SOC Playbook: Command Shell Monitoring (T1059)

1. Objective

Detect, investigate, and respond to suspicious **command shell usage**, including adversarial use of interpreters for enumeration, persistence, lateral movement, or payload execution.

2. Scope

- Monitor command-line activity across Windows, Linux, macOS.
- Detect use of interactive shells, reverse shells, or fileless execution.
- Flag suspicious parent-child process relationships, unusual command-line arguments, and shell usage in restricted environments (e.g., IIS, SQL, SSH).

3. Log Sources

Platform	Log Source	Description
Windows	Security Logs (4688)	Tracks process creation
Windows	Sysmon (Event ID 1, 7, 11)	Detailed telemetry for command execution
Windows	PowerShell Logs (4104, 4103)	Command/script content
Linux/macOS	Auditd (execve syscall)	Captures command execution
All	EDR/XDR/OSQuery	Enhanced telemetry (command lines, users, etc.)
All	Network Logs (Zeek, NetFlow)	Shell over network detection (e.g., reverse shell)

4. Detection Rules / Alerts

Alert Name	Description	Triggers / Examples
Unusual Shell Execution	cmd.exe, sh, bash run from unknown context	Parent: winword.exe → cmd.exe
Interactive Shell from Service	Shell launched from webserver, SQL server, etc.	Parent: w3wp.exe, sqlservr.exe
Encoded or Obfuscated CLI	Suspicious use of base64 or escape sequences	powershell -EncodedCommand, bash -c $(echo ...)
Shell over Reverse Connection	Netcat or bash opening remote session	/dev/tcp/, nc -e /bin/sh
Chained Commands Detected	Command injection via shell chaining	cmd /c whoami && net user

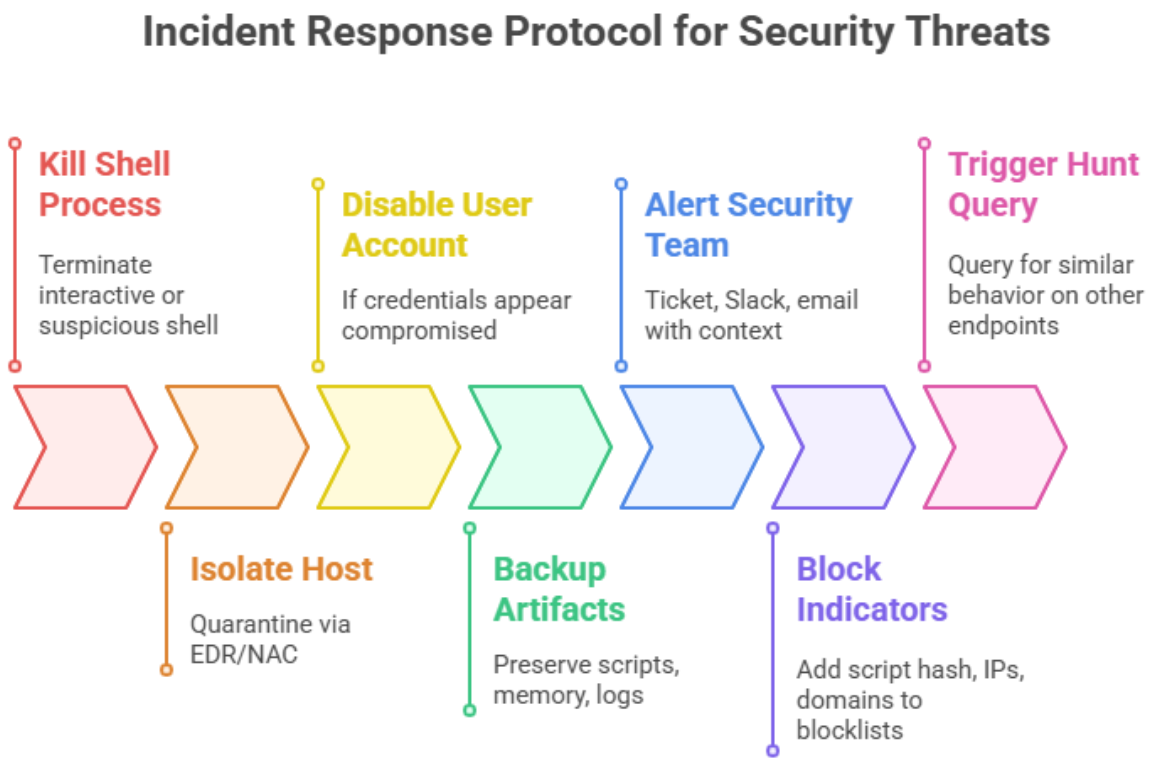
Shell via Scheduled Task or Service	Abnormal persistence execution context	schtasks.exe, systemd, cron spawning shell

5. Automated Enrichment

Enrichment Task	Description
User Context	Logged-in user, privilege level
Process Lineage	Full parent-child-grandparent chain
Command Line Extraction	Extract full shell arguments
Known Good Validation	Check against allowlisted admin tools
Hash Reputation	Scan binary hashes via VT / internal DB
Session Info	Terminal ID, remote IP, interactive Y/N

6. Automated Response Play

Step	Action
1. Kill Shell Process	Terminate interactive or suspicious shell
2. Isolate Host	Quarantine via EDR/NAC
3. Disable User Account	If credentials appear compromised
4. Backup Artifacts	Preserve scripts, memory, logs
5. Alert Security Team	Ticket, Slack, email with context
6. Block Indicators	Add script hash, IPs, domains to blocklists
7. Trigger Hunt Query	Query for similar behavior on other endpoints



7. Investigation Checklist

Step	Description
1. Review Alert Metadata	Source host, username, time, command line
2. Check Parent Process Tree	Is shell launched from trusted/expected source?
3. Analyze Command Arguments	Review for enumeration, C2, obfuscation
4. Validate Script or Payload	If present, decode and analyze
5. Inspect Network Connections	Check if shell opened reverse connection
6. Review Past User Behavior	Has this user launched shells before?
7. Persistence Check	Look for cron jobs, startup scripts, etc.
8. Correlate with Other Alerts	Multiple hosts, similar behavior?
9. Threat Intel Correlation	Known TTPs, APT overlaps, IOCs
10. Document Case	Summarize findings, attach evidence

8. Playbook Notes

- Enable full command-line auditing (4688 on Windows, auditd on Linux).
- Monitor shell launches from high-risk parent processes.
- Watch for signs of interactive shells where none should exist.
- Establish normal shell usage baselines per host/user group.
- Decode and deobfuscate command arguments for full visibility.