Rule query

Linux Restricted Shell Breakout via Linux Binary(s)

edit

Identifies the abuse of a Linux binary to break out of a restricted shell or environment by spawning an interactive system shell. The activity of spawning a shell from a binary is not common behavior for a user or system administrator, and may indicate an attempt to evade detection, increase capabilities or enhance the stability of an adversary.

Rule type: eql

Rule indices:

- logs-endpoint.events.*

Severity: medium

Risk score: 47

Runs every: 5m

Searches indices from: now-9m (Date Math format, see also `Additional look-back time`)

Maximum alerts per execution: 100

References:

- https://gtfobins.github.io/gtfobins/apt/
- https://gtfobins.github.io/gtfobins/apt-get/
- https://gtfobins.github.io/gtfobins/nawk/
- https://gtfobins.github.io/gtfobins/mawk/
- https://gtfobins.github.io/gtfobins/awk/
- https://gtfobins.github.io/gtfobins/gawk/
- https://gtfobins.github.io/gtfobins/busybox/
- https://gtfobins.github.io/gtfobins/c89/
- https://gtfobins.github.io/gtfobins/c99/
- https://gtfobins.github.io/gtfobins/cpulimit/
- https://gtfobins.github.io/gtfobins/crash/
- https://gtfobins.github.io/gtfobins/env/
- https://gtfobins.github.io/gtfobins/expect/
- https://gtfobins.github.io/gtfobins/find/
- https://gtfobins.github.io/gtfobins/flock/
- https://gtfobins.github.io/gtfobins/gcc/
- https://gtfobins.github.io/gtfobins/mysql/
- https://gtfobins.github.io/gtfobins/nice/
- https://gtfobins.github.io/gtfobins/ssh/
- https://gtfobins.github.io/gtfobins/vi/
- https://gtfobins.github.io/gtfobins/vim/

- Domain: Endpoint
- OS: Linux
- Use Case: Threat Detection
- Tactic: Execution
- Data Source: Elastic Endgame
- Data Source: Elastic Defend

Version: 113

Rule authors:

- Elastic

Rule license: Elastic License v2

Investigation guide

edit

Triage and analysis

Investigating Shell Evasion via Linux Utilities

Detection alerts from this rule indicate that a Linux utility has been abused to breakout of restricted shells or environments by spawning an interactive system shell. Here are some possible avenues of investigation: - Examine the entry point to the host and user in action via the Analyse View. - Identify the session entry leader and session user - Examine the contents of session leading to the abuse via the Session View. - Examine the command execution pattern in the session, which may lead to suspricous activities - Examine the execution of commands in the spawned shell. - Identify imment threat to the system from the executed commands - Take necessary incident response actions to contain any malicious behviour caused via this execution.

Related rules

- A malicious spawned shell can execute any of the possible MITTRE ATT&CK vectors mainly to impair defences.
- Hence its adviced to enable defence evasion and privilige escalation rules accordingly in your environment

Response and remediation

Initiate the incident response process based on the outcome of the triage.

- If the triage releaved suspicious netwrok activity from the malicious spawned shell,
- Isolate the involved host to prevent further post-compromise behavior.
- If the triage identified malware execution via the maliciously spawned shell,
- Search the environment for additional compromised hosts.
- Implement temporary network rules, procedures, and segmentation to contain the malware.
- Stop suspicious processes.

- If any tools have been disbaled / uninstalled or config tampered work towards reenabling the same.
- If the triage revelaed addition of persistence mechanism exploit like auto start scripts
- Isolate further login to the systems that can initae auto start scripts.
- Identify the auto start scripts and disable and remove the same from the systems
- If the triage revealed data crawling or data export via remote copy
- Investigate credential exposure on systems compromised / used / decoded by the attacker during the data crawling
- Intiate compromised credential deactivation and credential rotation process for all exposed crednetials.
- Investiagte if any IPR data was accessed during the data crawling and take appropriate actions.
- Determine the initial vector abused by the attacker and take action to prevent reinfection through the same vector.
- Using the incident response data, update logging and audit policies to improve the mean time to detect (MTTD) and the mean time to respond (MTTR).

Setup

edit

Setup

This rule requires data coming in from Elastic Defend.

Elastic Defend Integration Setup

Elastic Defend is integrated into the Elastic Agent using Fleet. Upon configuration, the integration allows the Elastic Agent to monitor events on your host and send data to the Elastic Security app.

Prerequisite Requirements:

- Fleet is required for Elastic Defend.
- To configure Fleet Server refer to the documentation.

The following steps should be executed in order to add the Elastic Defend integration on a Linux System:

- Go to the Kibana home page and click "Add integrations".
- In the query bar, search for "Elastic Defend" and select the integration to see more details about it.
- Click "Add Elastic Defend".
- Configure the integration name and optionally add a description.
- Select the type of environment you want to protect, either "Traditional Endpoints" or "Cloud Workloads".
- Select a configuration preset. Each preset comes with different default settings for Elastic Agent, you can further customize these later by configuring the Elastic Defend integration policy. Helper guide.

- To complete the integration, select “Add Elastic Agent to your hosts” and continue to the next section to install the Elastic Agent on your hosts. For more details on Elastic Defend refer to the helper guide.

Session View uses process data collected by the Elastic Defend integration, but this data is not always collected by default. Session View is available on enterprise subscription for versions 8.3 and above.

To confirm that Session View data is enabled:

- Go to “Manage → Policies”, and edit one or more of your Elastic Defend integration policies.
- Select the” Policy settings” tab, then scroll down to the “Linux event collection” section near the bottom.
- Check the box for “Process events”, and turn on the “Include session data” toggle.
- If you want to include file and network alerts in Session View, check the boxes for “Network and File events”.
- If you want to enable terminal output capture, turn on the “Capture terminal output” toggle. For more information about the additional fields collected when this setting is enabled and the usage of Session View for Analysis refer to the helper guide.

Rule query

edit

```
    (process.parent.name == "git" and process.parent.args : ("*
     process.args : ("*PAGER*", "!*sh", "exec *sh") and not pro
    (process.parent.name : ("byebug", "ftp", "strace", "zip", "
    (
      process.parent.args : "BEGIN {system(*)}" or
      (process.parent.args : ("*PAGER*", "!*sh", "exec *sh") or
      (
        (process.parent.args : "exec=*sh" or (process.parent.ar
        (process.args : "exec=*sh" or (process.args : "-I" and
        )
      )
    ) or

    /* shells specified in parent args */
    /* nice rule is broken in 8.2 */
    (process.parent.args : "*sh" and
      (
        (process.parent.name == "nice") or
        (process.parent.name == "cpulimit" and process.parent.a
        (process.parent.name == "find" and process.parent.args
         process.parent.args == ";" and process.parent.args : "
        (process.parent.name == "flock" and process.parent.args
      )
    )
  )) or

  /* shells specified in args */
  (process.args : "*sh" and (
    (process.parent.name == "crash" and process.parent.args ==
    (process.name == "sensible-pager" and process.parent.name i
    /* scope to include more sensible-pager invoked shells with

  )) or
  (process.name == "busybox" and event.action == "exec" and pro
   process.executable : "/var/lib/docker/overlay2/*/merged/bin/
   process.parent.args == "runc") and not process.parent.args i
  (process.name == "env" and process.args_count == 2 and proces
  (process.parent.name in ("vi", "vim") and process.parent.args
  (process.parent.name in ("c89", "c99", "gcc") and process.par
  (process.parent.name == "expect" and process.parent.args == "
  (process.parent.name == "mysql" and process.parent.args == "-
  (process.parent.name == "ssh" and process.parent.args == "-o"
)
```

Framework: MITRE ATT&CK™

- Tactic:
  - Name: Execution
  - ID: TA0002
  - Reference URL: https://attack.mitre.org/tactics/TA0002/

- ID: T1059.004
- Reference URL: https://attack.mitre.org/techniques/T1059/004/

« Linux Process Hooking via GDB   Linux SSH X11 Forwarding »

Follow us

Blog

Newsroom

Become a partner

Join us

Careers

Career portal

Trust & Security

Trust center

EthicsPoint portal

ECCN report

Ethics email

Investor relations

Investor resources

Governance

Financials

Stock

EXCELLENCE AWARDS

Previous winners

ElasticON Tour

Become a sponsor

All events

Trademarks Terms of Use Privacy Sitemap