GRIZZLY%20STEPPE-2016-1229.pdf

[15] CrowdStrike's work with the Democratic National Committee: Setting the record straight - https://www.crowdstrike.com/blog/bears-midst-intrusion-democratic-national-committee/

[16] The CozyDuke APT - https://securelist.com/the-cozyduke-apt/69731/

[17] Analysis of cyberattack on U.S. think tanks, non-profits, public sector by unidentified attackers - https://www.microsoft.com/security/blog/2018/12/03/analysis-of-cyberattack-on-u-s-think-tanks-non-profits-public-sector-by-unidentified-attackers/