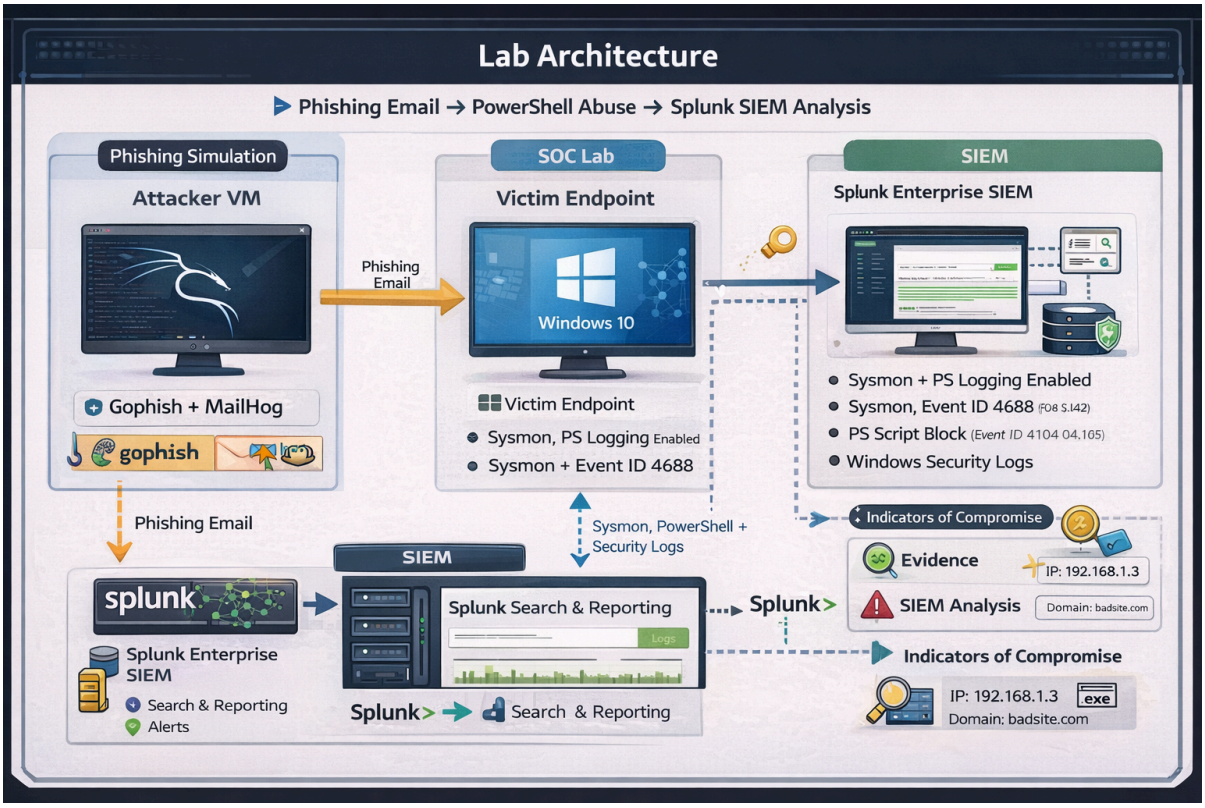
Phishing Attack Detection & Incident Investigation using Splunk SIEM (SOC Lab POC)

End-to-End Detection of Phishing Email Leading to PowerShell Execution -Shaiva Kumar Turyseril

---

Lab architecture diagram



PROJECT OVERVIEW

Purpose

This Proof of Concept demonstrates how a phishing email can lead to malicious PowerShell execution and how the activity is detected, investigated, and validated using Splunk SIEM in a SOC lab environment.

Objectives

- Simulate a real-world phishing attack
- Monitor endpoint behavior
- Detect PowerShell abuse
- Investigate alerts using SPL queries
- Document the incident like a real SOC analyst

Tools Used

- GoPhish (Phishing simulation)
- MailHog (SMTP capture)
- Windows 10 Endpoint
- Sysmon + PowerShell Logging
- Splunk Enterprise SIEM

LAB ENVIRONMENT SETUP

Infrastructure

- Attacker VM: Kali Linux
- Victim VM: Windows 10
- SIEM: Splunk Enterprise
- Log Forwarding: Splunk Universal Forwarder

Logging Enabled

- Windows Security Logs
- PowerShell Script Block Logging (Event ID 4104)
- Process Creation (Event ID 4688)
- Sysmon logs

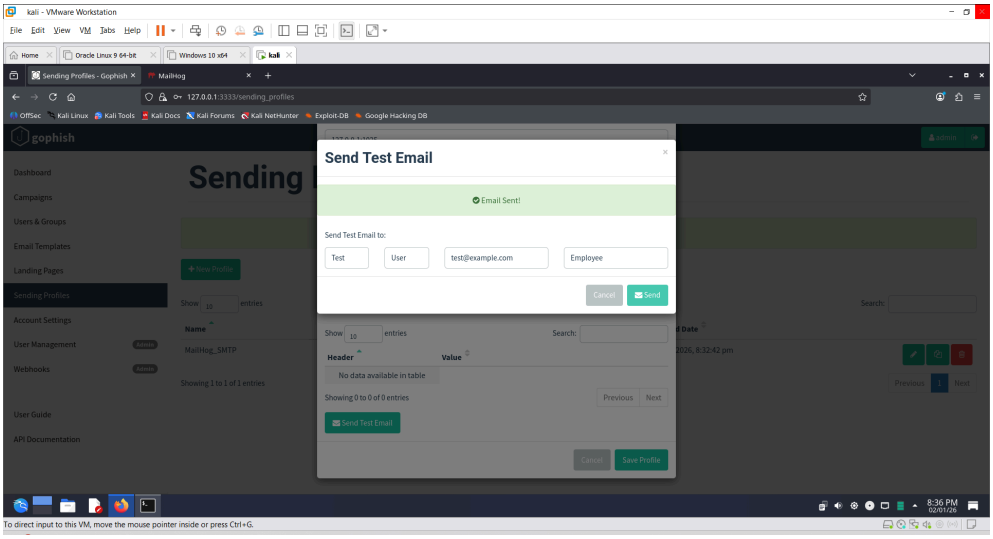
ATTACK SIMULATION – PHISHING STAGE

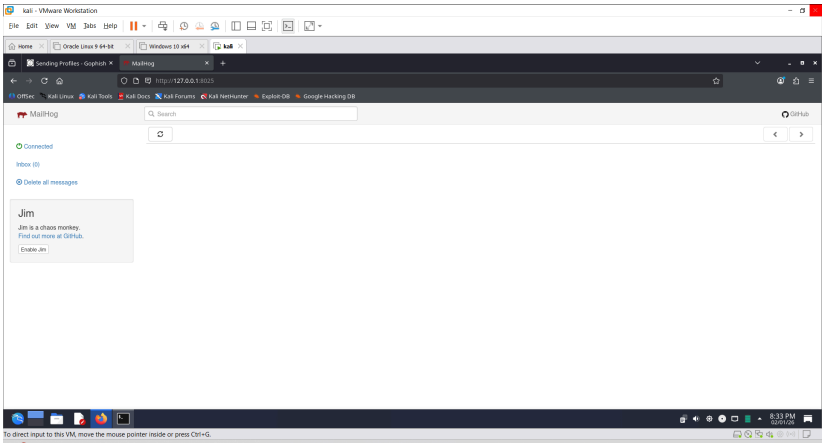
Phishing Email Creation

- Email template created in GoPhish
- Subject: *Password Expiring Today*
- Embedded malicious URL

Delivery

- Email sent via MailHog SMTP
- User received phishing email





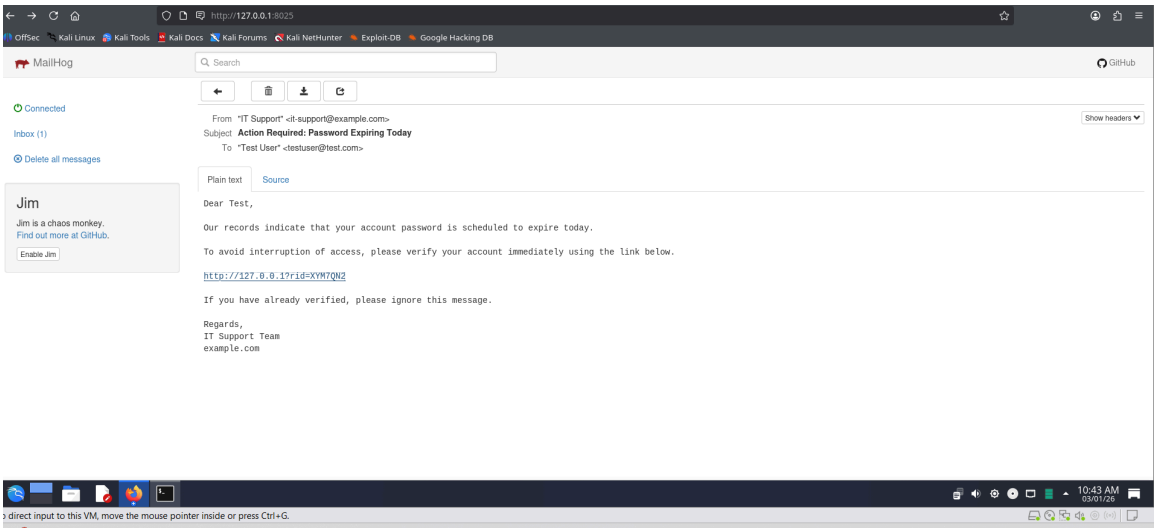
USER INTERACTION (INITIAL COMPROMISE)

User Action

- User clicks phishing link
- Redirected to fake verification landing page

Outcome

- Attack progresses to command execution phase



POST-CLICK ACTIVITY – POWERSHELL EXECUTION

Malicious Behavior Observed

- PowerShell executed with:
  - `-EncodedCommand`
  - `IEX`
  - Execution Policy Bypass

Windows Events Generated

- Event ID 4104 – Script Block Logging
- Event ID 4688 – Process Creation

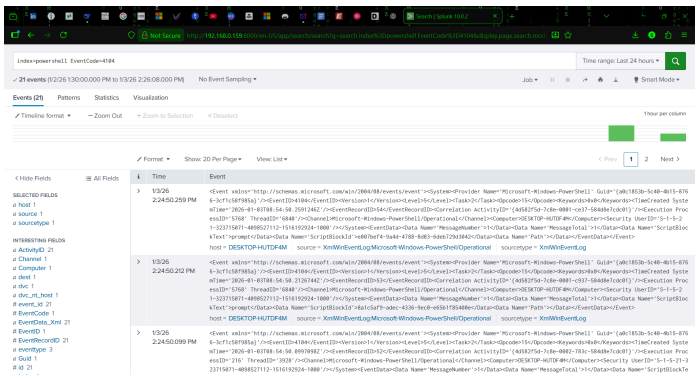
DETECTION IN SPLUNK (CORE SECTION)

Detection 1: PowerShell Script Block Logging

- index=powershell EventCode=4104

Why:

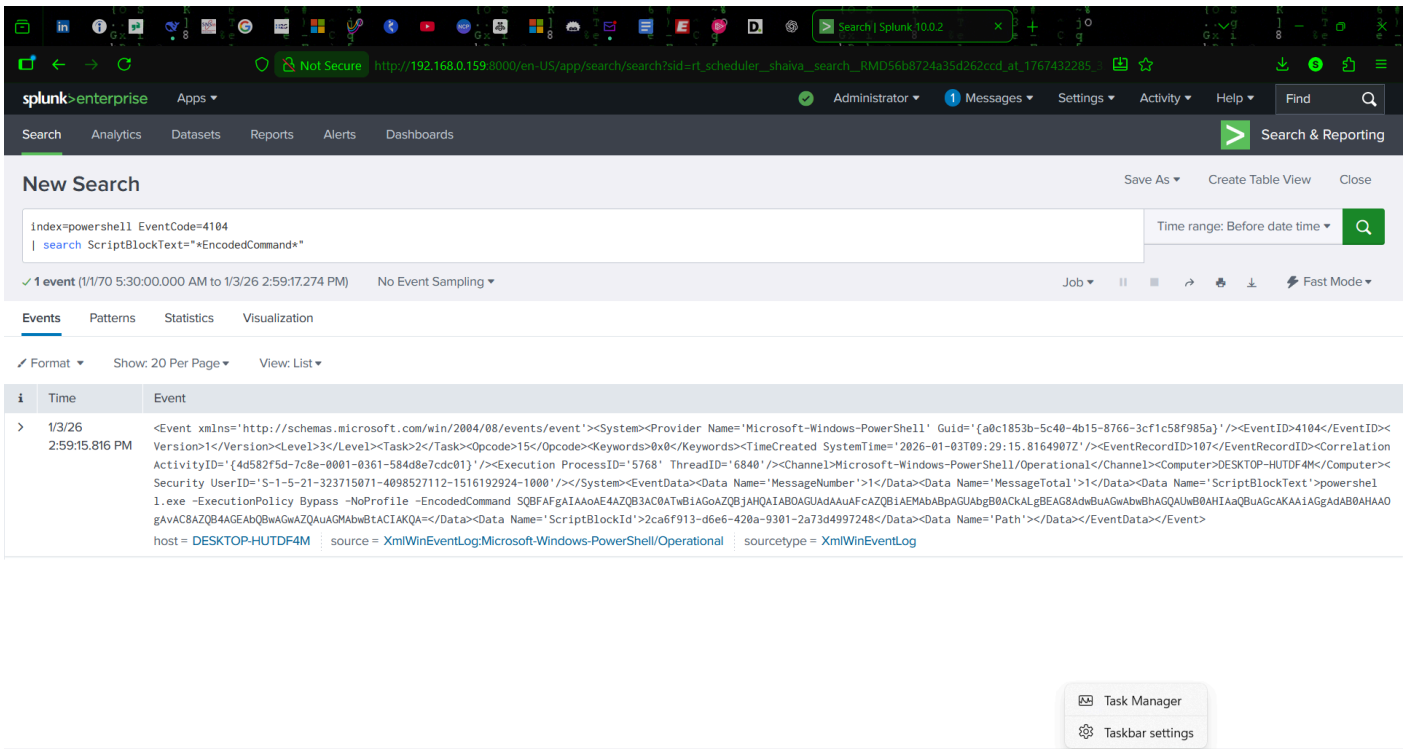
Detects PowerShell scripts executed on the endpoint.



Detection 2: Encoded PowerShell Commands

- index=powershell EventCode=4104 | search ScriptBlockText="*EncodedCommand*"

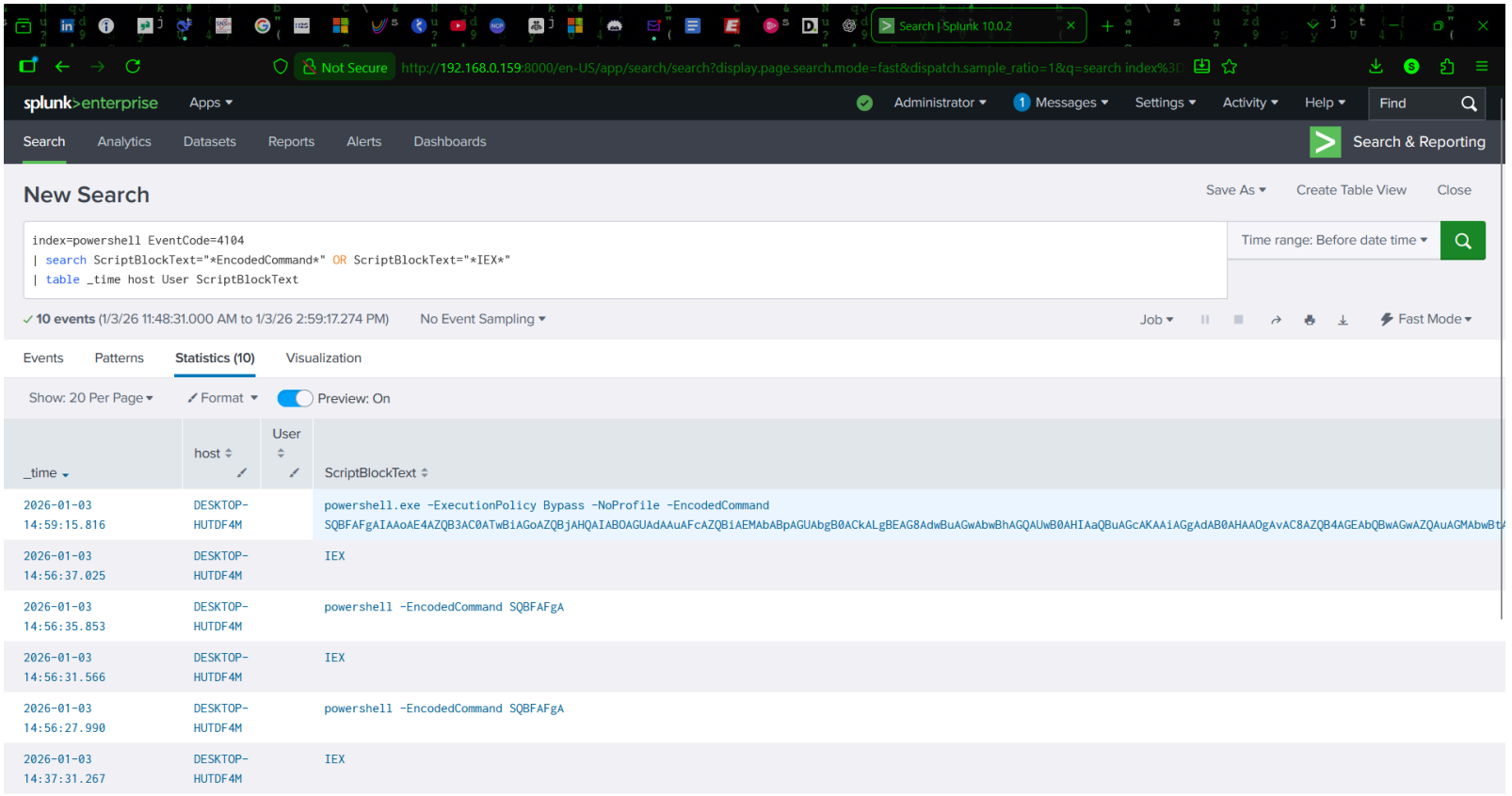
Why:
Encoded commands are commonly used by attackers to evade detection.



Detection 3: IEX Usage

- index=powershell EventCode=4104 | search ScriptBlockText="*IEX*"

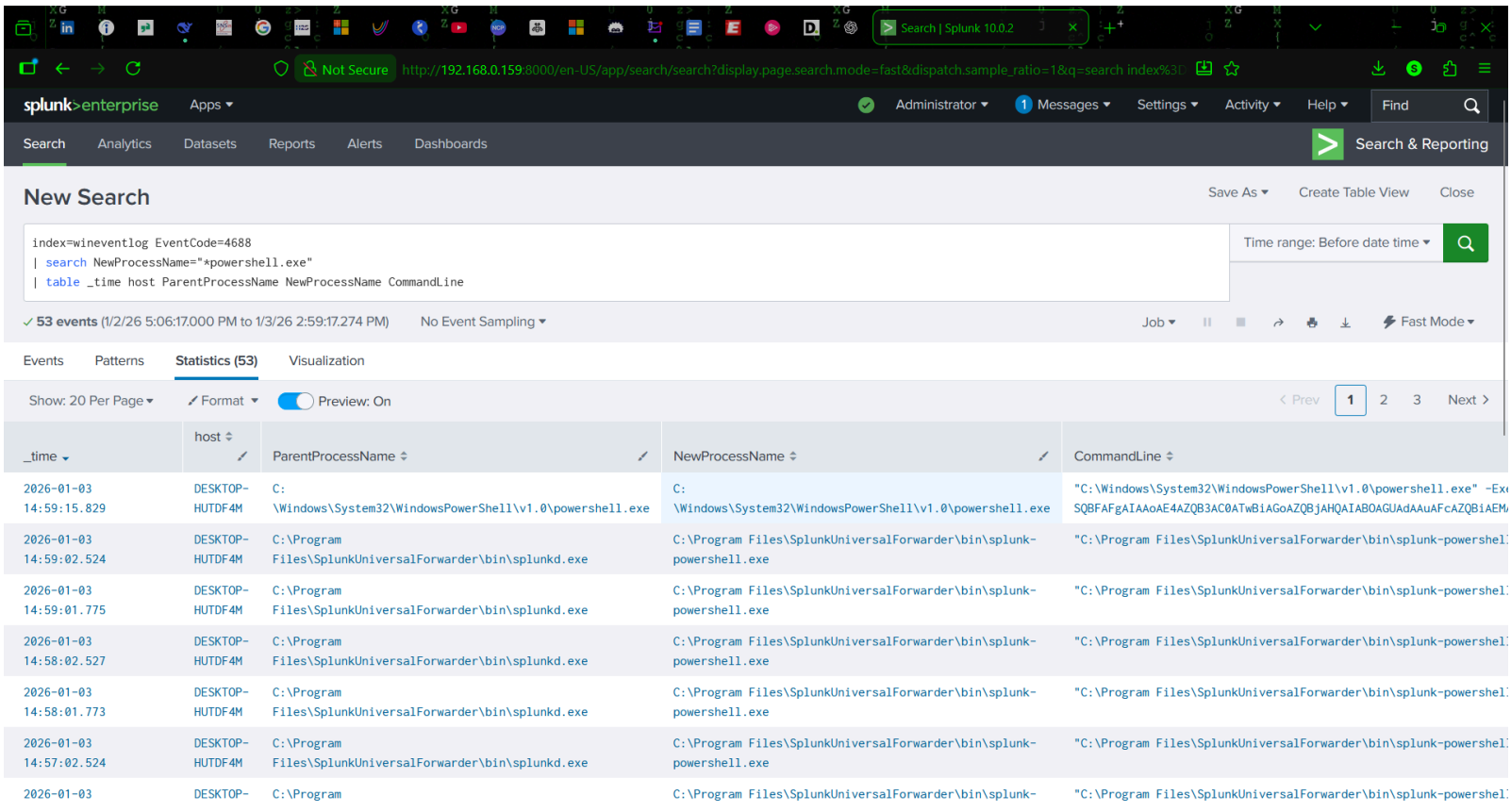
Why:
IEX (Invoke-Expression) is frequently used in fileless malware.



Detection 4: PowerShell Process Creation

- index=wineventlog EventCode=4688 | search NewProcessName="*powershell.exe*"

Why:
Confirms PowerShell execution at OS level.



ANALYST INVESTIGATION & VALIDATION

Analyst Actions

- Correlated PowerShell logs with process creation
- Identified encoded payload
- Decoded Base64 command in Cyber Chef
- Verified suspicious execution

Findings

- User-triggered execution
- No legitimate business justification
- High confidence malicious behavior

INCIDENT CLASSIFICATION

Incident Type	Phishing → Execution
MITRE ATT&CK	T1566, T1059.001
Severity	High
Status	Confirmed Malicious

RESPONSE & RECOMMENDATIONS

Immediate Actions

- Endpoint isolated using EDR to prevent further execution
- User credentials reset and authentication activity reviewed
- User awareness training
- Blocking similar PowerShell patterns
- Threat hunt conducted to identify similar PowerShell executions

Preventive Measures

- PowerShell hardening via Script Block Logging and disabling legacy PowerShell
- Email security strengthened using DMARC, DKIM, SPF and phishing domain filtering
- SIEM detection rules tuned and correlated to reduce false positives and improve early detection

---

CONCLUSION & LEARNING

This lab demonstrates real SOC workflows including:

- Threat detection
- Log correlation
- Alert validation
- Incident documentation

The POC closely mirrors real-world SOC investigations and highlights the importance of PowerShell monitoring in phishing-based attacks.