Beyond Login Attempts: *Detecting Threats in SaaS Applications*

Julie Agnes Sparks
Detection Engineer

Welcome!

Loving the blue team life:

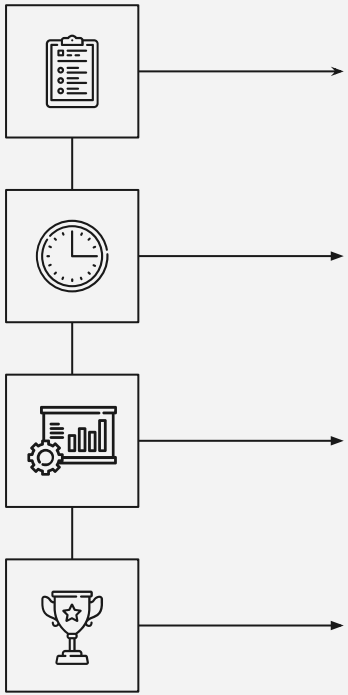
- Detection Engineering
- Incident Response
- Response Automation
- Log Ingestion
- Threat Hunting

Currently Security Research at Datadog,
Formerly doing D&R at Brex & Cloudflare

Fox

Bramble

Agenda



Detection Engineering 101

Logging & Data Quality

SaaS Application Attacks

Let’s Write Some Detections

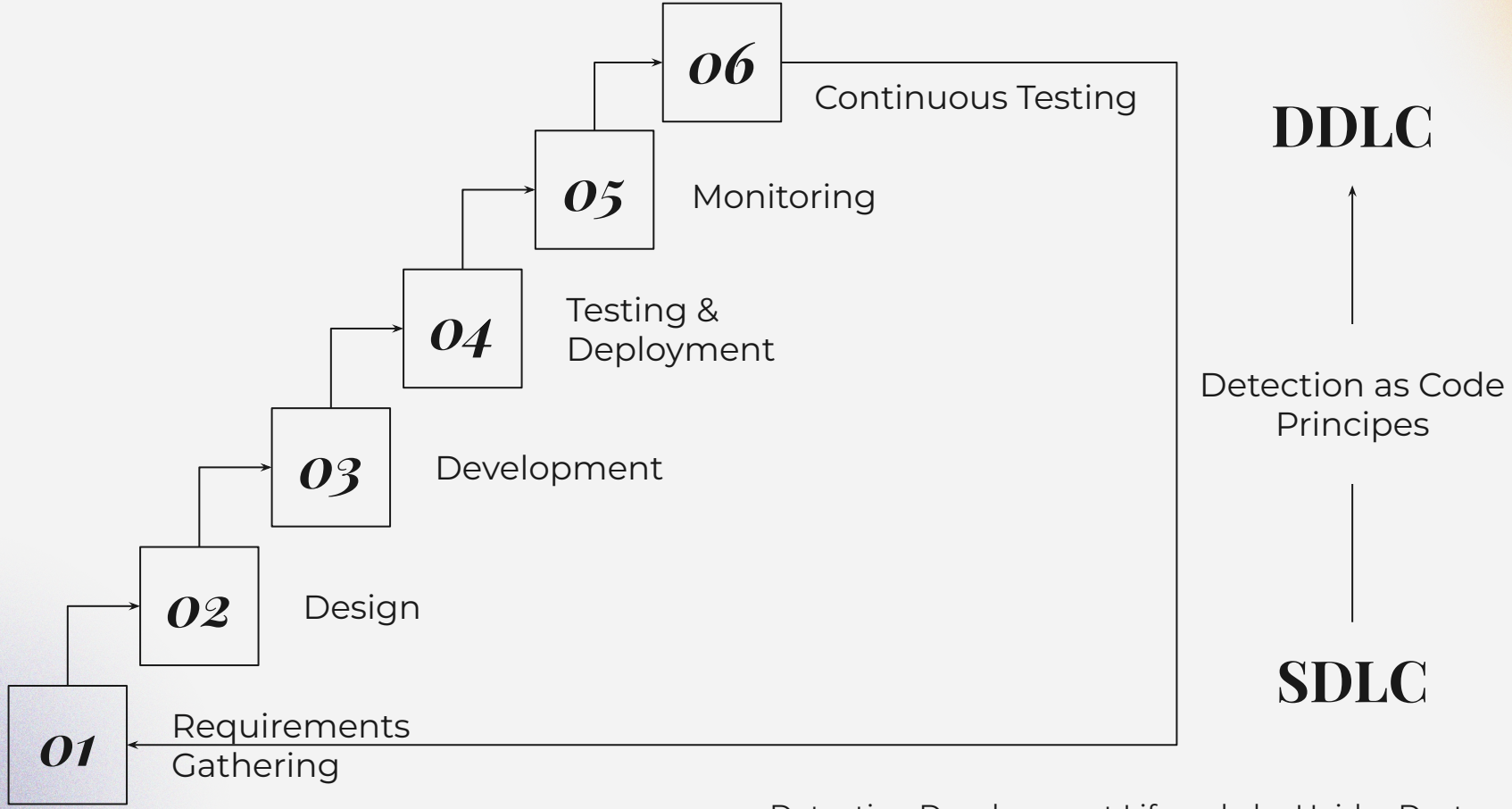
02

Detection Engineering 101

What is Detection Engineering?

- Focused on detecting threats in our environments based on logs created by a system/service.
- Can be approached in a similar way to the SDLC.
- Works alongside other security functions such as incident response, cloud security, and compliance.
- Shifted to focus on detecting techniques attackers use rather than relying solely on indicators of compromise (IOCs).

Detection Development Lifecycle



Detection Development Lifecycle by Haider Dost

Detection Engineering 101

Let’s go over some key detection types...

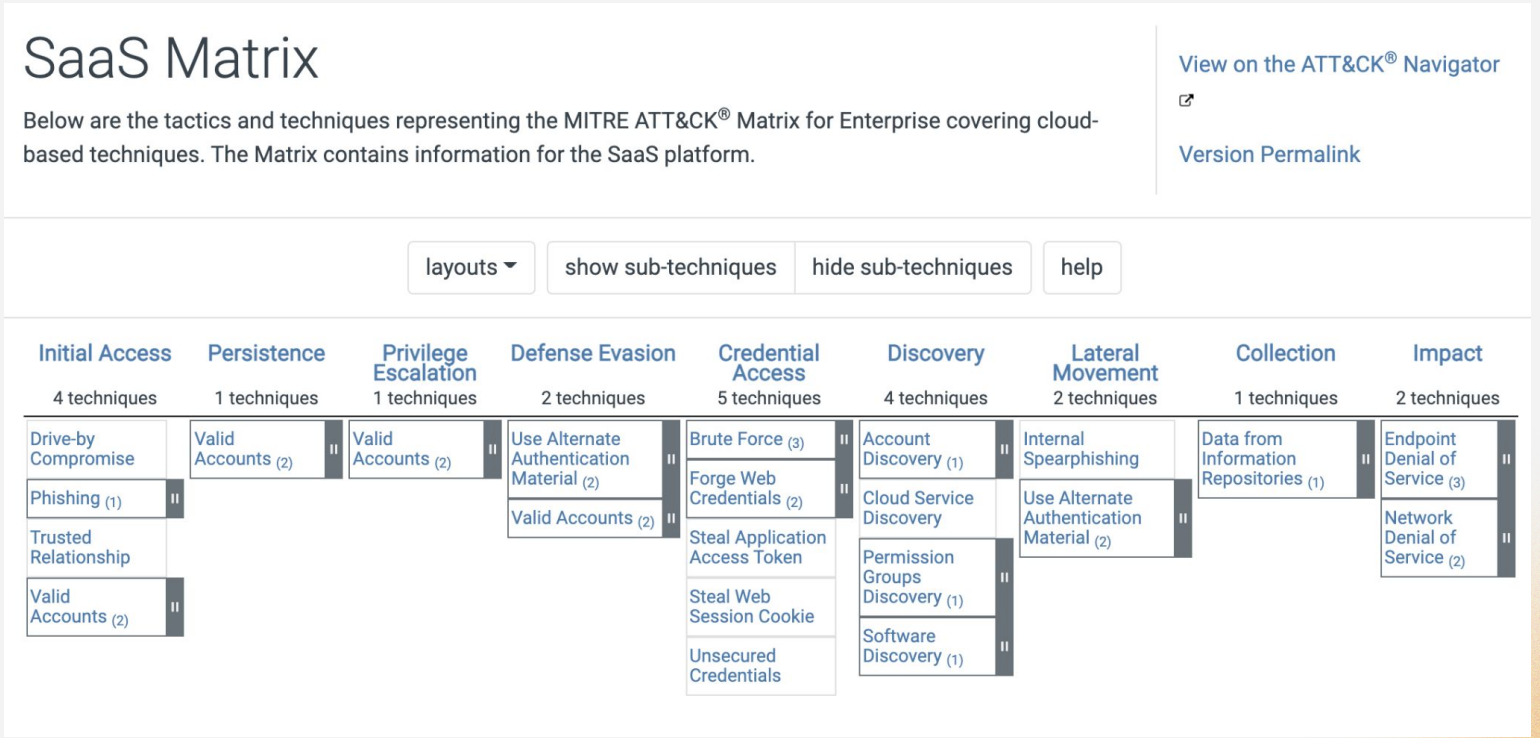
Single Event/Streaming

Multiple Event/Batch

Risk Based

Model Based

What tools do we have for guidance?



SaaS Matrix from MITRE

Reconnaissance	Initial Access	Execution	Persistence	Privilege Escalation	Defense Evasion	Credential Access
SAML enumeration	Consent phishing	Shadow workflows	API keys	Link backdooring	API keys	Password scraping
Subdomain tenant discovery	Poisoned tenants	OAuth tokens	OAuth tokens	Abuse existing OAuth integrations	OAuth tokens	API secret theft
Slug tenant enumeration	SAMLjacking	Client-side app spoofing	Evil twin integrations	Malicious mail rules	Evil twin integrations	
DNS reconnaissance	Account ambushing		Malicious mail rules		Malicious mail rules	
Username enumeration	Credential stuffing		Link sharing		Link sharing	

SaaS Matrix from Push Security

03 Logging & Data Quality

What logs can be available?

- User Activity
- API Activity
- Administrative Activity
- Integration Activity
- Authentication

User Activity Logs

- These logs capture user behavior such as access attempts, file downloads, creation of rules, and other user actions within the SaaS application.

Example: Github repository clone, Snowflake query, or Jira search

API Activity Logs

- These logs record API calls made to the SaaS application, including the user, source IP, API endpoint accessed, and data involved.

Example: Github API request to update a repository, Gitlab deployment of project via access token

Administrative Activity Logs

- These logs track administrative actions performed within the SaaS application, such as user creation, permission changes, or configuration modifications.

Example: Jira administrator created, Snowflake network policy for external IPs modified, etc.

Integration Activity Logs

- If your SaaS application integrates with other services, these logs track data exchange and activity related to those integrations.

Example: Snowflake client applications, Github third party application installed, etc.

Log Limitations

Lack of Log Content

Licensing & Cost

Poor Quality & Lack of Consistency in Formatting

Difficult Log Collection Mechanism

Lack of Log Content

Event types should cover all actions taken in the system and include critical fields, such as source ip address.

Audit logs should have external facing documentation on event types.

Logs should contain enough information to attribute activity to a user within the platform.

Poor Quality & Consistent Formatting

There should be log consistency across product versioning and operating systems, including when pulling the logs. Backwards compatibility introduced when needed.

There should be a low rate of log quality related incidents. Logs are reliable and can be taken as a source of truth.

There should be limited latency between when an action occurs and when the log event is available.

Difficult Log Collection Mechanisms

There should be the ability to stream logs to a cloud storage or SIEM provider (such as log push to S3).

Log collection should make it possible for event IDs to be sorted and straightforward in order to not miss log events or get duplicate events in the pipeline.

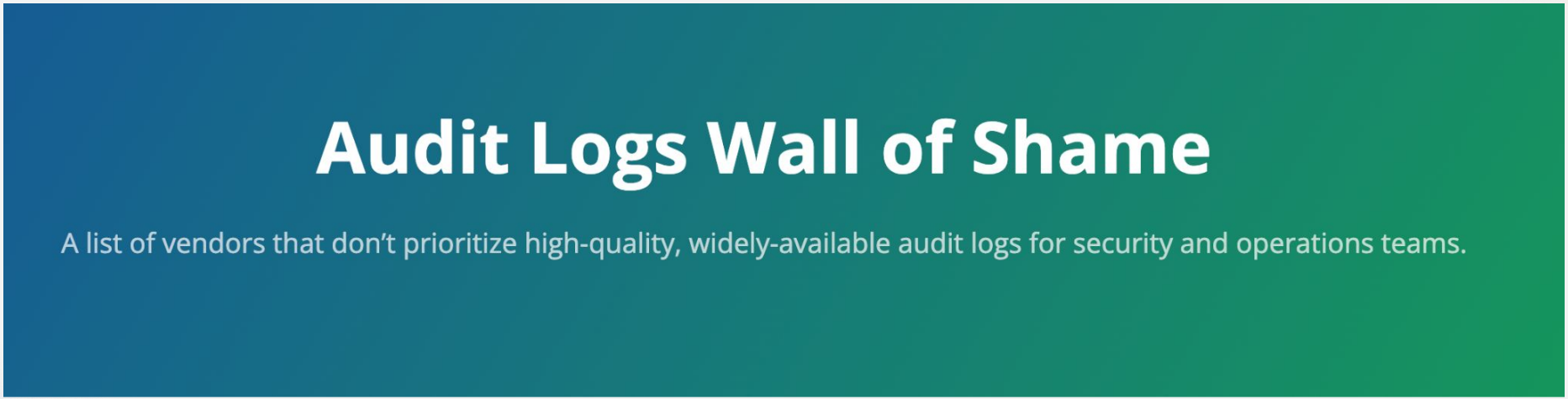
Good log formatting and data structure choice is critical, making it easy to parse logs once they are retrieved.

Licensing & Cost

There should be the ability to get detailed audit logs as part of the core product or reasonably priced.

The process to add audit logs to the contract should not be cumbersome.

Want to know more?



audit-logs.tax

Example Logging Issues for SaaS Apps

- No API collection

- Can only export logs via the UI, in batches of 500 messages per export.
- If you don't want to use the UI, you have to manually poll each machine in your environment to get the machines audit logs.

- Logs don't link to company email
- Standard logs don't include IPs
- Does not share all user activity logging with the enterprise organization
- Logged changes don't include both the new & old values

- Forces you to pay for the highest tier to stream events
- Does not include audit events for project settings, group settings, or deployment approval activity.
- The timezone used differs based on where you view audit logs (local time vs. UTC logged)

How can we make logs better ourselves?

Reference/Lookup Tables & Caching Data

Imagine... we had every IP address that checked in with our EDR provider in a table of lower risk device activity to reference our detections against.

Data Ingestion Cross Enrichment

Imagine... that same IP address is enriched into every other log source for that user to understand if they're accessing that application from a known location

03
SaaS Attacks & Detection

Detection Focus for SaaS

General Areas to Consider:

- Known bad patterns (Threat Research is your best friend)
- API activity
- User and service account pattern analysis
- Token usage
- Critical assets & data access

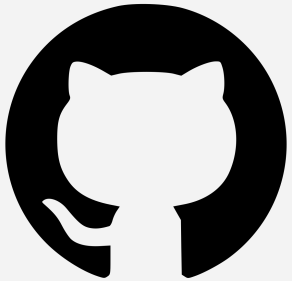
MITRE ATT&CK focus:

- Initial Access
- Persistence
- Collection
- Exfiltration

Inputs to Detection Engineering Research for SaaS

- Audit log documentation from the provider
- Past history of log data to use for hunting
- Research using current security articles and content
- Threat intelligence indicators

Let’s Focus on Two Cases



Github

Developer platform for Code Interaction & Storage

Snowflake

Cloud-based data storage and analytics service

Attackers Targeting Github
Attackers Targeting Snowflake

Google Security on Github Detection
Examining Github Security
Github Security Guide

Github Log Visibility

- Github has GA attribution of associated user email addresses to activities in audit logs.
- They allow the ability to include source_ip address in logs.
- Github provides granular detail on type of token taking the action, such as:
    - Personal Access Token (Regular or Fine Grained)
    - OAuth Access Token
    - Server to Server Access Token
    - User to Server Access Token
- There's now Github API request logs that provide granular usage of tokens to take actions via API.

Github token formats & usage

Github Threat Actors

Various Groups →

Malicious Payload Delivery & Packages

Github has been used to host and deliver malicious payloads and act as dead drop resolvers, command-and-control, and data exfiltration points.

ShinyHunters & Various Groups →

Credential Theft & Data Exfiltration

Compromising user accounts through credential theft and then exfiltrating data, stealing further access keys, or ultimately extorting the company

Threat Actors & Github Gitloker Extortion

Github Detection Focus

Abnormal Usage of Access Tokens

Repository Cloning Behaviors

Utilizing Stolen Credentials

Persistence through New User Accounts

History of Access Token Usage

April 2022

GitHub Security announced it had detected the compromise of OAuth access tokens issued to Heroku and Travis-CI integrations to download data from dozens of organizations.

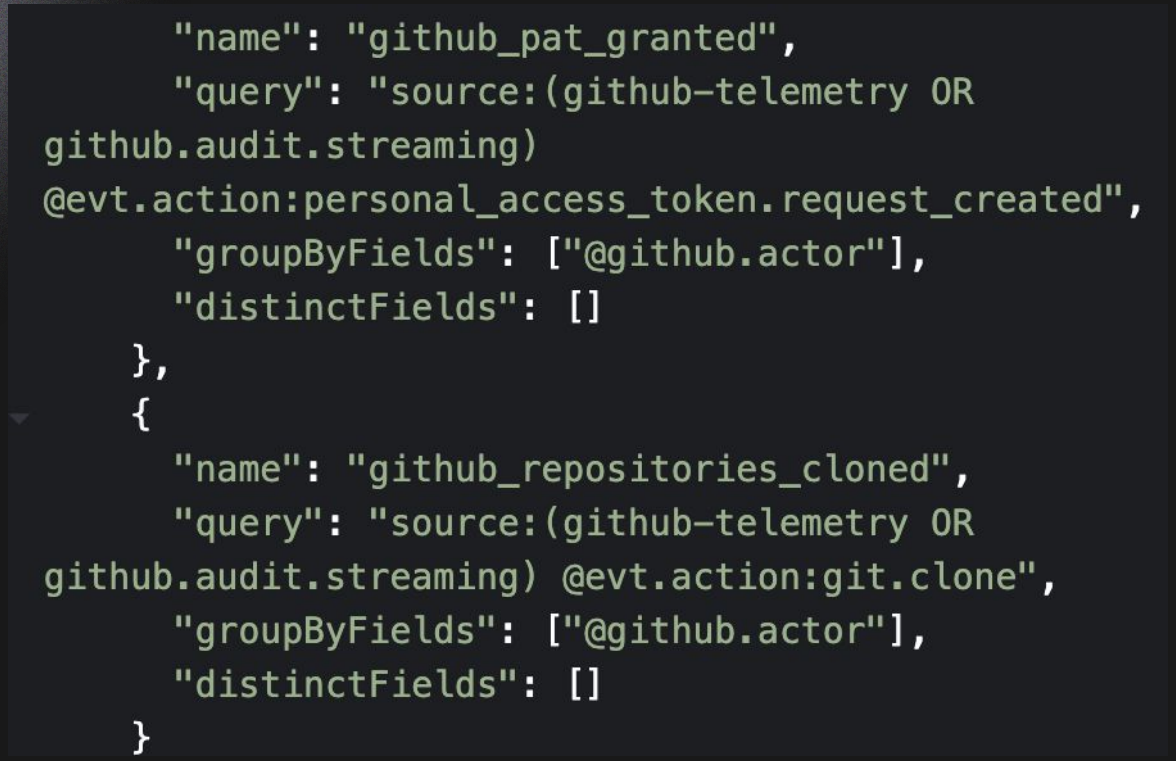
October 2023

In another compromise, organizations found that attackers accessed their Github accounts using compromised PATs (Personal Access Token) – most likely exfiltrated silently from the victim's development environment.

```
query: |
  @github.actor AND @network.client.asn AND
@http.useragent
aggregation: |
  group_by(@github.actor),
count_distinct(@network.client.asn) >= 2 AND
count_distinct(@http.useragent) >= 2
```

Github OAuth Token Actions taken by Various ASNs and UAs

Okta Session Hijacking Research



GitHub Personal Access Token created and used to repo clone

```
query:"
event_dataset=github
event_action="api.request"
http_request_method = "GET"
| where url_path contains ("/secrets/")
| where !(url_path contains
("/repositories/:repository_id/actions/secrets/public-key"))"
```

Github API Request to Secrets Path

Additional Detection Ideas

- Personal Access Token generated and cloning of repositories
- Abnormal OAuth access token usage
- SSH Key Created from Suspicious IP Address
- Private repository changed to public
- User downloaded data as zip file
- Unknown user cloned private repository
- Non-company email joining Github Org

Snowflake Threat Actors

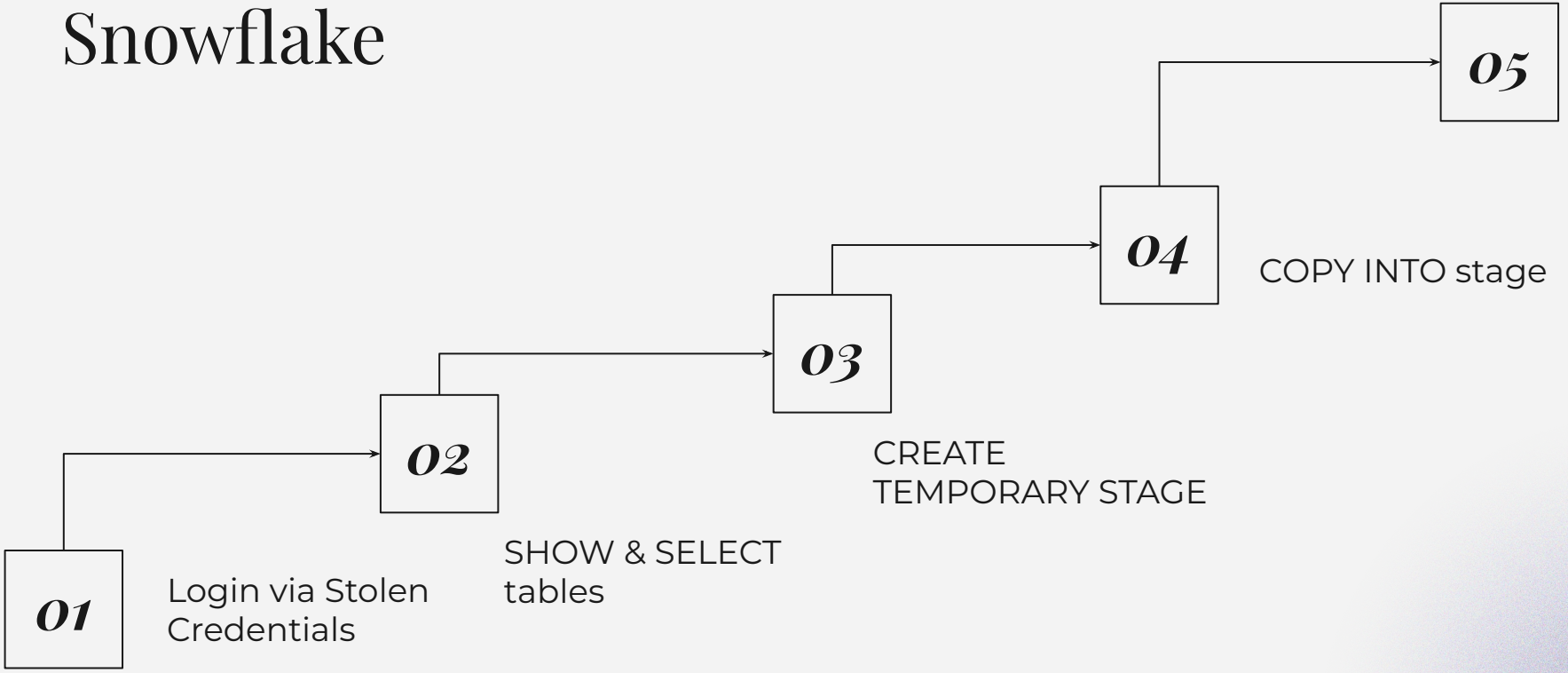
UNC 5537

Active 2024 - Present

Infostealer Malware Used to Gain User Account Access

A financially motivated threat actor suspected to have stolen a significant volume of records from Snowflake customer environments

UNC 5537 behaviors in Snowflake



Snowflake Attack Techniques

Client Applications

Exfiltration Behaviors

Malicious IP Addresses

Data Transfer Actions

History of Exfiltration

Early 2024

Snowflake announced compromise of 200+ customers that had data exfiltration performed by UNC-5537.

TBD

When will the next Snowflake data extortion be?

```
query: "source:snowflake snowflake.table:stages
@stage_action:CREATED (@stage_url:s3* OR @stage_url:gcs* OR
@stage_url:azure*)",
groupByFields: ["@stage_url"],
```

Snowflake stage set to anomalous external cloud location

```
"query": "source:snowflake snowflake.table:query_history",
"groupByFields": ["@usr.name"],
"distinctFields": ["@database.name"],
"aggregation": "cardinality"
```

Snowflake user anomalously querying data

```
SELECT *,
FROM
 snowflake.account_usage.query_history
WHERE
 CONTAINS(QUERY_TEXT, 'COPY INTO') AND
CONTAINS(QUERY_TEXT, 'http')
 AND START_TIME >= dateadd('minutes', -7,
current_timestamp())
LIMIT 100;
```

Snowflake copying of data into external location

Additional Detection Ideas

- New Client Application Authorized for Snowflake Instance
- Grants of Administrator role to User
- Network Policy Modified to Allow External IPs

If you want to read more about threat hunting in Snowflake..

EMERGING THREATS AND VULNERABILITIES

A guide to threat hunting and monitoring in Snowflake

June 7, 2024

THREAT DETECTION

Thanks!

Do you have any questions?

Reach out to me on LinkedIn or after the talk.

https://www.linkedin.com/in/julie-a-sparks/

Check out a list of OOTB detections from here and here.

Read more about Github Detections in a previous BSidesLV talk, here.