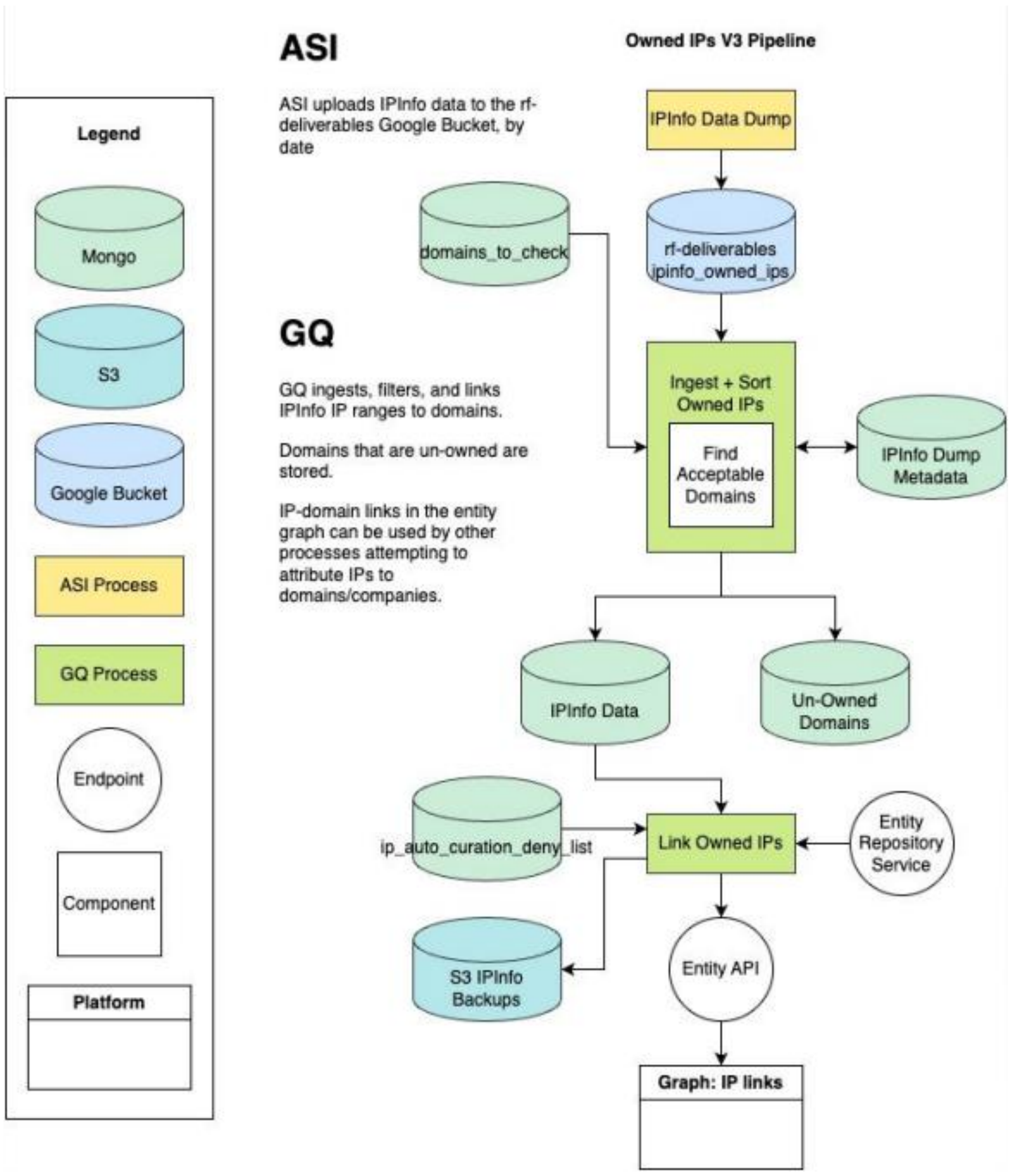

Legend
Mongo
S3
Google Bucket
ASI Process
GQ Process
Endpoint
Component
Platform
ASI
ASI uploads IPInfo data to the rf-deliverables Google Bucket, by date
GQ
GQ ingests, filters, and links IPInfo IP ranges to domains.
Domains that are un-owned are stored.
IP-domain links in the entity graph can be used by other processes attempting to attribute IPs to domains/companies.
Owned IPs V3 Pipeline
IPInfo Data Dump
domains_to_check
rf-deliverables ipinfo_owned_ips
Ingest + Sort Owned IPs
Find Acceptable Domains
IPInfo Dump Metadata
IPInfo Data
Un-Owned Domains
ip_auto_curation_deny_list
Link Owned IPs
Entity Repository Service
S3 IPInfo Backups
Entity API
Graph: IP links

Identity: Core Architecture Overview Data Collection - Automated scripts continuously scrape Telegram channels, bots and third-party data providers to gather malware logs, which are then repacked and uploaded to an encrypted S3 bucket. These scripts run on a dedicated VPS solely used for downloading, repacking, and uploading logs. - Manually collected data is acquired and downloaded on VPS. We perform repacking of logs and then upload the zipped archives to an encrypted S3 bucket. Archive Unpacking and Credential

Indexing - One encrypted S3 bucket hosts raw malware log archives. - Unpacker process unpacks archives and uploads files with structured credentials (json objects) into an output path in S3. - The indexer process indexes credential objects from S3 into the self-hosted credentials ES cluster. The ES cluster contains a series of indexes that are filled up and get closed as new data arrives. Detection Generation - Using client configurations (stored in self-hosted mongo), detections are generated in an event-based fashion whenever: a) new data comes in b) new client rules are added. - Configuration management involves a system of interconnected python processes and apis that are used by the front end. External-facing APIs - Credentials API - Allows clients to search credential data for a set of subjects (/lookup) or a set of domains (/search) - Detections API - Allows clients to search pre-computed detection records stored in the detections index

Data Store Systems Summary of Approach MongoDB Data retention and lifetime policies dictated by product operations in "AWS Main Zone" account. Contains stateful/transactional data, client configurations and alert data. Data is split between two clusters: the datascience cluster and cred-leaks cluster Client configurations exist in Mongodb. These need vault-managed credentials to be accessed. The primary way to edit this database is via a configuration API. AWS S3 Encryption occurs at rest within our S3 buckets by default and requires AWS authentication via Okta SSO. The bucket used for the source data is encrypted and read/write access is managed using IAM keys, which are managed by Ops Encryption occurs within S3 but not when data is in transit or at rest at other points. We rely on HashiCorp Vault (hosted and maintained by product operations) for storing our programmatically used keys and credentials securely for any testing or deployed production applications. Compressed archives are stored in S3 which is encrypted. Replica backups supported by our Ops team. Elasticsearch This is self-hosted in the main VPC. It features about 40 inter-connected EC2 instances. It is password-protected. The Ops team dictates read/write access. Data retention and lifetime policies implemented by product operations in the form of index lifecycle management policies. Replica backups supported by our Ops team. Extracted credentials are stored in ElasticSearch. There is no encryption, but access is managed through shared access policies.

Identity: Data Duplication / Classification / Sensitivity Sources Summary of Approach Possible Duplicated Efforts Raw archives collected and sent to s3 are not deduplicated. Deduplication occurs in the archive unpacker process at the malware log level. Detection records in Elastic have natural deduplication (deduping by _id in elastic search). Playbook alerts also also deduplication (i.e. there will always only be one alert sent per detection, even if generated multiple times.) Data Classification Raw credentials from archives are stored in the password-protected elastic cluster, which lives in self-hosted EC2 instances in our AWS VPC. Client configurations are stored in pw-protected mongo clusters, which are hosted in our VPC in the DS Main Zone. Detections are considered more sensitive than credentials (since they map to client configurations), but they have the same permission requirements as credentials in the credentials cluster

Brand Features ● Domain Abuse ○ monitoring typosquats, keywords in domain names, and providing assessments per-domain ● Data Leakage ○ Code Repository Data Leakage – mentions/keywords/credential leaks on Github and Postman ○ Leaked Documents – monitoring potentially sensitive Scribd documents ● Brand Impersonation ○ Company/Executive Impersonations from Meta, Linkedin, X ○ Mobile App Impersonations on Apple/Google App Store ● Brand Mentions ○ Dark Web and Messaging Platform mentions, ○ Logo Detections and OCR mentions ● Takedowns ○ Takedown requests via PhishFort for phishing, malware, brand abuse, social media, mobile apps, messaging

Brand - Data Harvested and Sources ● Leaked Documents ○ Sourced from Scribd via search engine dorking ● Leaked Credentials ○ Postman ○ Github ○ Data from Dark Web Team ● Website DOM + Screenshots ○ HTML Grabber internal scraper ○ Urlscan ● Certificate Registrations ○ SecurityTrails ● Domain Registrations ○ SecurityTrails ○ ZoneFilesIO ○ CyberToolBelt ● Subdomain Observations ○ SecurityTrails ● App Impersonations ○ Apple iTunes Search API ○ Google Play Store ● Company/Executive Impersonations ○ Vetric API ○ SocialLinks API ● Brand Mentions ○ References harvested from various sources ■ Dark Web/Messaging references produced from Dark Web Team ■ Logo Detection/OCR mention references produced from Image Analytics Team

Authentication

● Mongo user/pass authentication for databases

● S3 Authentication with Access and Secret Keys stored in Vault

● Internal APIs

○ OAuth authentication

○ What services can access what services with what privileges is defined in manifests. Such manifest changes are applied in

production only with a special RFAdmin permission

● External API auth tokens stored in Vault

● ACLs: Entity framework and attribute system have per-entity ACLs. Data access is limited mostly to owners

Encryption

- Alert framework has alert details encrypted, although accessible via debug API given an RFAdmin permission
- S3 buckets have server side encryption enabled

Hardening

- MongoDB replicas & snapshots
- S3 buckets configured with backups
- Services that do not need internet are in a private subnet

Tech Stack

- Languages: Python, Scala, TypeScript
- Infrastructure:
  - Main Zone (Managed/standardized by Operations)
    - EC2, MongoDB, AWS S3, RabbitMQ
  - Data Science AWS Account
    - ECS (Fargate), ECR, Lambda, DocumentDB, AWS S3, Terraform
  - Analytics Framework AWS Account
    - EMR on EKS (Apache Spark)
    - Apache Airflow
- Proxy providers: Oxylabs Datacenter IPs, Proxymesh
- SendGrid via Alert Framework for emails

Monitoring

- PagerDuty
- Grafana

Brand - Data Retention

- GitHub Code Leakage
  - 5-10 line snippets of code per leak

        - Indefinite in platform
    - Commit metadata
        - Indefinite in Mongo
- DOM and Screenshot Scraped Data
    - Full DOM
        - Indefinite in S3
        - Eventually will be pushed to glacial storage
    - Base64 Encoded Screenshots
        - Indefinite in S3
        - Eventually will be pushed to glacial storage
- URLScan
    - Full DOM
        - Indefinite in S3
        - Indefinite in SiteRecorder service
    - Full Image
        - Indefinite in S3
        - Indefinite in SiteRecorder service

T&S Attacker Structured Data

What are we talking about?

■ ~240 python processes

■ ~30+ source ingestions

■ Signals (Risk rules, risk scoring)

■ Delivers data to core platform through ADFs.

■ TnS does not scrape raw data from sites, most harvesting comes from API based pathways and approved

vendors.

■ We do not classify our data with any particular schema, though it's something we would like to implement

■ We do not directly store client provided data, most of our source come from other private or public companies /

sources

■ Common data includes TnS produces References, Events, Entities (mostly IOCs like C2 IPs), ThreatLists, and Entity Annotations.

T&S Signals: Assessing severity of each piece of evidence

We now have a list of triggered risk rules and their severities. We can condense this into a number we call the risk score.

The highest severity risk rule sets the base risk score.

Every additional unique source on a rule gives it a higher score in the same category, but cannot move it up beyond the maximum score for that risk band.

Backups

■ S3 buckets are backed up (Cross Region)

■ Not by default, but some have been backed up

■ Multiple instances of API endpoints behind ALB

■ We take EBS snapshots every day

■ Mongo replicas

Deployment

■ Backstage (Bob!),

■ GitLab Runners,

■ ArgoCD,

■ Jenkins (CI) with Chef (CD)

Monitoring

■ AWS Cloudwatch (SaaS)

■ Grafana (Self hosted)

■ Alerts: Slack

■ PagerDuty

What is the Defender Space Graph?

"Connected" entities:

● 5M⌵ companies/organizations

with at least one domain

● 1B⌵ domains and subdomains

● 20M⌵ IP addresses directly linked

to a domain, including CIDR blocks

○ • 1.8B individual IP

addresses attributable to

companies

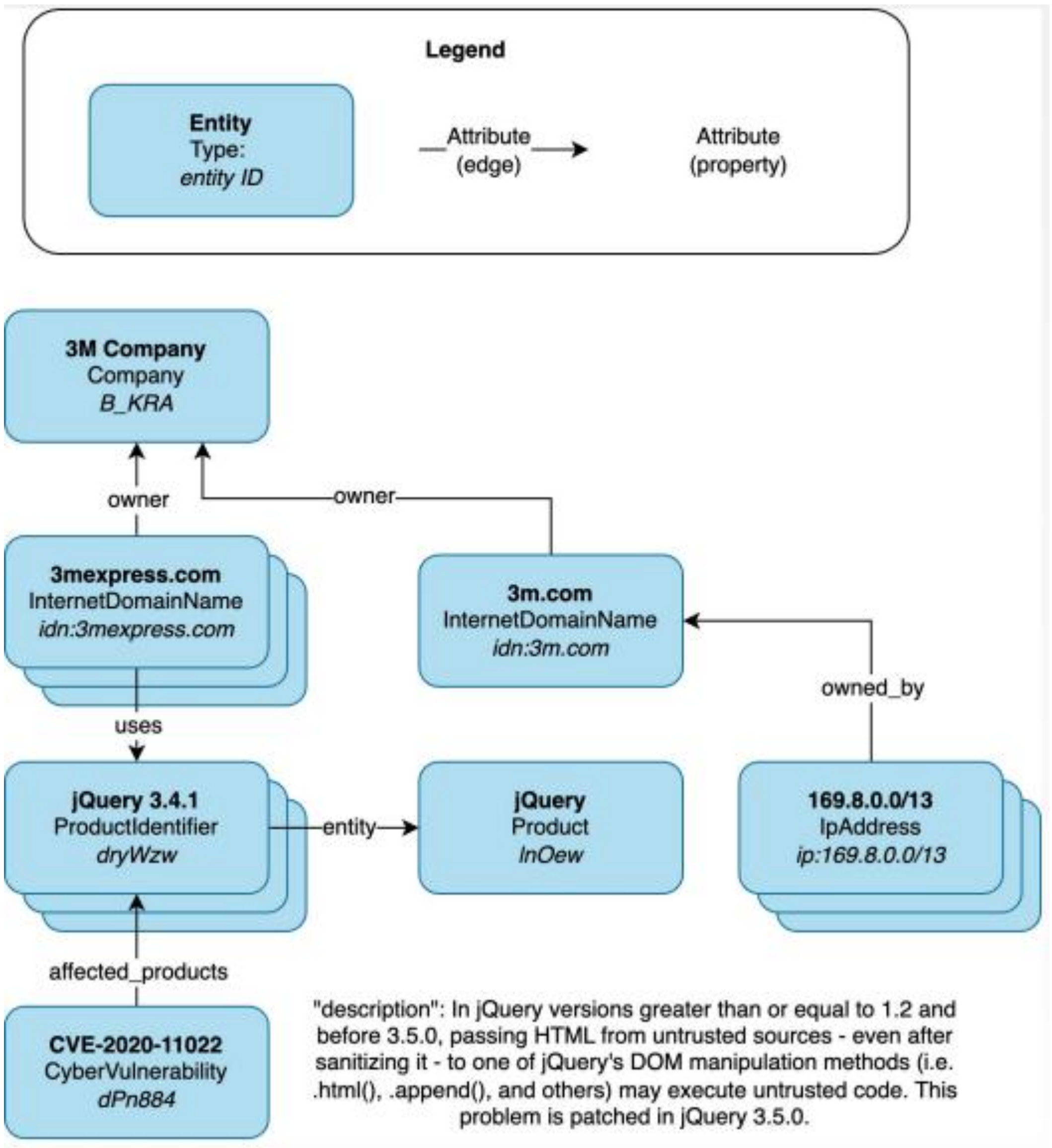
● 1.2M⌵ products (versioned and

unversioned) linked to

vulnerabilities

● 250K⌵ vulnerabilities with at least

one affected product



"description": In jQuery versions greater than or equal to 1.2 and before 3.5.0, passing HTML from untrusted sources - even after sanitizing it - to one of jQuery's DOM manipulation methods (i.e. .html(), .append(), and others) may execute untrusted code. This problem is patched in jQuery 3.5.0.

Discussion on Subnets and Prefixes:

DMZ is used for services directly accessible from the Internet, primarily for ingress.

Process subnet is for applications harvesting and downloading data from the Internet.

Isolated process subnet is for applications that cannot access the Internet.

Tools subnet is for internal tools used by non-engineering parts of the company.

Infrastructure subnet is mainly for databases and other infrastructure services.

Public subnet is for machines with public IP addresses.

Prefix and Tagging:

Discussion on the prefixes used for EC2 instances and auto-scaling groups, such as BE, DT, IS, and others.

Clarification on the use of tags like 'type' and 'role' for better identification and management of instances.

Identification of instances without proper tags and the need for cleanup.

Action Items:

Review and cleanup of stopped instances and subnets that are no longer in use.

Verification of instances with public IP addresses to ensure they are correctly configured.

Technical Details:

Mention of specific tools and services running in different subnets, such as Chef, Elasticsearch, and RabbitMQ.

Discussion on the use of EKS for long-term storage of Prometheus data.

Next Steps:

Further investigation into specific instances and subnets to ensure proper configuration and usage.

Continued collaboration to refine tagging and prefix usage for better management and identification of resources.

Kubernetes Cluster Usage:

The Kubernetes cluster is currently used for various processes, including text analytics and serving models.

There are multiple clusters for different purposes, such as Dev, Production, Hatching, and GitLab runners.

The clusters are isolated by VPCs, and there are plans to move Dev and Production clusters to separate accounts for better isolation.

ElasticSearch Cluster Details:

Coordinating nodes act as load balancers and routers, handling queries and distributing them across data nodes.

Interactive coordinating nodes are dedicated to the portal to prevent crashes from large queries.

Various clusters serve different purposes, such as alert notifications, APM ingestion, collective insights, credentials, data analytics, entity index, and more.

MongoDB Cluster Details:

The role tag in MongoDB instances indicates the cluster name, which helps identify the specific function of each cluster.

There are multiple clusters, including entity, reference, signal, text collection, and more.

RabbitMQ Exchanges and Queues:

A list of RabbitMQ exchanges and queues will be provided to help map the data flow within the system.

S3 Buckets:

The largest S3 buckets are used for data ingestion into the platform, with significant growth observed in the HPC-related buckets.

Data science and analyst teams are responsible for managing the data coming into the platform.

General Observations:

The meeting discussed the importance of proper tagging and naming conventions to ensure clarity and consistency across the infrastructure.

There is a need for cleanup and consolidation of resources to optimize costs and improve efficiency.

GQ Tech Stack

~150+ services: All GQ-owned services are deployed on AWS EC2 instances in the Main Zone.

Tech Stack

Code	Data	Authentication
Python	Elastic	OAuth
Java Script	MongoDB	MongoDB
React	AWS Neptune	AWS Access Keys
Open Cypher	AWS S3	
Gremlin	InfluxDB	
	RabbitMQ	
	ASI	
	GCP Google Bucket	

Hardening

MongoDB replicas & snapshots

S3 buckets configured with backups

Google bucket configured with backups

AWS Neptune snapshots

Curation - Tech Stack

- SaaS products

○ Business communications Gmail, Gchat, Slack

○ Tickets, attachments, work breakdowns, reporting Jira, Google Sheets, Domo

○ Configuration source control GitLab

○ Procedures, FAQs, training materials Confluence, Google Docs

○ Time cards Clockify (provided by BPO partner)

● Curation self-managed

○ EC2, Python Ticket automation scripts, ETL scripts for reporting

○ Raven (EC2, Python, Mongo, JS Deprecated internal tool web app

● DS services - main dependencies

○ Torvest-Manila web data collection (scrapers)

○ Februus, Zana internal tool web applications

● RnD services - main dependencies

○ RSS reader, RSSifier RFMachine web data collection (feed readers)

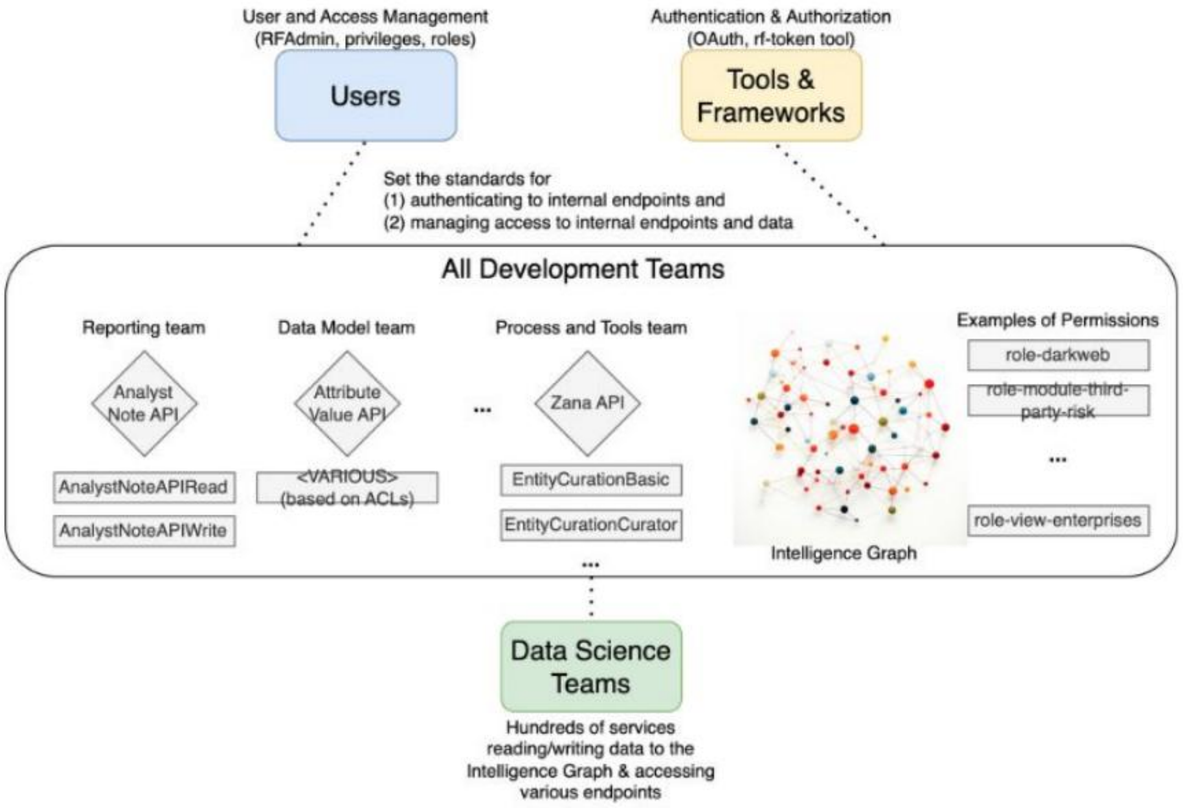
○ DirectStore one time data loads

○ Source API data updates to our sources, InfoNGEN feed management

○ Entity API, Annotation API data updates to our ontologies

○ Event, Reference, Signal API reanalysis of platform data after ontology changes

GQ - Authentication + Authorization for Internal APIs



How Data Science interacts with various endpoints

Traffic Management and Service Discovery:

- Lutz explained the use of tags in the console for traffic management, emphasizing the importance of preventing accidental exposure of services to the internet.
- The team uses a path prefix to easily find microservices, making it simple to locate services via specific URLs.
- The console service discovery system is used programmatically with tags to define services, and a dashboard is utilized to monitor response times and identify issues.

API and Microservices:

- The team discussed the analyze API, which uses a path prefix for easy access and identification of microservices.
- There was a focus on ensuring that services exposed externally are properly tagged to avoid accidental exposure.

Monitoring and Troubleshooting:

- Lutz highlighted the use of dashboards to monitor service response times and identify slow microservices, which helps in troubleshooting performance issues.
- The team uses a combination of tools like Graphite, InfluxDB, and Telegraph for system metrics and application monitoring.

Scaling and Configuration:

- The team discussed the process of scaling Elasticsearch clusters, including the use of internal tools for managing autoscaling groups and the importance of balancing shards across availability zones.
- MongoDB clusters are managed with internal tools, and scaling typically involves changing instance types rather than adding nodes due to MongoDB's primary-secondary architecture.

Security and Upgrades:

- The team follows a strategy to stay out of end-of-life versions for security reasons and prefers to be on the second newest version for both MongoDB and Elasticsearch.
- Upgrades for MongoDB and Elasticsearch are coordinated to minimize downtime and ensure smooth transitions.

Authentication and Access:

- MongoDB uses username and password authentication, while Elasticsearch employs API keys and is moving towards TLS for better security.

Tools and Automation:

- The team uses Terraform for managing infrastructure components like VPCs, security groups, and load balancers, but not for starting instances.
- Internal tools are used for specific tasks like scaling autoscaling groups and managing instance lifetimes.

Action Items:

- **Traffic Management** - Lutz to ensure proper tagging of services to prevent accidental exposure.

- **Scaling** - Team to monitor Elasticsearch cluster and add nodes as needed based on shard usage alerts.
- **Upgrades** - Team to coordinate upgrades for MongoDB and Elasticsearch to stay current with security patches.

Discussion on Data Centers and Infrastructure:

- Frank explained the process of decommissioning old servers, including Horace and FU2M1, and the effort involved in physically removing them from the data center.
- The team discussed the transition to using Wire Guard concentrators for egress traffic, replacing the old servers.
- There was a conversation about potentially moving the remaining infrastructure from Digital Realty to Hetzner, considering cost and maintenance factors.
- Frank mentioned the possibility of moving Mac Mini's to a different provider due to the high cost and maintenance burden.

Monitoring and Alerts:

- Frank described the use of Prometheus for monitoring server health and performance, including node metrics and external health monitoring on public endpoints.
- Alerts are set up to notify specific team members if critical services go down.

Plans for Singapore Data Center:

- The team discussed the upcoming deployment of sandbox environments in Singapore, planned for Q3, to address latency issues for APJ clients.
- The deployment will start with three sandboxes and three edges, similar to the setup in the US.

Data Growth and Storage:

- Frank provided insights into data growth, mentioning that the US data growth is relatively insignificant compared to other regions.
- Storage servers are individual and not pooled, with data offloaded to S3 for reports and sample files.

Security and Compliance:

- The team discussed the challenges of supporting Mac OS and iOS due to strict security requirements and licensing limitations.
- Android malware analysis is virtualized, with ongoing efforts to support ARM for performance improvements.

Miscellaneous:

- Frank confirmed that the Git server is hosted internally but plans to move it due to its current location in the data center.
- The team briefly touched on the use of Datadog for monitoring and alerting in the HPC environment.