OPEN-SOURCING AWS PENTEST METHODOLOGY

$AWS STS GET-SPEAKER-IDENTITY

Lizzie Moratti (@MorattiSec)

- **Professional Background**
  - Former PM @ Rhino Security Labs (2018)
  - Security Consultant @ Leviathan Security Group
- **Free time**
  - Blogposts & Blogpost Accessories
    - AWS Security research & AWS Community Builder
    - TunnelVision CVE Coauthor (May 2024)

DISCLAIMERS

- My thoughts and opinions are my own
- Customer side of the shared responsibility model

OUTLINE

1. What is a pentest methodology
2. Prior methodologies
3. My AWS pentest methodology
4. The limitations of my methodology
5. The future of AWS pentest methodology

CALL TO ACTION

Help contribute to my project!

- Other Pentesters: Writing and structure
- General: Find a place to host/maintain it

WHAT IS A PENTEST METHODOLOGY?

Definition:

A standardized framework that guides pentest professionals through the process of identifying, exploiting, and reporting ***vulnerabilities*** *or* ***misconfigurations*** *in an environment in a repeatable way.*

Goals of a methodology:

- Holistically cover attack surfaces and identify risks
- Systematically and efficiently approach an environment
- Consistently achieve similar results between practitioners
- **Obtain as much context as possible**

PRIOR METHODOLOGY

Cloud Security Alliance Penetration Testing Playbook (2019)

https://cloudsecurityalliance.org/artifacts/cloud-penetration-testing-playbook

The good

- Ahead of the curve
- Focuses on threat modeling
- Provides resources to learn from

4. Testing

a. Validating baseline security requirements
b. Employ security test cases, guides and checklists relevant to domain & technologies
   - web? mobile? mative? serverside? API?
   - c# mvc? objective c IOS? Python redhat? c++ winforms
c. Test for Spoofing of user identity and other entities
   i. Steal hardcoded serverless workloads function (a workload implemented as a function) credentials and secrets (like hardcodedAzure function code or by pulling a lambda deploy package)
   ii. Attempt load balancer MiTM for session hijacking (elb) by cloud service configuration or load balancer instance compromise
   iii. Attempt domain transfer to another registrar for domains not transfer prohibited (Route53, aka domain hijacking)

THE ROOM FOR IMPROVEMENT

- It does not explicitly consider AWS Organizations/ SCPs
- Largely focused around testing a single account
- Includes many resources for external enumeration

PRIOR METHODOLOGY

SecWiki.cloud

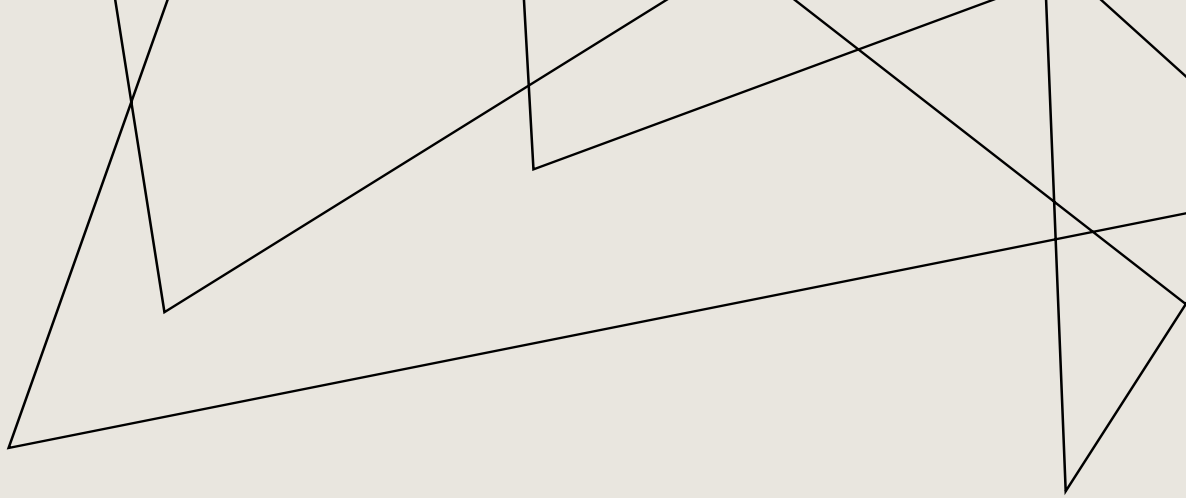
https://www.secwiki.cloud/aws/assessment-guide

The good

- Specific to AWS
- High level bullets for general things to check for
- Service specific subguides

General Approach

- Book a kick off call, meeting or similar with the project team for them to run through the architecture with you, highlight any expected privileged IAM accounts and what they're for, get details on any particular concerns they have. This should ideally be the first thing you do.
- Run the automated audit tools - Scoutsuite and Prowler
- Run cloudmapper, first in collect mode and then in audit mode. It provides a useful snapshot of some of the environment if we have to come back to it later
- Do an IAM user/role/policy review
  - Best to do this early, to leave time to talk through your results with project team if needed as a lot of this is contextual.
  - Tools that can help with this:
    - PMapper
    - Cartography
    - awspx



THE ROOM FOR IMPROVEMENT

- Does not mention AWS Organizations or SCP’s
- Maintenance
- More for a cloud configuration review rather than a pentest

SUMMING UP THE PROBLEMS:

- AWS orgs/ multi-account architectures
- Practical workflow
- Maintenance
- Relevant context gathering

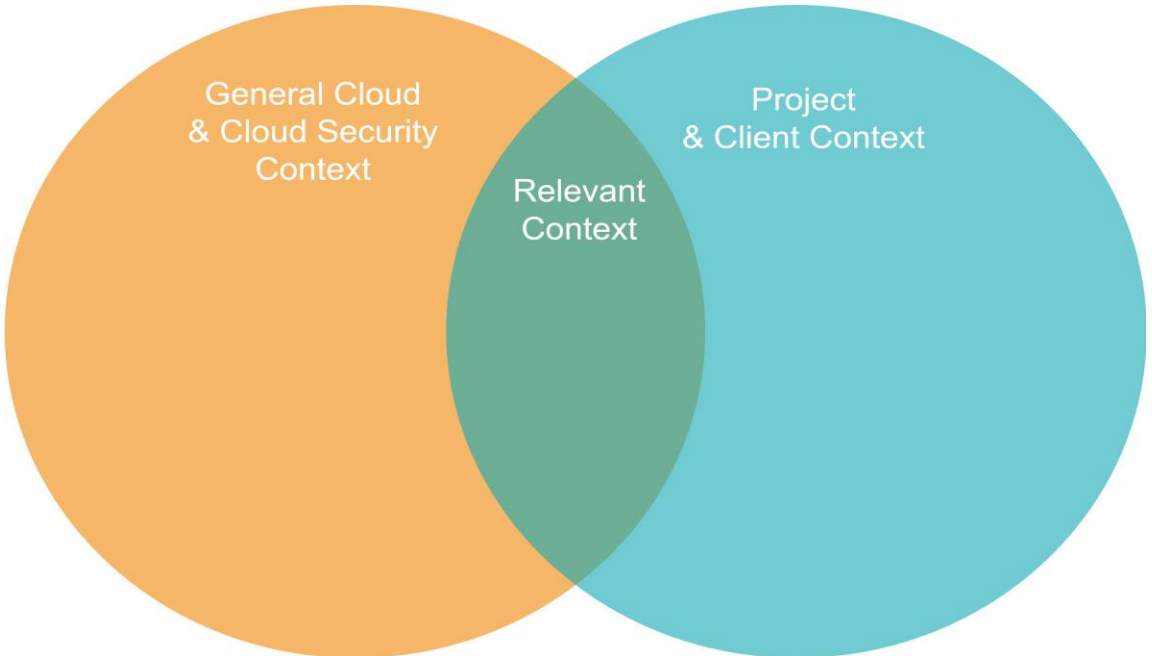
THE MOST VALUABLE THING TO A CLOUD PENTESTER

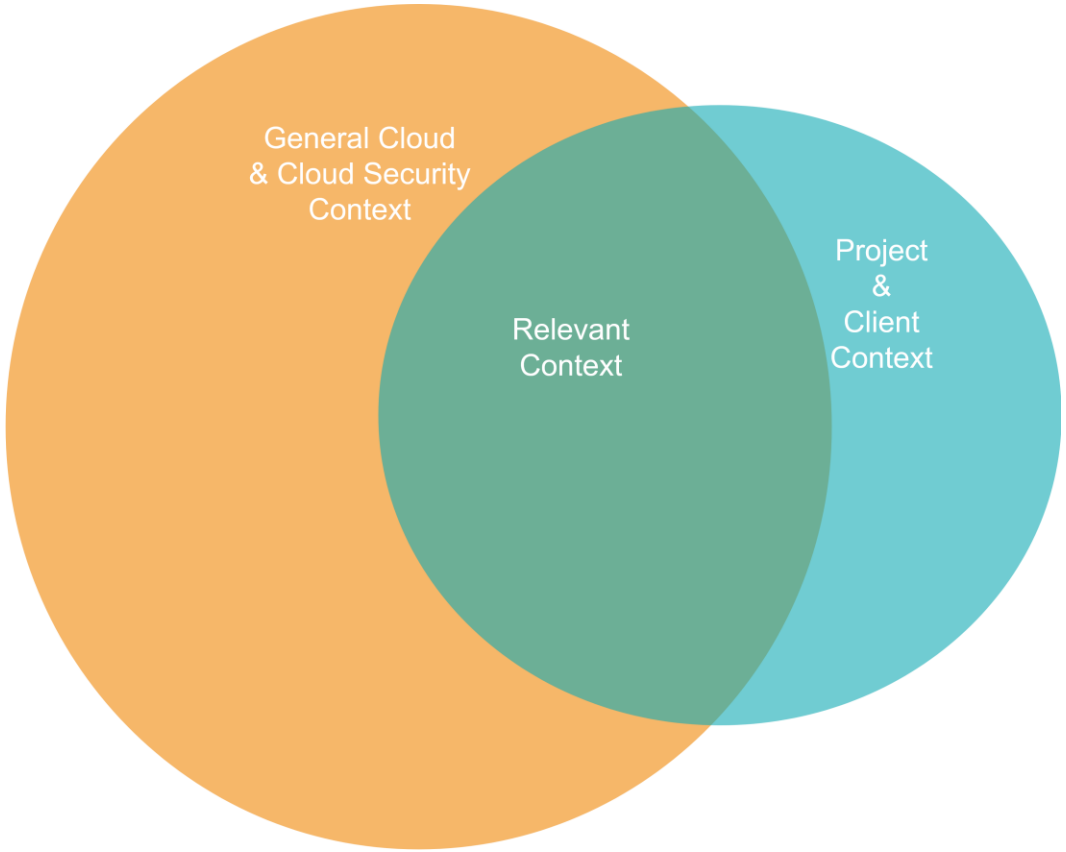
Context.
The more context you have the more thorough and precise you can be in *identifying* **true issues**, determining **severity**, and making good **recommendations**. The more the better.

CONTEXT IS A BUZZWORD

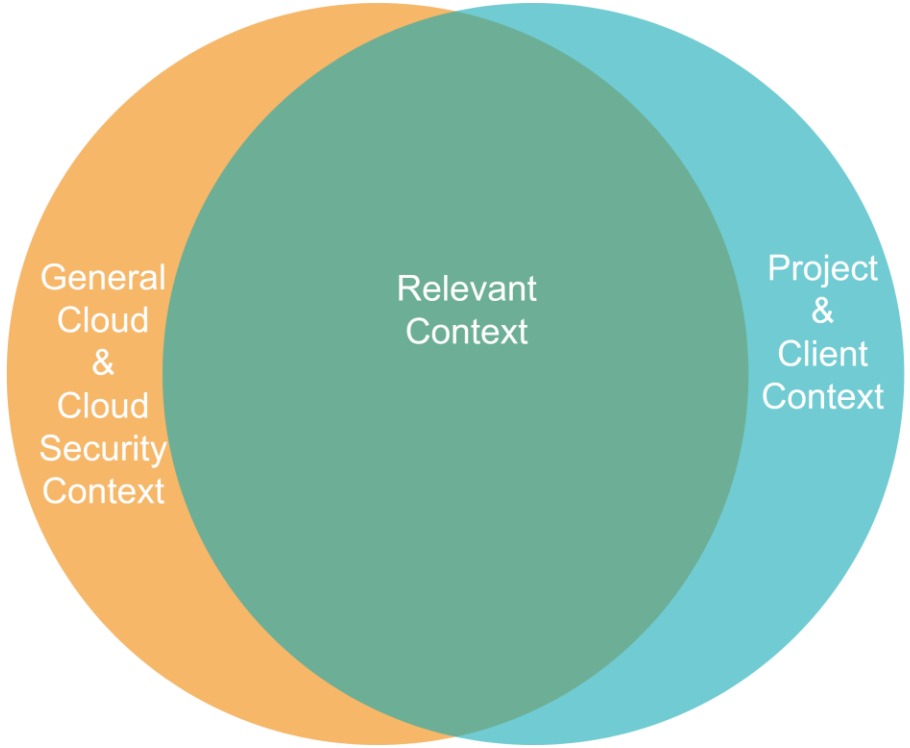
SYSTEMATICALLY OBTAINING RELEVANT CONTEXT

- General Cloud & Cloud Security Context
  - ”I know about IAM”
  - ”I know about IAM Privesc”
- Project & Client Context
  - “The scope is 3 accounts”
  - “We don’t care about CIS benchmarks unless they actually have risk”

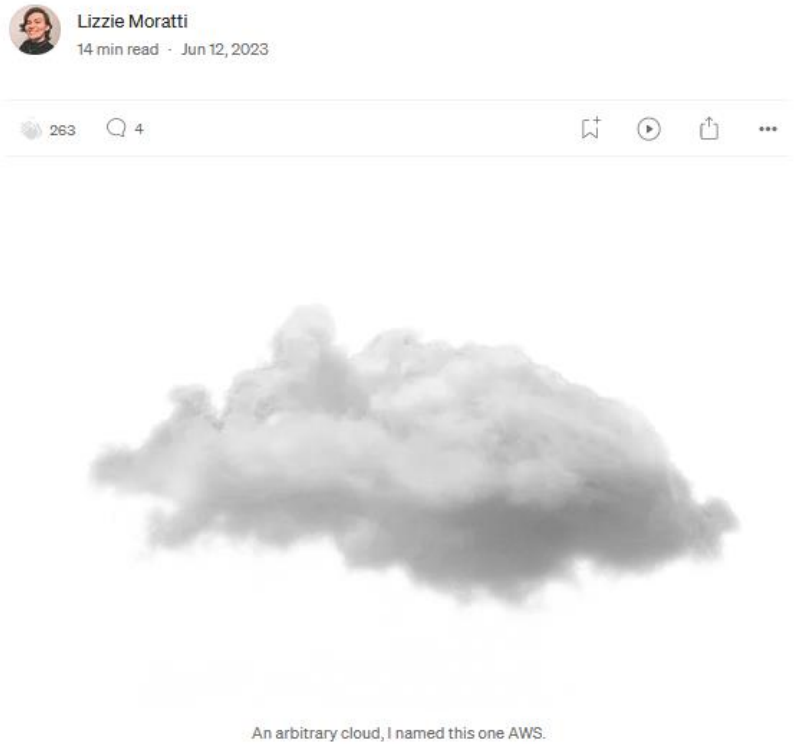




Growing your own knowledge of cloud & cloud security



Growing your project & client context



https://medium.com/@MorattiSec/

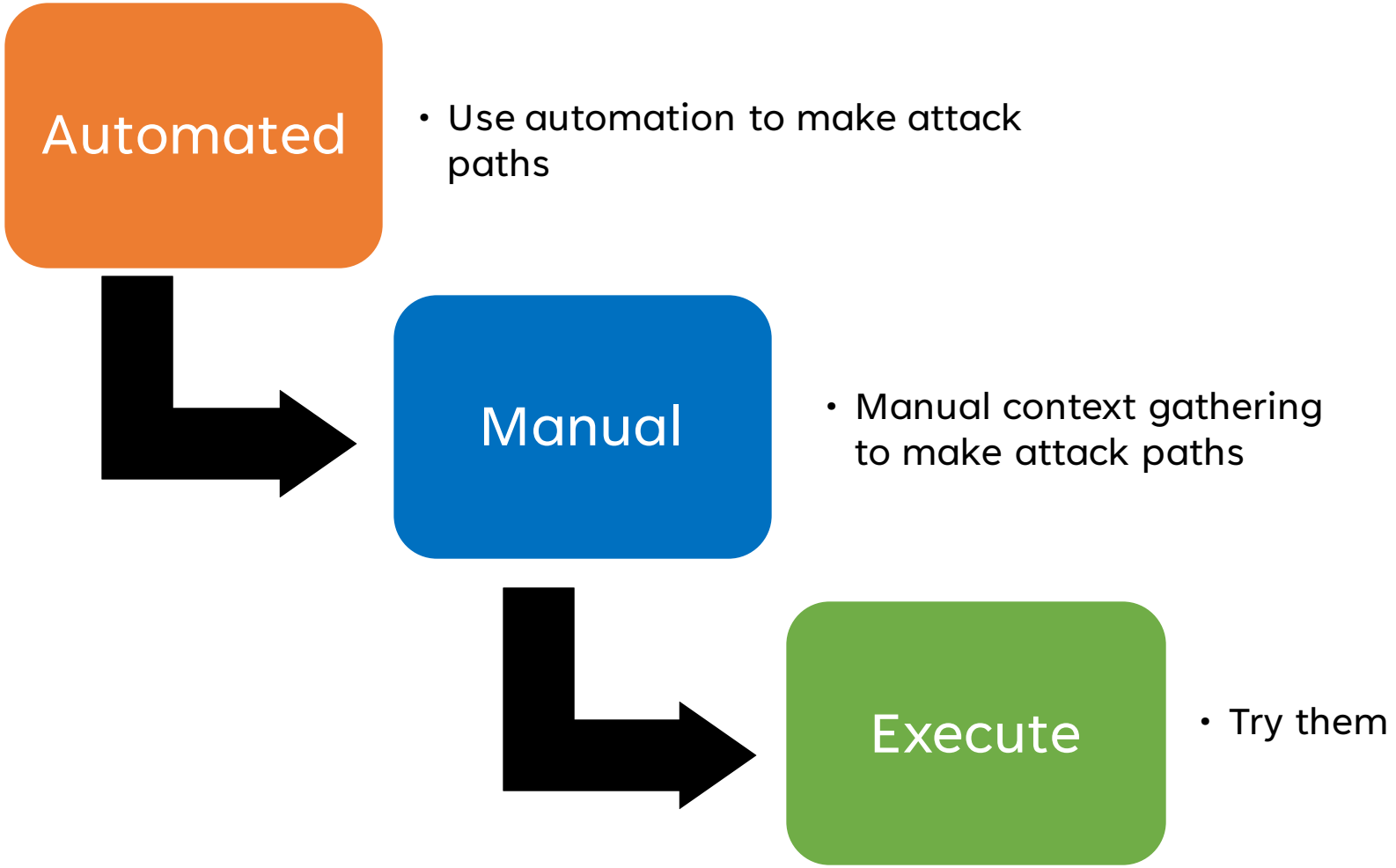
MY AWS METHODOLOGY

How hard could this be?

- 6 months to make something 80% ready
- Published my own on my Medium blog
- It turns out making definitive statements about anything is fraught with peril and "well actually..."
  - https://blog.plerion.com/things-you-wish-you-didnt-need-to-know-about-s3/

High-quality images will help convert your online leads by: Increasing user engagement — which means readers spend more time on your article and are more likely to check out your other content.

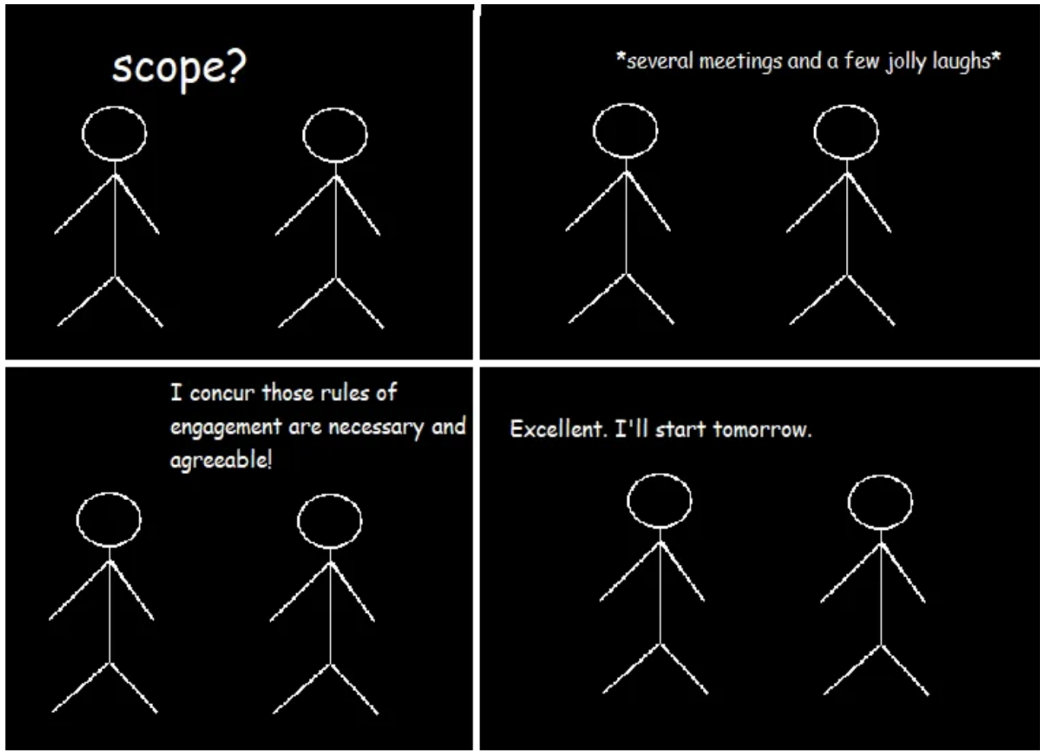
THE 10,000 FOOT OVERVIEW METHODOLOGY



WHAT TYPE OF ACCESS?

- We want **white-box access**
  - Read or View permissions to **all** accounts in scope
  - Console + API keys

- We also want **assumed breach access** for dynamic testing
  - Compromised developer IAM user/role,
  - EC2 or Lambda roles
  - Leaked access keys
  - Whatever situation the client is most worried about



A high-quality discussion in low quality.

Opinion: Blackbox cloud pentesting is wasting money on gathering context

HOW ARE THE ACCOUNT(S) BEING USED?

- Get **diagrams** and **documentation**
- Talk to **developers/admins** in 1:1 settings
  - TIP: If you get conflicting diagrams/documentation, make a note for later
- Use the white-box access for **automated tools** that **visualize** resources and API usage in the account

- https://github.com/Fennerr/PMapper (I like this fork)
- https://github.com/nccgroup/PMapper

- https://awstip.com/visualizing-api-call-activity-in-your-aws-account-e5b37b520106
- https://awstip.com/how-to-list-all-resources-in-your-aws-account-c3f18061f71b
- https://medium.com/@michael.kirchner/exploring-aws-resource-explorer-825498b5307d

A pentester mapping clouds, 2023. JPG. Color.

WHAT LOW HANGING FRUIT EXISTS EVEN A SKIDDIE COULD FIND?

- Perform an automated configuration review across **every** account
- Tools like ScoutSuite, Prowler, CloudFox, Pacu, or a **CSPM** your client is using
- **Programmatically identify attack paths**

Tip: note the services known to be in use that are **NOT covered by automated checks**

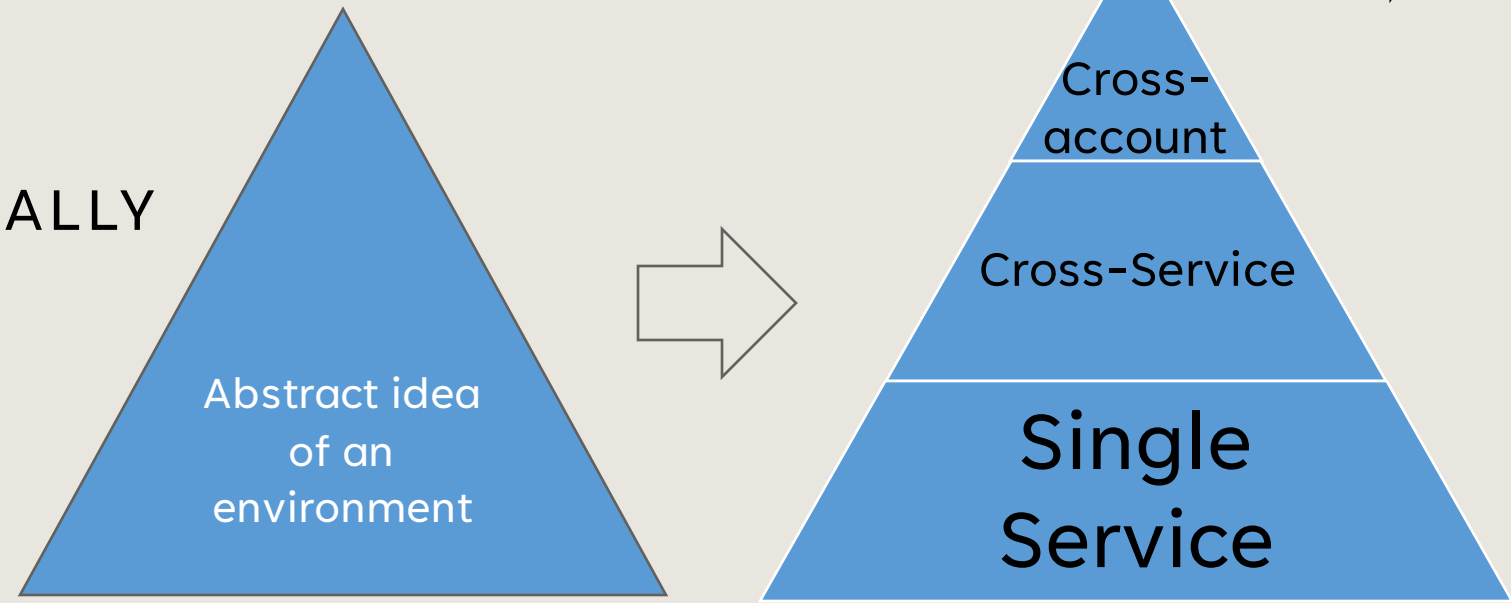
WE’VE DONE THE AUTOMATION!

Now we need to do the manual work

“A developer’s job is to work with abstractions to achieve their goal. Our job [pentesting] is to fact check their assumptions about those abstractions so we can all sleep at night.”

-Unknown

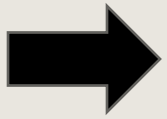
“LOGICAL LEVELS” TO MANUALLY GATHER CONTEXT FROM



WHAT ARE WE DOING WITH THIS RELEVANT CONTEXT?

We make attack paths!

Create manual attack paths using context gathered treating them as a hypothesis



Confirm or disprove our attack path hypotheses with dynamic testing

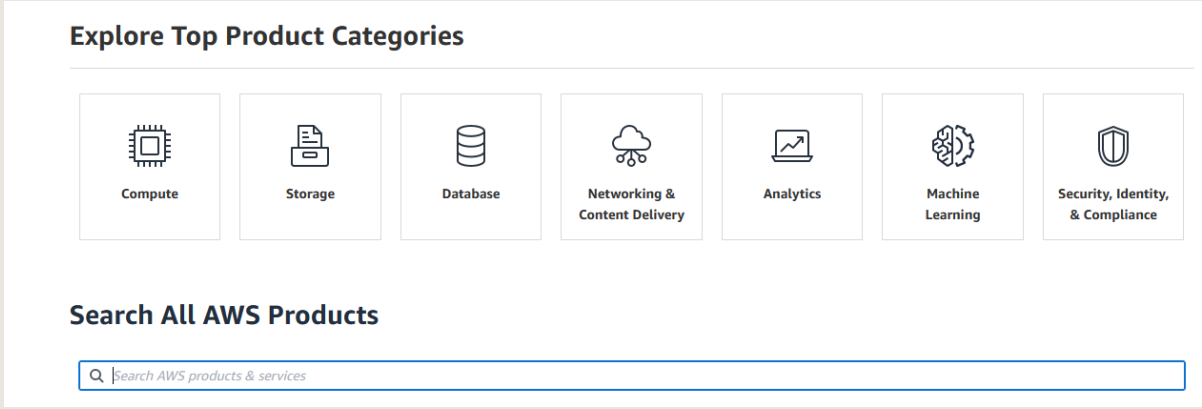
MANUAL CONTEXT GATHERING: SINGLE SERVICE

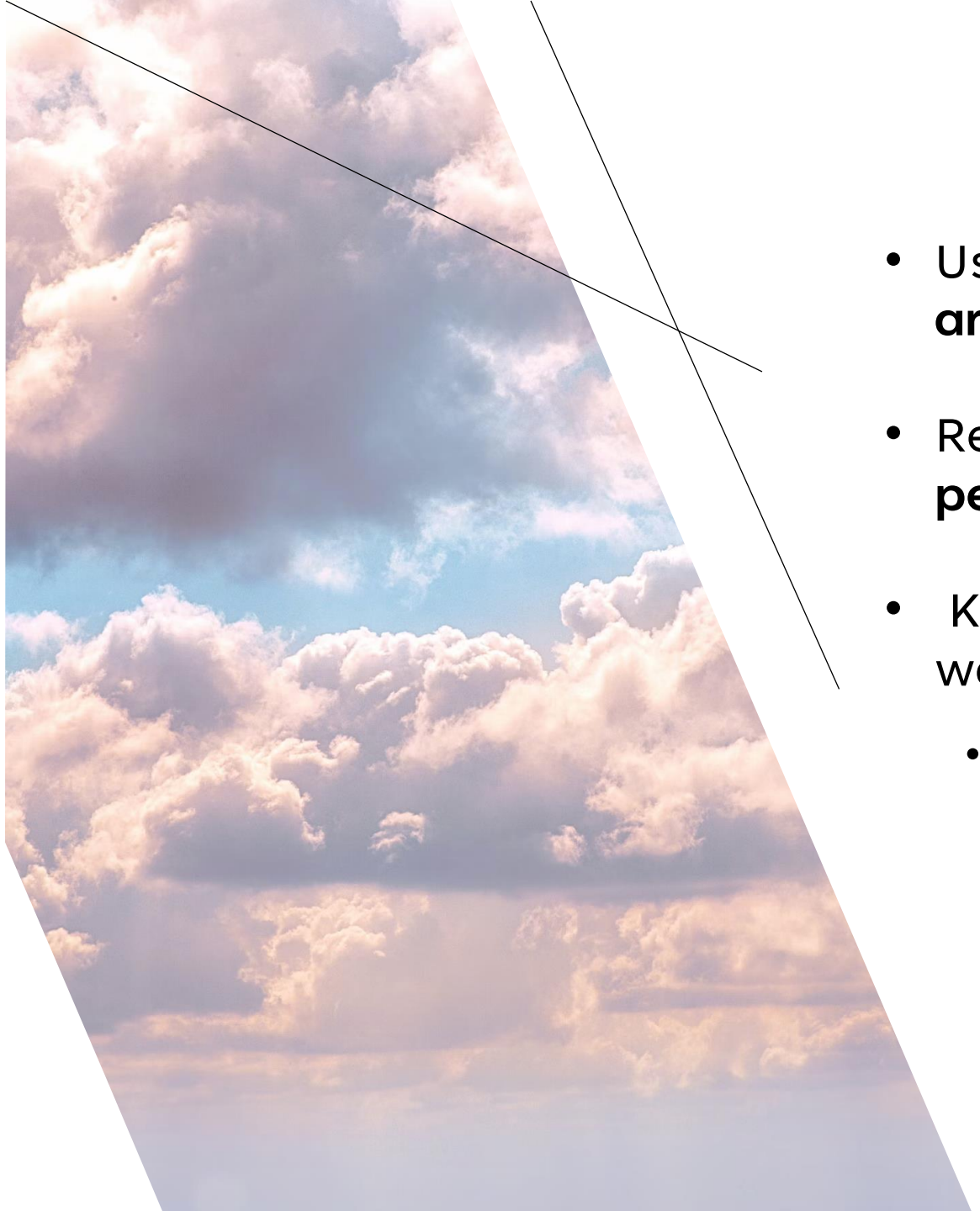
- Research & check for **publicly known misconfigs or issues**
- Read AWS documentation for **API calls**, **best practices**, and **features**
- Note **attack paths** to try

Example Note: A Lambda role has a lot of permissions and is used for all Lambdas. It is a good target for privilege escalation.

Tip: refer to existing research to similar services in the service's category to find possible misconfigs

https://aws.amazon.com/products/





MANUAL CONTEXT GATHERING: CROSS-SERVICE

- Use the relevant context to see **which services are used with each other**
- Review the configurations from the **perspective** of how they interact
- Keep **more notes** for what attack paths you want to try
  - **Example Hypothesis**: *This Lambda is dependent on code in an s3 object. My starting position has s3:PutObject on * resource. If I modify the code in that s3 object to retrieve the Lambda role credentials it will result in privilege escalation.*

MANUAL CONTEXT GATHERING: CROSS-ACCOUNT INSPECTION

- Look at **AWS Organizations** configs
- Look at **trust policies**
- Look at **SCP for guardrails** that ruin your lovely attack paths
- **Combine notes** on attack paths to try and from **which accounts**

Example Hypothesis: There is no SCP in the account affecting S3:PutObject in Account B. I need to pivot to Account B to use that attack path.

🎉 MANUAL CONTEXT GATHERING COMPLETE 🎉

You've now explored the configurations at many logical levels and obtained more relevant context

You've also kept notes on what attack paths you want to try

WHAT DO WE DO NOW?

Dynamic Testing

- Confirm/disprove assumptions and theories dynamically
- Write your findings
- Bonus points for recommending architecture or guardrail changes instead of “wack-a-mole” fixes

HIGH-LEVEL OVERVIEW OF MY METHODOLOGY

- Use meetings to gather project context quickly
- Get white-box access & assumed breach access

- Run automated scans
- Get attack paths from automated tooling

- Get context from every logical level
- Add custom attack paths

- Dynamically test attack paths
- Write findings for the issues

THE FUTURE OF AWS PENTEST METHODOLOGY

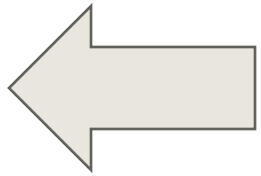
- We need a **standard** for an AWS pentest methodology that is **community accepted**
  - Scan-and-rebrand vendors exist
  - No way to know the quality of cloud assessment
  - To help train the next generation

- Well… someone has to get this started

PROJECT: STANDARD

Create

- Create the documentation we wished we had when we started
- Open-source it!



We are here

Improve

- Find an independent organization to host
- Allow practitioners to contribute & correct
- Keep out blatant promotion

Adopt

- Train juniors around it
- Advocate for internal adoption
- Scan-and-rebrand wall of shame?

THANK YOU

I hope this talk was helpful or, at the very least, interesting.

Come help me make it!

DM me on the Cloud Security Forum Slack

We'd love to compare methodology and build something great for those who come after us

Lizzie Moratti
@MorattiSec (Mastodon, Medium, X)