Scriptspråk

Kurs_NÄTD24LIN_SCSP25

Workshop 4: PowerShell - Configuration Audit & Log Analysis

https://github.com/muhad308/PowerShell---Configuration-Audit-Log-Analysis-Part-4

1. Problem Solving and Challenges:-

What were the biggest challenges you encountered in this workshop part?

- As I worked on the exercise questions, I faced some difficulties with PowerShell commands as I have more familiarity with using Linux bash commands. One of the problems I faced was using handling so many file types and conditions in one script.
  - **For Example:** Detecting weak passwords, missing backups and SNMP issues all required different logic.
- In reference material provided, "Export-Csv" and "Write-Host" is used only. it would be better to have more explanation on this part
  - **For Example:** When to use "Write-Output" , "Append" and ">>" etc

How did you solve them?

- I tried different ways
  - **For Example:** Streaming data directly to file storing data to variable and stream/push the variable to check data. Also looked to Microsoft reference material etc. I created separate counters and summaries to handle multiple statistics at once.
- I used pattern matching with regex and state tracking.
  - **For Example:** To handle banner configurations in the baseline compliance check, I used a flag ($inBannerBlock) to track when we are inside a multi-line banner block and skip those lines during comparison.
- By breaking tasks into clear sections: one loop for .conf files, one for .rules and another for logs.
- I also used pattern matching like if ($line -match "enable password|secret") to catch risky configurations. Managing file paths and ensuring the script worked recursively across folders also took careful testing.

2. Learning and Development

What have you learned that you couldn't do before?

- I learnt how using built-in functions like "Recurse" and "Get-ChildItem -Path" we can efficiently parse large chunk of data.
- I also learnt how "Select-String" can be used to capture data.
    - **For Example:** How to get Ip-address form files in sub-directories etc.
- How to automate large-scale file audits with PowerShell counting files, filtering by extension, checking timestamps and writing structured reports.
- I also improved my understanding of regular expressions for detecting insecure settings in config files.

Which concept(s) was/are the most difficult to understand and why?

- Understanding Function was difficult. It would be nice to have some explanation and a function-template showing what is mandatory/optional in functions usage.
- Handling nested loops efficiently while keeping the report readable and avoiding duplicate results.
- Baseline configuration compliance checking was challenging. It required comparing each router's configuration against a template while intelligently ignoring comments, empty lines and multi-line blocks which isn't a simple file comparison.

3. Professional Relevance

How can you use these skills in your future role as a network engineer?

- This kind of automation is highly valuable for a network engineer. It can quickly scan hundreds of network device configs for security issues, track failed logins and check compliance with baseline standards.
- I can now efficiently look to different network-logs and extract important information quickly.

Examples of real-world situations where this type of automation would be valuable:

- During troubleshooting, it is required to analyse the problem and fix quickly especially if network outage is impacting real-time applications. So, automation is required to analyse/debug lot of system, security and access logs and find the root-cause.
- Running a nightly audit on router configs could catch unencrypted HTTP rules or missing backups before they become problems.
- Quickly parsing through gigabytes of log files to identify attack patterns during a security breach.

4. Code Quality and Improvements

If you were to do the workshop part again, what would you do differently?

- More emphasis on searching the files with "Select-String" as I find it very powerful tool. Perhaps exploring techniques into how to check few lines of data above and below the search keyword etc.
- I would create reusable functions for common tasks like file parsing and pattern matching and parameterize the audit thresholds (like 7 days for backups) to make the script more flexible.

Which parts of your solution are you most satisfied with and why?

- "Security Review" section in workshop-4 is very satisfactory as I find it very powerful tool to parse many log-files without even opening them.
    - **For Example:** Iterate the root directory using "Recurse" and get the required information from all/selected logs.
- The comprehensive reporting that combines file inventory, security findings, log analysis, backup status and compliance checking into a single executive report. It transforms raw technical data into actionable insights.

5. Tool Selection

When are Python scripts and PowerShell scripts best suited? Explain and justify.

Tool	Best suited for	Why
Python	Data processing, automation, report generation	It has strong libraries for JSON, formatting, and is cross-platform. Great for multi-vendor and platform-independent environments.
PowerShell	Windows system administration, managing AD, Exchange or local Windows devices	Built for managing Windows environments, tightly integrated with system tools and Windows APIs.

Example:

- Use Python for generating a network-wide report from devices exported via API.
    - When you need extensive third-party libraries (for networking, data science, web scraping). Its readability and vast ecosystem make it superior for complex automation and integration projects.
- Use PowerShell to restart Windows services or check Windows Event Logs on servers.
    - When you need deep integration with Microsoft ecosystems (Active Directory, Exchange, Azure). Its object-oriented pipeline and seamless Windows API access make it ideal for system administration tasks.

- Choose PowerShell for Windows-centric automation and Python for complex, cross-platform network automation that might involve APIs, data analysis or web interfaces.