PLANNING: Solution Architecture

Goal: To provide contextual awareness for **suspicious events** in incident alerts to help Security Operations triage incidents.

Background: Most obvious ("universal bad") attacks are secured against using existing AWS solutions such as setting strong IAM guard rails and GuardDuty. While the insights GuardDuty provides forms part of the contextual awareness, our focus is on events that *can* occur (not designed against) but is ambiguous whether they should (IAM modification, PII in S3 etc).

Output

Types:

- **Event**
    - Something that's happened which is ambiguously/definitely bad
- **Incident**
    - A group of events that have either been automatically or manually correlated together
- **Alerts**
    - A message that is constantly updated with new incident details

Machine readable alerts with actionable and contextual information

For demo purposes connecting that information to Slack for alerting but could be generalised into other systems such as a SIEM, PagerDuty etc.

- Event ID(s) (easy to reference ID(s))
- Item/Activity
- Resource being read/created/updated/deleted
- Timestamp
- IP Address/MAC address/other network information
- Location (via GeoIP or otherwise)
- Environment (e.g. AWS account name for larger organisations or other contextual clues e.g. STAGE=dev tags)
- Suspicious IAM User/Group
- Owner IAM/Group
- User Agent/Browser
- Resource-specific information
    - IAM: highlight dangerous role information (s3:* etc.)
    - S3: what file was read/edited etc. whether it contained PII etc.
    - EC2: unusual flow patterns (unusually high volume, strange location - all context specific, even better if determined using historical flow data from

        CloudWatch to see if it is a regular pattern e.g. peak hour traffic v.s. exfiltration)
    - CloudWatch: related log events
    - Tags
- Alert severity (as dynamically determined)
- Quick-link to relevant resources e.g. open in AWS Console

Stretch goals
- Link to apply “pause” remediation e.g. to temporarily apply an IAM lock while investigating
- Link to full incident information
- Assign to specific member of SecOps
- Manually assigned alert severity (only if changed)
- More detailed explanation of why alert generated e.g. alert generated because PII put in S3 bucket which is bad because ....
- Ruleset ID(s) (the rulesets which are incorporated as part of the incident)

MVP **IDEAS**

Phase 1

1. Intake a select amount of data
2. Set up initial Lambdas for rule processing, adding context, and severity assignment
3. Transforming that data into a standard machine readable format e.g. JSON w/ schema
4. For demo purposes connecting that information to Slack for alerting but could be generalised into other systems such as a SIEM, PagerDuty etc.

Use cases
1. Someone **PUT**s PII into an S3 bucket that has unrestricted permissions (e.g. a group which shouldn’t be able to read PII ordinarily has permissions to)
    - Or buckets are marked specifically for “PII”
    a. Mistake
    b. Exfiltration

2. Someone **READ**s from an S3 bucket that they shouldn’t typically read from e.g. a Canary bucket or PII-marked bucket but they technically can read from e.g. they have ownership as a C-level etc.
    a. Insider
    b. Phishing attack
    c. Leaked credentials

Phase 2
1. Improving automatic severity escalation/de-escalation

2. Adding more sources/use cases
3. Improving “contextual” information display (e.g. hiding less relevant pieces of information in the initial alert e.g. don’t need to show location if not suspicious origination etc.)
4. Creating a basic full display of all notes/related events chronologically e.g. webpage
5. Creating the concept of “incidents” where different events are grouped together
6. Change windows
7. Ops teams can create their own basic rulesets
    a. ...

Phase 3 (Stretch)

1. Storing incident metadata/state in a database so it can be updated on-the-fly
2. SecOps can remove extraneous events from an incident, SecOps can add events they think is related
3. Overview of all active incidents, ability to “resolve” them and for them to re-open
4. SecOps can manually add notes or escalate/de-escalate events according to their own reasoning
5. SecOps can assign individuals to incidents so they know who is working on it
    a. Alternatively teams can simply use Slack or a more advanced SIEM to manage that e.g. Slack :eyes: for looking into etc.
6. Improved timeline view

Phase ∞

1. Teams can create more advanced rulesets with logic