Unauthorized Access Incident Response Playbook

Scenario Summary:

- **User**: jcarter
- **Attack vector**: Brute-force login attempts
- **Indicators**:
  - Multiple failed logins from `203.0.113.88`
  - Successful login followed by admin privilege escalation
- **Assets affected**: WIN-SQL01
- **Risk**: Unauthorized admin access, possible lateral movement
- **Source**: Log analysis using internal events (event codes 4625, 4624, 4672)

Purpose

This playbook walks through how I would respond to a specific type of cybersecurity incident: a brute-force login attempt that escalated into unauthorized admin access. The goal is to show the steps a security team should take from the moment an alert is triggered all the way through the containment and recovery.

While this case uses a simulated environment, the format and process follow real-world expectations for an entry-level SOC role or analyst job.

Using this playbook

Follow the steps in this playbook in the order in which they are listed. Note that steps may overlap.

Step 1: Alert Triggered

A system alert is generated showing multiple failed login attempts from an external IP address `203.0.113.88`. This is the what and when, it tells us a potential attack is starting. The user account being targeted is `jcarter`, and the system under attack is `WIN-SQL01`. (a Window based SQL server).

What happened next?

Eventually, the attacker guesses the correct password, successfully logs in, and then elevates the account's privileges to admin level.

The alert is built on Windows Even Log entries:

- `4625` = failed login
- `4624` = successful login
- `4672` = special privileges assigned (admin or equivalent)

Step 2: Evaluate the alert

Upon receiving the alert, investigate the alert details and any relevant log information. Here is the breakdown of how this was handled in the context of a brute-force login event.

1. **Alert severity**

   Type: Brute-force login + privilege escalation
   - **High**: Requires immediate escalation due to the unauthorized admin access to a critical asset (`WIN-SQL01`).

Step 3: Escalate the Alert

Once I confirmed the attack was real, multiple failed logins, successful access from a suspicious IP, and immediate privilege escalation, I knew this needed to be escalated to Tier 2 or a senior SOC analyst.

Why escalate?

Because I'm acting in the Tier 1 role here, and anytime there's confirmed admin-level access by an unknown or external actor, it's a serious event. This isn't a false positive or low-severity nuisance alert. It's a breach attempt that succeeded.

How I know it met the threshold:

- "Event ID `4625` = failed login"
- "Event ID `4624` = successful login"
- "Event ID `4672` = special privileges assigned (admin or equivalent).

These are red flags, especially in sequence.

I would:

- Add all the event IDs and their timestamps to the incident ticket
- Note the IP `203.0.113.88` is not on our allowlist
- Flag the account `jcarter` for immediate review
- Mark the incident as Escalated in the tracking system

Then I'd notify the appropriate Tier 2 responder, hand off the logs, and notify endpoint detection teams to scan for lateral movement.

Step 4: Contain the Threat

Now that it's confirmed and escalated, containment begins. This step is all about limiting the attacker's access before they can cause more damage.

My immediate containment steps:

1. Disable the compromised user account (`jcarter`)
2. Remove all administrative privileges from the account
3. Isolate the affected assets (`WIN-SQL01`) from the network
4. Notify endpoint detection teams to scan for lateral movement
5. Check for new accounts, backdoors, scheduled tasks, etc.

Why isolate the server?

If the attacker planted malware, created new users, or opened RDP ports, this prevents further spread while we investigate.

This part is high priority and time-sensitive. Contain first, analyze deeper later.

Step 5: Eradicate the Threat

Once the attacker is locked out, the goal shifts to cleaning up whatever they left behind.

Tasks in this phase include:

- Scan `WIN-SQL01` with antivirus/EDR tools
- Search for:
    - Suspicious registry entries
    - Unauthorized startup scripts
    - Unknown executables
- Remove or quarantine anything malicious
- Patch any exploited vulnerabilities
- Rebuild the system if needed (worse-case)

Why does this matter?

Containment stops the bleeding, but eradication prevents infection from returning.

Step 6: Close the alert ticket and summary

Closing the ticket isn't just an admin step, it ensures traceability, metrics for incident trends, and validated that the threat was handled according to policy.

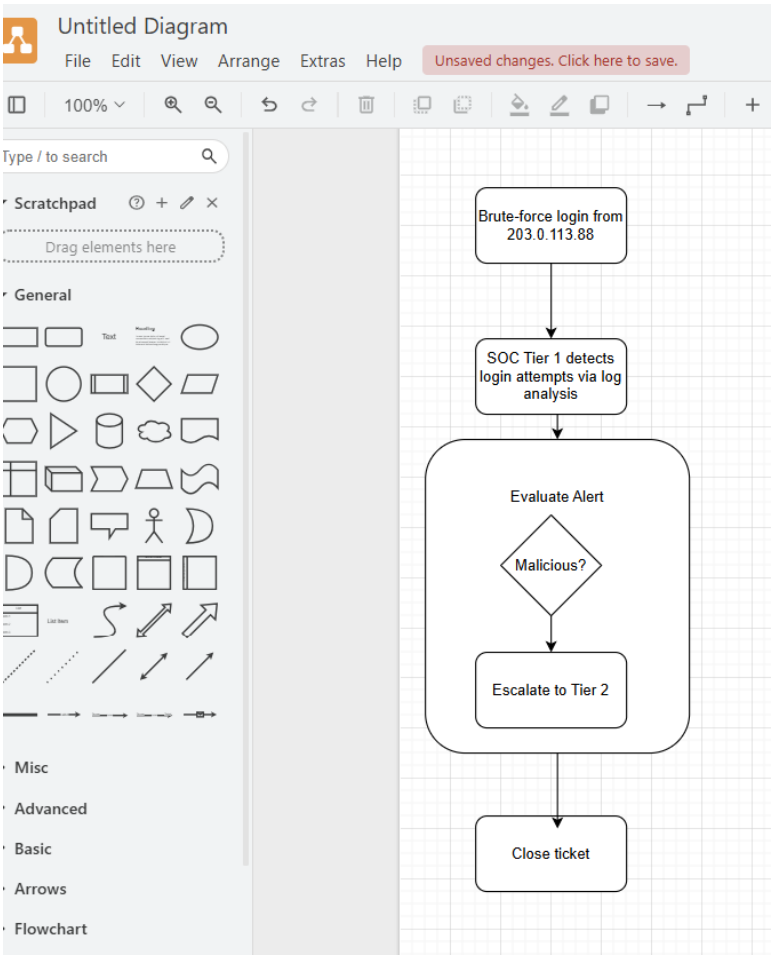
Based on the initial log review, I identified multiple failed login attempts from the external IP `203.0.113.88`, followed by a successful login and privilege escalation on the system `WIN-SQL01` using the account `jcarter`. I escalated the alert to Tier 2, provided the relevant Windows Events IDs (`4625`, `4624`, and `4672`), and notified the endpoint detection team to begin scanning for lateral movement.

Since I had escalated the incident and no further Tier 1 actions were required, I updated the ticket status to Closed on my end, with a note that the case had been formally handed off and is under active Tier 2 investigation. This follows protocol for Tier 1 scope: document, contain what you can, escalate appropriately, and then step back to monitor and log any new activity.

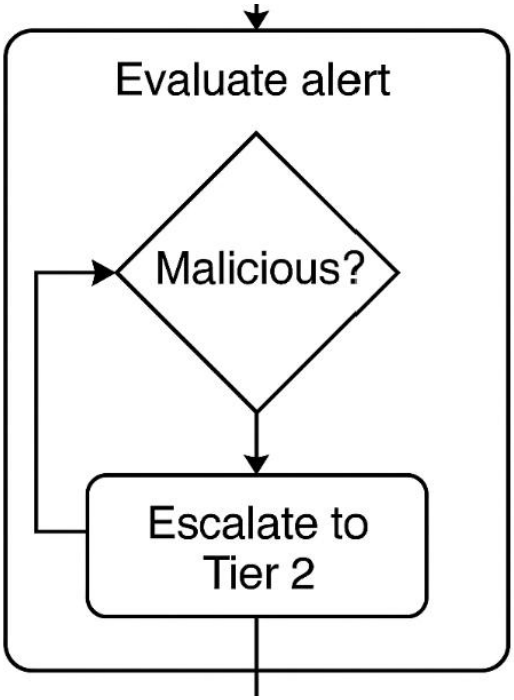
How I know the ticket can be closed:

The initial alert has been escalated, contained, eradicated, and reviewed. There's no more suspicious activity tied to this incident, and the user account and asset are secure.

Brute-force attack flow chart (Version 1.0)



I couldn’t figure out how to get the arrow inside the box to go up from Escalate alert to Malicious? So I had Chat do one for me. Below:


Evaluate alert
Malicious?
Escalate to
Tier 2