

Cyber Deception in GCP with Generative Traps

BSidesCharm 2025

Matt Maisel

“

Defenders think in lists.
Attackers think in graphs.
As long as this is true, attackers win.

John Lambert, 2015

It's 2025.

It's 2025.

Defenders think in graphs.

It's 2025.

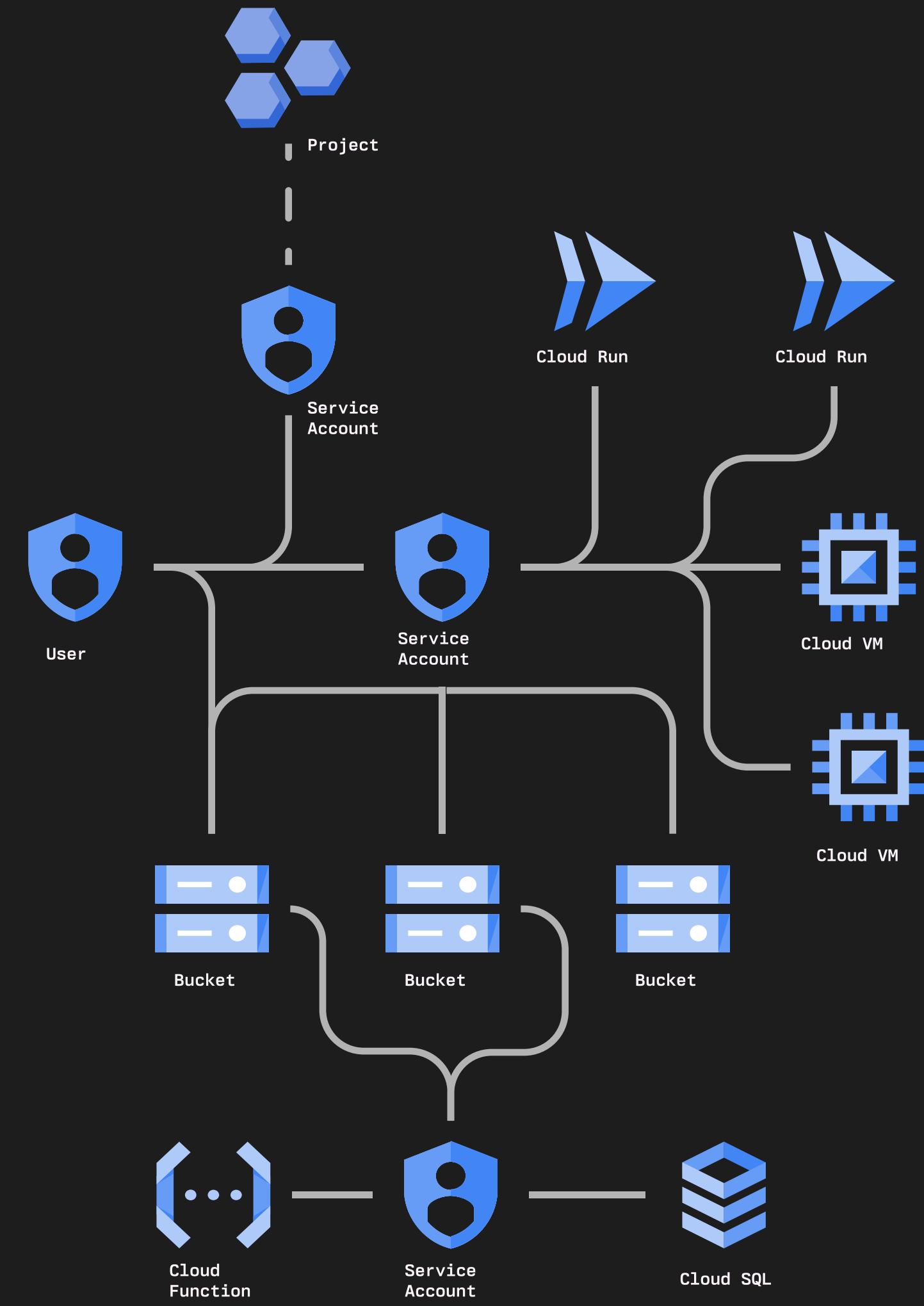
Defenders think in graphs:
so do security tools and vendors.

It's 2025.

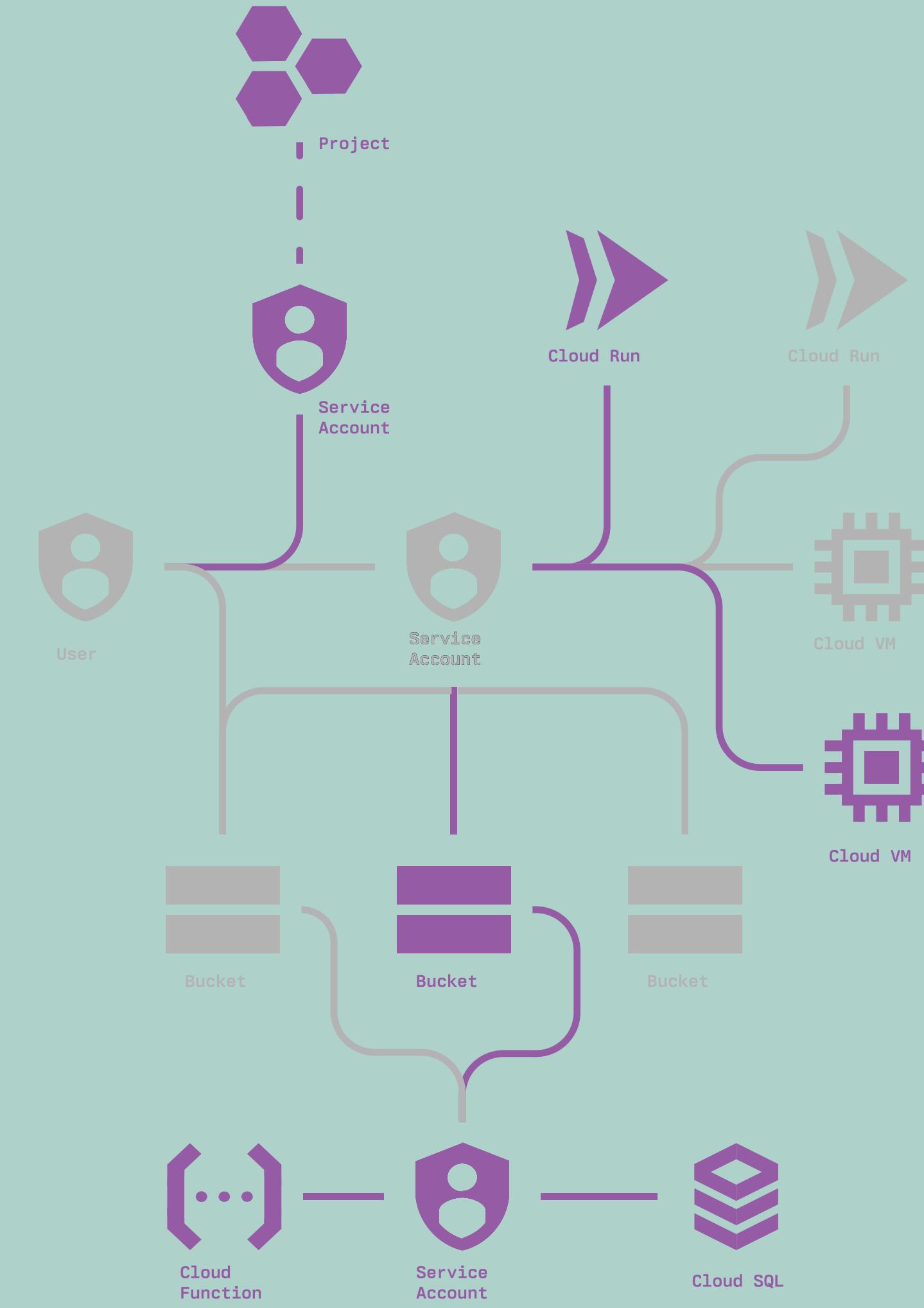
Defenders think in graphs:
so do security tools and vendors.

But attackers still win.

So what now?



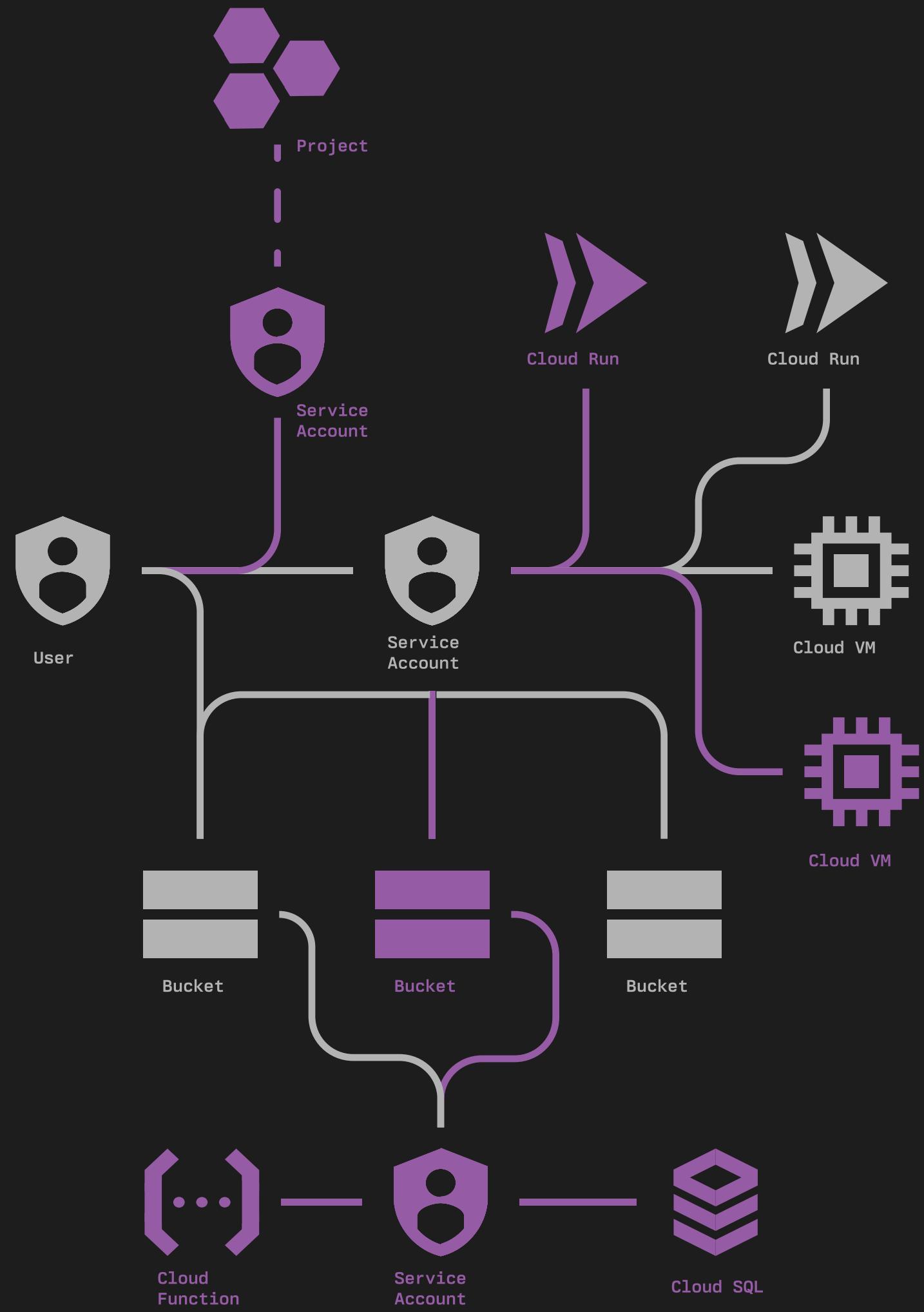
We lie to them.



Reinforce the seams in detection surfaces.

Cyber Deception is a ruse to mislead or disrupt adversaries by exploiting their cognitive biases.

Traps detect adversary interaction and alert SecOps.



Old idea.

History

~35 years since The Cuckoo's Egg
(1989)

MITRE Engage for adversarial
engagement

Security Canary Maturity Model

Old idea; new tools.

History

~35 years since [The Cuckoo's Egg \(1989\)](#)

[MITRE Engage](#) for adversarial engagement

[Security Canary Maturity Model](#)

Tools

100+ tools listed on [awesome-honeypot](#) and [awesome-deception](#)

[HoneyTrail](#) uses CloudTrail with AWS honeypot resources.

[Koney K8s Operator](#) honeytokens and fake API endpoints.

LLM-based honeypot frameworks like [Galah](#), [GenAIPot](#), and [VelLMes](#).

[SAP Cloud Active](#) deploys decoys to application layer.

Old idea; new tools--- not dead.

History

~35 years since [The Cuckoo's Egg \(1989\)](#)

[MITRE Engage](#) for adversarial engagement

[Security Canary Maturity Model](#)

Tools

100+ tools listed on [awesome-honeypot](#) and [awesome-deception](#)

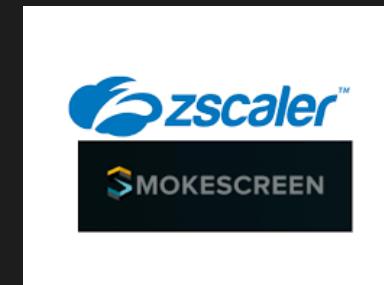
[HoneyTrail](#) uses CloudTrail with AWS honeypot resources.

[Koney K8s Operator](#) honeytokens and fake API endpoints.

LLM-based honeypot frameworks like [Galah](#), [GenAIPot](#), and [VelLMes](#).

[SAP Cloud Active](#) deploys decoys to application layer.

Vendors



Why don't we have more
deception-based detections?

“

Within hours of sending the letter,
Deel's spy inside of Ripple searched -
for the first time - for this empty and
never-before-used Slack channel...

[Lawsuit Alleges \\$12 Billion "Unicorn" Deel Cultivated Spy, Orchestrated Long-Running Trade-Secret Theft & Corporate Espionage Against Competitor, 2025](#)

Why don't we have more
deception-based detections?

No one talks about it?

“

By 2030, 75% of organizations will have deployed deception capabilities in their enterprise environments, up from less than 10% today.

Gartner, 2025

1. Questionable Effectiveness

37% fell for traps, reduce
finding true risk by 22%

Don't be too obvious or camouflaged.

Imitate true risks.

Honeyquest (2024)

1. Questionable Effectiveness

37% fell for traps, reducing
finding true risk by 22%

Don't be too obvious or camouflaged.
Imitate true risks.

Honeyquest (2024)

100% triggered decoy
alert

More decoy alerts when **informed** vs
informed!

Tularosa Study (2021)

1. Questionable Effectiveness

37% fell for traps, reduce
finding true risk by 22%

Don't be too obvious or camouflaged.

Imitate true risks.

Honeyquest (2024)

100% triggered decoy
alert

More decoy alerts when **informed** vs
informed!

Tularosa Study (2021)

> 0.5% network, increase
attack path

"it would completely break all of my
workflows"

"I might give up and find a softer
target."

"...you're just going to have to be slower.
More methodical."

Reeves and Ashenden (2025)

2. Distinguishable from Targets

Reputation of honeypots

honeydet and nuclei templates

AWS canary tokens shared the same
account ids.

3. Tedious and Risky Infrastructure

Who does the book keeping?

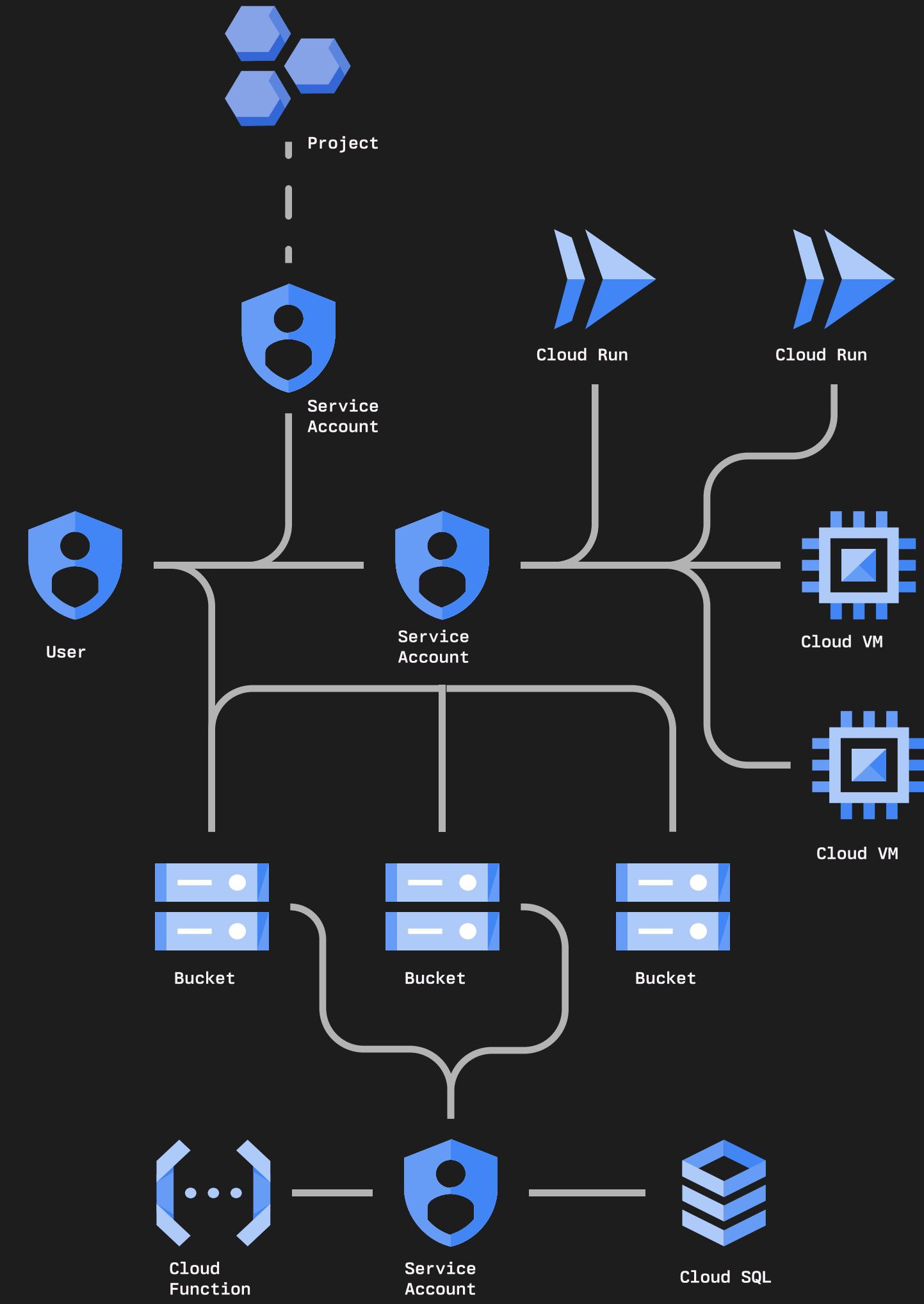
What if deception introduces risk
or disrupts production?

“

What held Deception back
was not the concept, but
the technology.

Lance Spitzer, 2019

So what now?



Paltergeist

is a cyber deception tool
for **generation**,
orchestration, and
monitoring of
cloud-native traps
that lure and detect
attackers.

**Generates
tailored,
believable
traps.**

Don't craft by hand anymore.

Sample your environment.

Apply n-shot learning, LLM-as-a-judge, etc.

Orchestrates trap infra & monitoring.

Programmatic IaaC with Pulumi
Automation API

“Scale to zero”

Shared infrastructure lifecycle

Monitoring and observability

Manages deception engagements.

Groups traps with related targets.

Stratagems are templates for traps:

1. What use case?
2. Where in the attack path?
3. What do they masquerade as?

Follow The Yellow Brick Road.

Expose

Lateral Movement, Privilege
Escalation

Valid Cloud Accounts,
Credentials

Save Your Pets.

Expose or Elicit

Shoot Your Cattle.

Discovery, Exploitation,
Persistence

Watch Your Canaries.

Compute traps with
preconfigured vulnerabilities
or weaknesses.

Crown Jewel Gravity Well

Expose

Collection, Exfiltration

Storage traps

Enter My Cloud Labyrinth.

Elicit TTPs

Multiple stratagems

Project traps!

2025/04/10 11:08:05 INFO initialized provisioner provisioner="Provisioner{projectId: nomaladies-health-11479c5, paltergeistProjectId: paltergeist-55a9e0a, stackFullyQualifiedNames: [mmaisel/nomaladies-health/app]}"

NAME:

paltergeist - A new cli application

USAGE:

paltergeist [global options] [command [command options]]

COMMANDS:

deploy Deploy the paltergeist resources to the target stacks
destroy Destroy the paltergeist resources from the target stacks
help, h Shows a list of commands or help for one command

GLOBAL OPTIONS:

--help, -h show help

maisel:paltergeist/ (mainx) \$./paltergeist deploy

[11:08:05]

Deployment Demo

Logging

Logs Explorer

Share link

unavailable

→

<

Actions

More

0 results

Actions

More

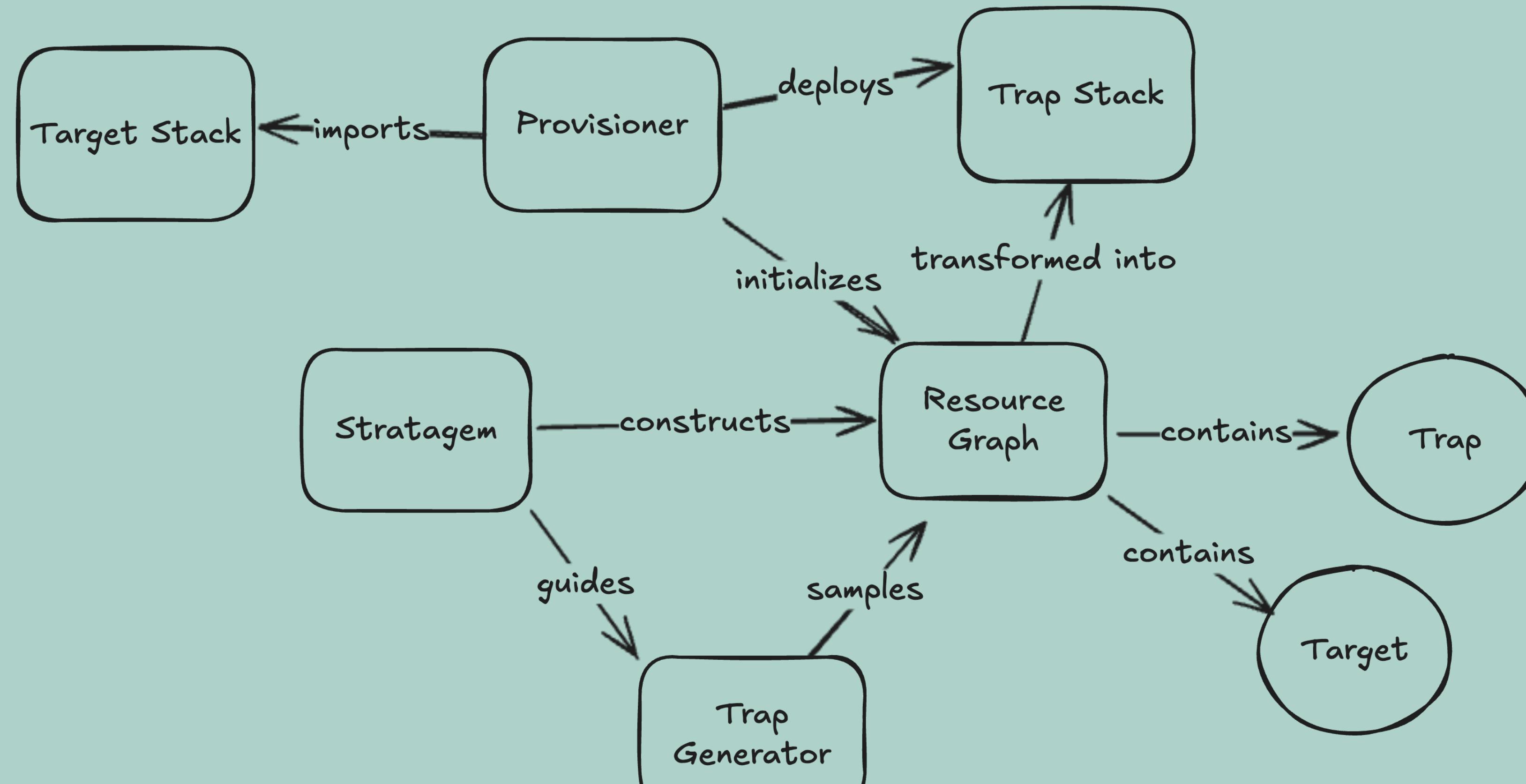
SEVERITY TIME SUMMARY

Streaming logs...

>	i	2025-04-10 11:39:44.308	storage.googleapis.com	storage.objects.list	projects/_/buckets/nomadadies-logs/objec
>	i	2025-04-10 11:38:37.041	storage.googleapis.com	storage.objects.get	_s/nomadadies-logs/objec
>	i	2025-04-10 11:38:33.979	storage.googleapis.com	storage.objects.list	projects/_/buckets/nomadadies-logs/objec
>	i	2025-04-10 11:38:30.833	iamcredentials.googleapis.com	GenerateAccessToken	_/-/serviceAccount
>	i	2025-04-10 11:38:28.148	iamcredentials.googleapis.com	GenerateAccessToken	_/-/serviceAccount
>	i	2025-04-10 11:36:24.612	iamcredentials.googleapis.com	GenerateAccessToken	_/-/serviceAccount
>	i	2025-04-10 11:24:14.702	iamcredentials.googleapis.com	GenerateAccessToken	_/-/serviceAccount
>	i	2025-04-10 11:24:10.615	storage.googleapis.com	storage.objects.get	_s/nomadadies-logs/objec
>	i	2025-04-10 11:24:08.352	storage.googleapis.com	storage.objects.list	projects/_/buckets/nomadadies-logs/objec
>	i	2025-04-10 11:24:04.535	storage.googleapis.com	storage.objects.get	_s/nomadadies-logs/objec
>	i	2025-04-10 11:24:00.977	storage.googleapis.com	storage.objects.list	projects/_/buckets/nomadadies-logs/objec
>	i	2025-04-10 11:23:54.657	iamcredentials.googleapis.com	GenerateAccessToken	_/-/serviceAccount
>	i	2025-04-10 11:23:11.505	iamcredentials.googleapis.com	GenerateAccessToken	_/-/serviceAccount
>	i	2025-04-10 11:23:08.500	iamcredentials.googleapis.com	GenerateAccessToken	_/-/serviceAccount
>	i	2025-04-10 11:22:22.729	iam.googleapis.com	_iam.admin.v1.ListServiceAccountKeys	_/-/serviceAccount
>	i	2025-04-10 11:22:22.683	iam.googleapis.com	_iam.admin.v1.ListServiceAccountKeys	_/-/serviceAccount
>	i	2025-04-10 11:22:22.646	iam.googleapis.com	_iam.admin.v1.ListServiceAccountKeys	_/-/serviceAccount
>	i	2025-04-10 11:20:57.267	storage.googleapis.com	storage.objects.get	_s/nomadadies-logs/objec
>	i	2025-04-10 11:20:53.120	storage.googleapis.com	storage.objects.get	_s/nomadadies-logs/objec

Trap Monitoring Demo

Engagement



Paltergeist Components

Dynatrace Research, 2020

Where do we go from here?

Optimize placement.

Maximize discoverability,
minimize costs.

Minimize exposure
to production resources.

Where do we go from here?

Optimize placement.

Maximize discoverability,
minimize costs.

Minimize exposure
to production resources.

Build benchmark evals.

How do we measure the effectiveness
of generated traps?

1. Discoverability
2. Indistinguishability
3. Deployability

Where do we go from here?

Optimize placement.

Maximize discoverability,
minimize costs.

Minimize exposure
to production resources.

Build benchmark evals.

How do we measure the effectiveness
of generated traps?

1. Discoverability
2. Indistinguishability
3. Deployability

Expand coverage.

More strategems

K8s, AWS, Azure

“

*Everyone thinks in graphs.
Defenders think in game theory.
Attackers win— or did they?*



<https://github.com/mmaisel/paltergeist>