Potential Remote Desktop Tunneling Detected

Identifies potential use of an SSH utility to establish RDP over a reverse SSH Tunnel. This can be used by attackers to enable routing of network packets that would otherwise not reach their intended destination.

Rule type: eql

Rule indices:

- logs-endpoint.events.process-*
- winlogbeat-*
- logs-windows.*
- endgame-*
- logs-system.security*
- logs-sentinel_one_cloud_funnel.*
- logs-m365_defender.event-*

Severity: high

Risk score: 73

Runs every: 5m

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

Maximum alerts per execution: 100

References:

- https://blog.netspi.com/how-to-access-rdp-over-a-reverse-ssh-tunnel/

Tags:

- Domain: Endpoint
- OS: Windows
- Use Case: Threat Detection
- Tactic: Command and Control
- Tactic: Lateral Movement
- Resources: Investigation Guide
- Data Source: Elastic Endgame
- Data Source: Elastic Defend
- Data Source: SentinelOne
- Data Source: Microsoft Defender for Endpoint
- Data Source: System

Version: 415

Rule authors:

- Elastic

Rule license: Elastic License v2

Investigation guide

edit

Triage and analysis

Investigating Potential Remote Desktop Tunneling Detected

Protocol Tunneling is a mechanism that involves explicitly encapsulating a protocol within another for various use cases, ranging from providing an outer layer of encryption (similar to a VPN) to enabling traffic that network appliances would filter to reach their destination.

Attackers may tunnel Remote Desktop Protocol (RDP) traffic through other protocols like Secure Shell (SSH) to bypass network restrictions that block incoming RDP connections but may be more permissive to other protocols.

This rule looks for command lines involving the `3389` port, which RDP uses by default and options commonly associated with tools that perform tunneling.

Possible investigation steps

- Investigate the process execution chain (parent process tree) for unknown processes. Examine their executable files for prevalence, whether they are located in expected locations, and if they are signed with valid digital signatures.
- Identify the user account that performed the action and whether it should perform this kind of action.
- Contact the account and system owners and confirm whether they are aware of this activity.
- Investigate other alerts associated with the user/host during the past 48 hours.
- Assess whether this behavior is prevalent in the environment by looking for similar occurrences across hosts.
- Examine network data to determine if the host communicated with external servers using the tunnel.

False positive analysis

- This activity is unlikely to happen legitimately. Benign true positives (B-TPs) can be added as exceptions if necessary.
- Investigate the command line for the execution of programs that are unrelated to tunneling, like Remote Desktop clients.

Response and remediation

- Initiate the incident response process based on the outcome of the triage.
- Isolate the involved host to prevent further post-compromise behavior.
- Take the necessary actions to disable the tunneling, which can be a process kill, service deletion, registry key modification, etc. Inspect the host to learn which method was used and to determine a response for the case.
- Investigate credential exposure on systems compromised or used by the attacker to ensure all compromised accounts are identified. Reset passwords for these accounts and other potentially compromised credentials, such as email, business systems, and web services.
- Run a full antimalware scan. This may reveal additional artifacts left in the system, persistence mechanisms, and malware components.
- 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

Setup

If enabling an EQL rule on a non-elastic-agent index (such as beats) for versions <8.2, events will not define `event.ingested`

and default fallback for EQL rules was not added until version 8.2. Hence for this rule to work effectively, users will need to add a custom ingest pipeline to populate `event.ingested` to @timestamp. For more details on adding a custom ingest pipeline refer - https://www.elastic.co/guide/en/fleet/current/data-streams-pipeline-tutorial.html

Rule query

edit

```
process where host.os.type == "windows" and event.ty
  /* RDP port and usual SSH tunneling related switch
  process.args : "*:3389" and
  process.args : ("-L", "-P", "-R", "-pw", "-ssh")
```

Framework: MITRE ATT&CK™

- Tactic:
  - Name: Command and Control
  - ID: TA0011
  - Reference URL: https://attack.mitre.org/tactics/TA0011/
- Technique:
  - Name: Protocol Tunneling
  - ID: T1572
  - Reference URL: https://attack.mitre.org/techniques/T1572/
- Tactic:
  - Name: Lateral Movement
  - ID: TA0008

    - Reference URL: https://attack.mitre.org/tactics/TA0008/
- Technique:
    - Name: Remote Services
    - ID: T1021
    - Reference URL: https://attack.mitre.org/techniques/T1021/
- Sub-technique:
    - Name: SSH
    - ID: T1021.004
    - Reference URL: https://attack.mitre.org/techniques/T1021/004/

« Potential Remote Desktop Shadowing Activity

Potential Remote File Execution via MSIEXEC »