Discover, Analyze and Secure your APIs Anywhere

Pranav Dharwadkar
VP Products

@pranavdh01

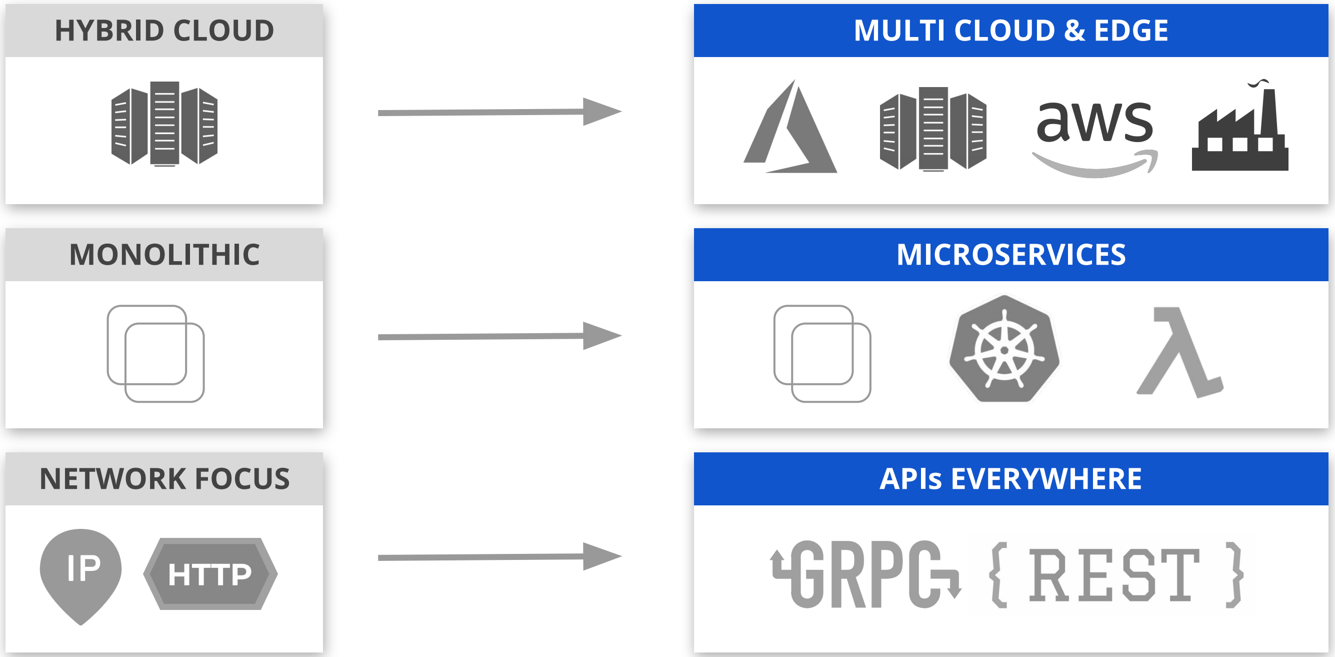
Jakub Pavlik
Director Engineering

@JakubPav

CNCF Webinar
Dec 2020

Volterra
Enable the Cloud for Everyone, Everywhere

THE NEEDS FOR MODERN APPS ARE CHANGING



Volterra

Causing APP/API Security challenges not solved by current tools

Traditional WAF only protects against these	**....**	**But not against these**
↓		✗ ↓
Top-10 Web Security Attacks		*Top-10 API Security Attacks*
Injection		Excessive Data Exposure
Cross-Site Scripting		Broken Object Level Authorization
XML External Entities (XXE)		Lack of Resources, Ratelimiting
....		....

App/API Security focussed approach required

*As seen in **real-world** examples next..*

Oct'20: Navigation App API exposed user-id, name

Attack Pattern

API Request

{ lat, long } →

{internal user-id } →

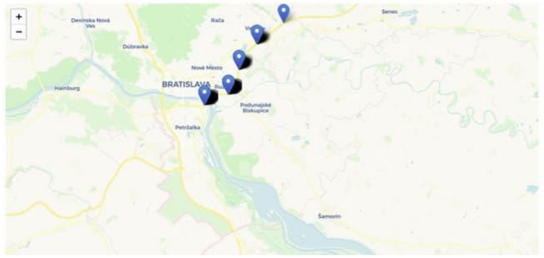
API Response

← { Internal user-id }

← { user-name }

Impact

User's personal identity and address revealed



Root Cause

Internal User-ID should never have been sent in the API-response

Oct'20: Dating App API allowed takeover of User's account

Attack Pattern

API Request

Password reset api { email address }

$\xrightarrow{\hspace{6cm}}$

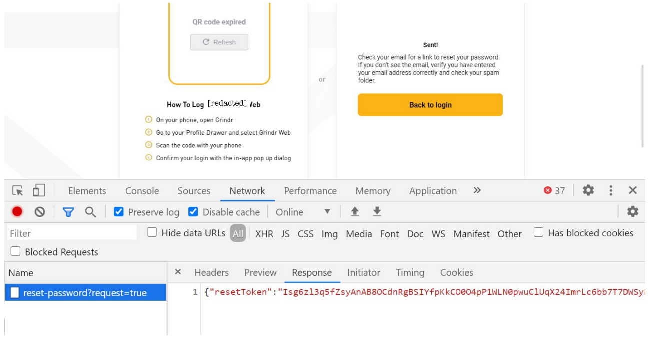
API Response

{ temporary password }

$\xleftarrow{\hspace{6cm}}$

Impact

Easily take-over users' account via password reset API, without even access to users' email



Check your email for a link to reset your password. If you don't see the email, verify you have entered your email address correctly and check your spam folder.

[✓] Preserve log [✓] Disable cache

[ ] Hide data URLs [ ] Has blocked cookies [ ] Blocked Requests

reset-password?request=true

```
{"resetToken":"Isg6zl3q5fZsyAnAB8OCdnRgBSIYfpKkCO0O4pP1WLN0pwuClUqX24ImrLc6bb7T7DWSy
```

Root Cause

Temporary password should never have been sent in the API-response

https://www.troyhunt.com/hacking [redacted] -accounts-with-copy-and-paste/

June'20: Coffee-shop App API exposed 100M customer records

Attack Pattern

$$\begin{tikzcd}\text{Coffee-Shop App} \arrow[r] & \text{Backend-For-Frontend} \arrow[r] & \text{Graph Service}\end{tikzcd}$$

```
GET /bff/proxy/stream/v1/users/me/streamItems/web\..\.\..\.\..\.\..\.\..\.\..\.\search\v1\Accounts\

GET /bff/proxy/stream/v1/users/me/streamItems/web\..\.\..\.\..\.\..\.\..\.\..\.\search\v1\Addresses\
```

Impact

100M customer records accessed using Coffee-shop gift card API

```
"@odata.context": "https://redacted.redacted.com/
    "AccountId": redacted,
    "AddressType": redacted,
    "AddressLine1": redacted,
    "City": redacted,
    "PostalCode": redacted,
    "Country": redacted,
    "FirstName": redacted,
    "LastName": redacted,
    "PhoneNumber": redacted
```

Root Cause

1. **BFF should not have been allowed to talk to Graph Service**
2. **WAF easily bypassed by this pattern "\..\.\..\.\..\.\..\.\..\.\..\.\". API gateway did not detect unusual unusual pattern.**
3. **100M Customer records should never have been sent in API response**

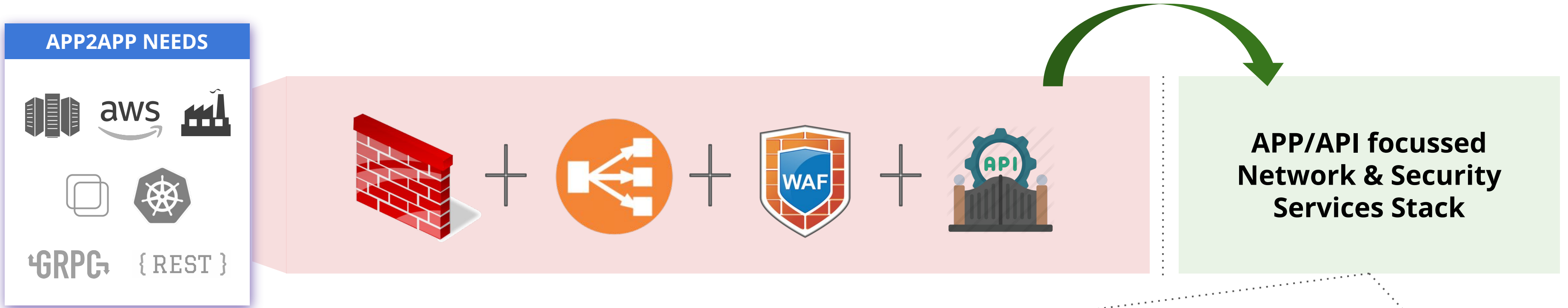
API Security Vulnerabilities Everywhere… Cloud & Edge

Feb’18	[redacted]g and [redacted] Smart TVs Vulnerable to Hacking, Consumer Reports Finds
Aug’20	Gym app management platform exposed info of thousands of users
Aug’20	Stealing a few million Wi-Fi PSKs
Aug’20	API flaws in [redacted]z allowed unauthorized start of engine.. remotely
Oct’20	Report: Online Fashion Retailer Exposes European Customers in Massive Data Leak

Key API Security Challenges seen in these examples

All	Traditional WAF did not protect against API attacks, as requests were legitimate
Coffee-shop	They didn't know which service was talking to which service?
Dating App	They didn't know which data is being shared by the APIs.
Navigation App	They didn't know if PII data being shared by APIs
Gym App	They didn't know the correct threshold values for ratelimiting the password reset API
Online Retailer	Manual application of policies led to security holes. Needed automated policies
Auto OEM **Smart TV OEM**	They only had API controls on cloud/DC. They didn't know how to secure APIs on device edge and branch

A NEW APPROACH IS NEEDED FOR APP/API Security…



Discover

Analyze

Secure

Deployable Anywhere

APP/API Security focused solution should solve the following...

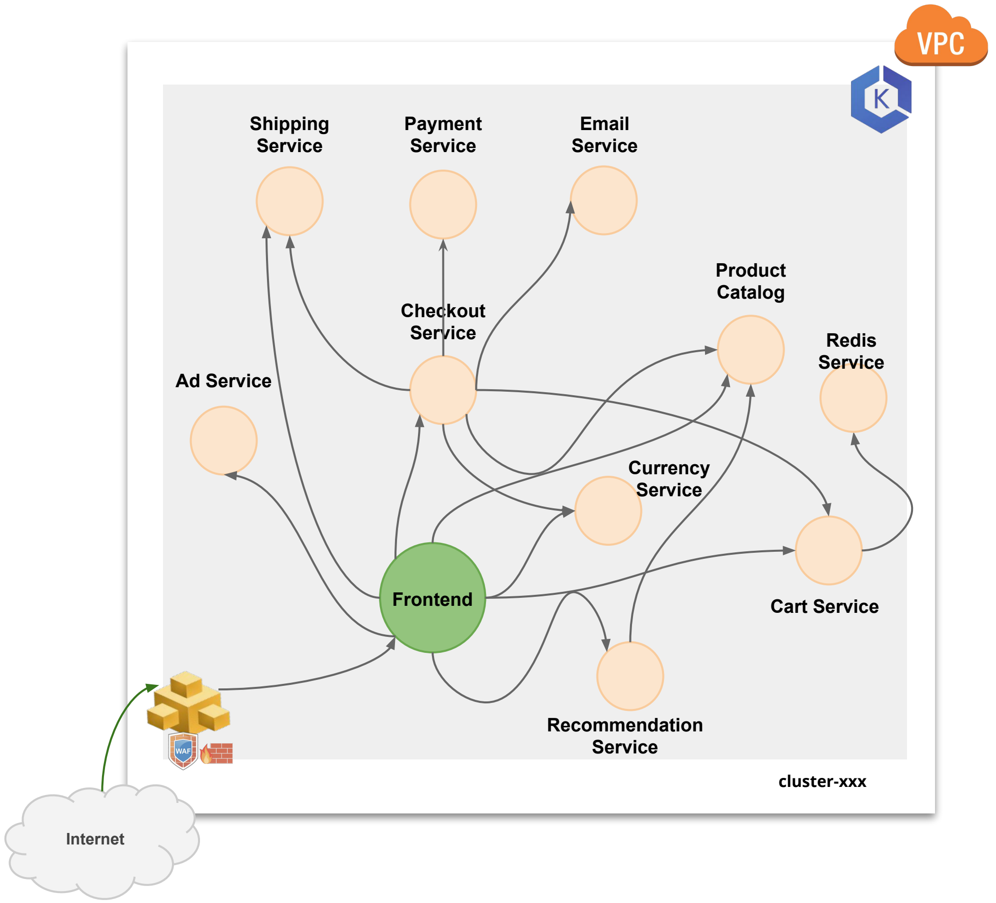
Discover	Who is talking to whom?
	How much? When?
Analyze	What is normal communication pattern?
	What data is being shared by the APIs? Is it PII data?
	What request rates are normal?
	What response sizes are normal?
Secure	What signature should I use to block PII data exposure ?
	What does a malicious API request pattern look like?
	How do I automatically apply policies?
Anywhere	Uniform API security policy across cloud/DC and edge (network, device, and branch)

Demonstration

Volterra

Enable the Cloud for Everyone, Everywhere

Hipster-shop (now Online Boutique) Application Topology



VIPs

● VIP

Hipster Webapp

Services

- Frontend
- Catalogue
- Shipping
- Payment
- Email
- Currency
- Cart
- Cache (redis)

Project: https://github.com/GoogleCloudPlatform/microservices-demo

Volterra

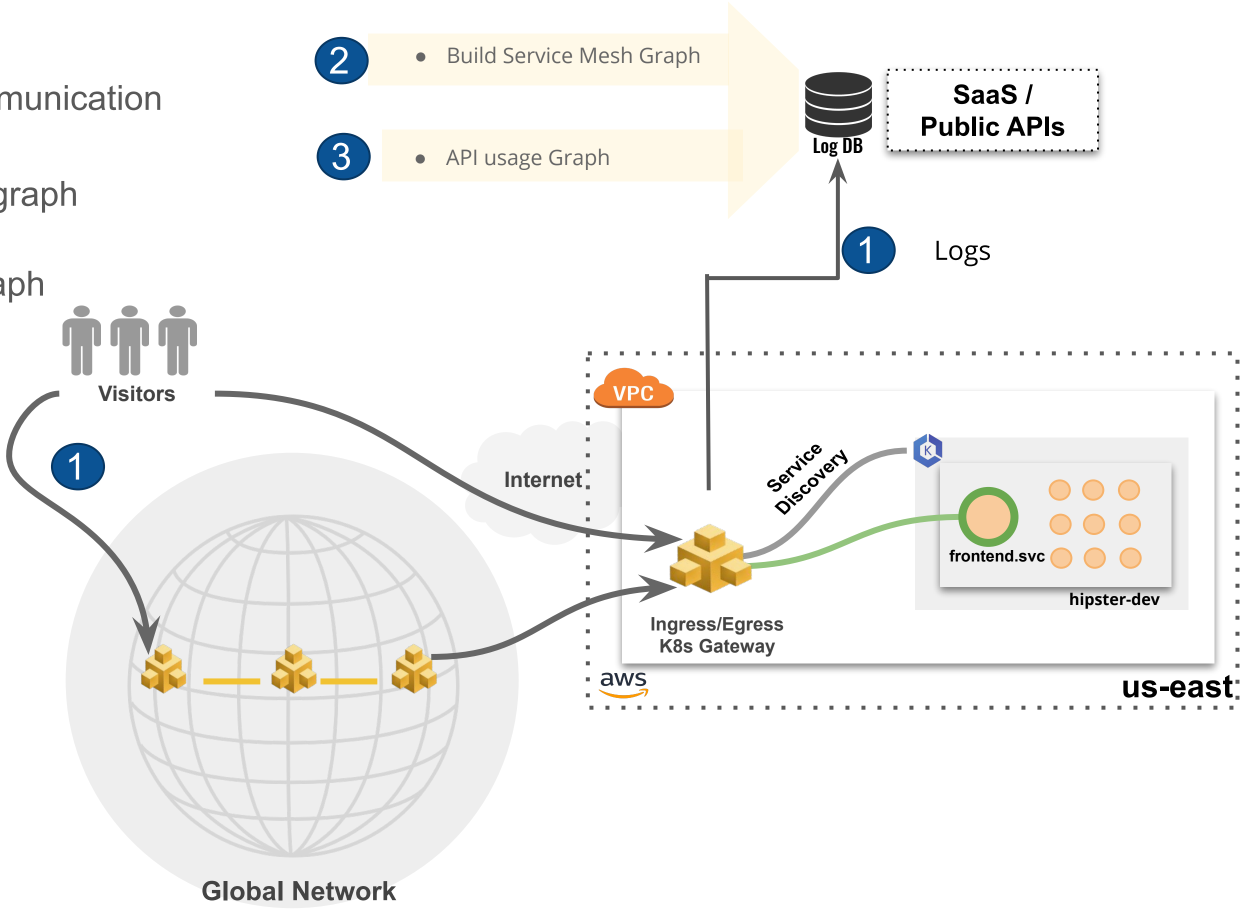
Discover app/api communication

Discover

1 Collect logs from app to app communication

2 Build app to app communication graph

3 Build API-API communication graph

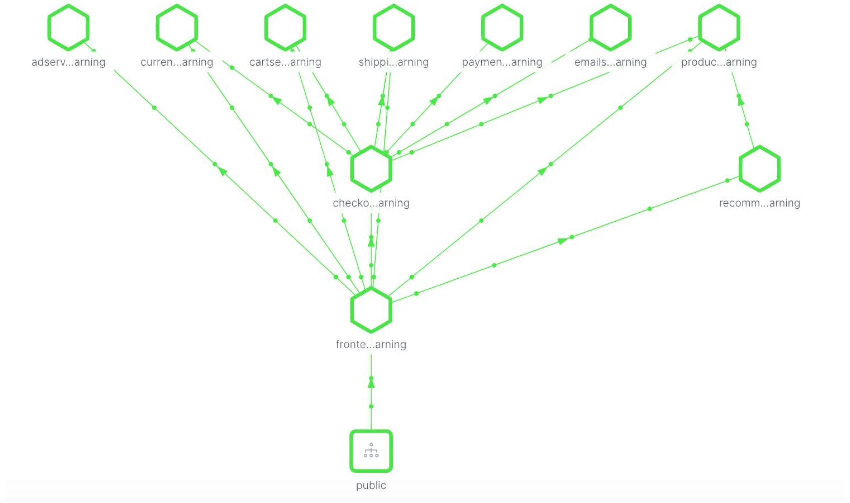


Volterra

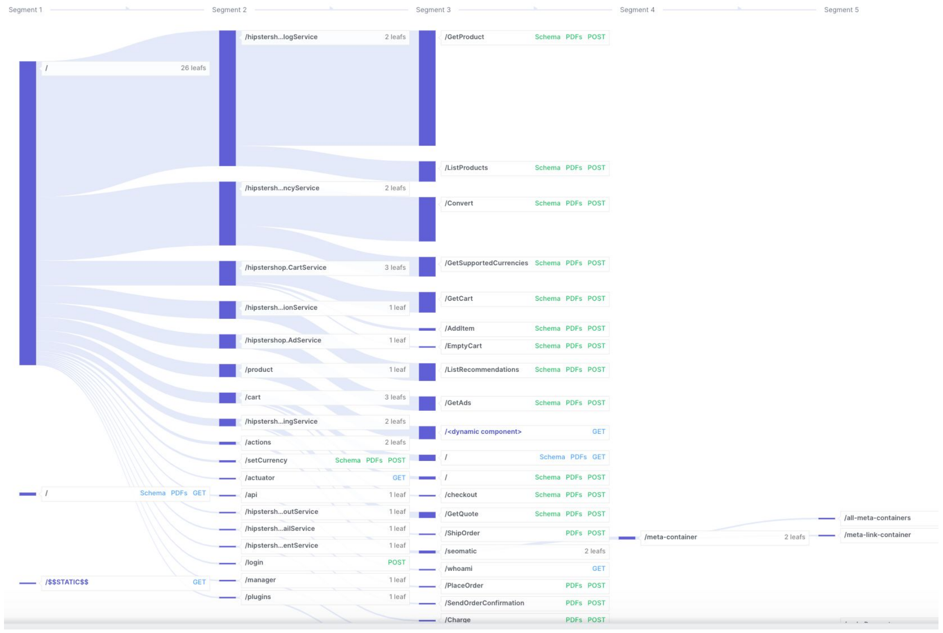
Discover app/api communication

Discover

App-to-App Communication Graph



API-to-API Communication Graph



Segment 1
Segment 2
Segment 3
Segment 4
Segment 5
/ 26 leafs
/ Schema PDFs GET
/$$STATIC$$ GET
/hipstersh...logService 2 leafs
/hipstersh...ncyService 2 leafs
/hipstershop.CartService 3 leafs
/hipstersh...ionService 1 leaf
/hipstershop.AdService 1 leaf
/product 1 leaf
/cart 3 leafs
/hipstersh...ingService 2 leafs
/actions 2 leafs
/setCurrency Schema PDFs POST
/actuator GET
/api 1 leaf
/hipstersh...outService 1 leaf
/hipstersh...ailService 1 leaf
/hipstersh...entService 1 leaf
/login POST
/manager 1 leaf
/plugins 1 leaf
/GetProduct Schema PDFs POST
/ListProducts Schema PDFs POST
/Convert Schema PDFs POST
/GetSupportedCurrencies Schema PDFs POST
/GetCart Schema PDFs POST
/AddItem Schema PDFs POST
/EmptyCart Schema PDFs POST
/ListRecommendations Schema PDFs POST
/GetAds Schema PDFs POST
/<dynamic component> GET
/ Schema PDFs GET
/ Schema PDFs POST
/checkout Schema PDFs POST
/GetQuote Schema PDFs POST
/ShipOrder PDFs POST
/seomatic 2 leafs
/whoami GET
/PlaceOrder PDFs POST
/SendOrderConfirmation PDFs POST
/Charge PDFs POST
/meta-container 2 leafs
/all-meta-containers
/meta-link-container

Volterra

Analyze APP/API patterns & data

Analyze

1 Baseline normal App-to-App and API-to-API patterns

2 Analyze APIs using ML

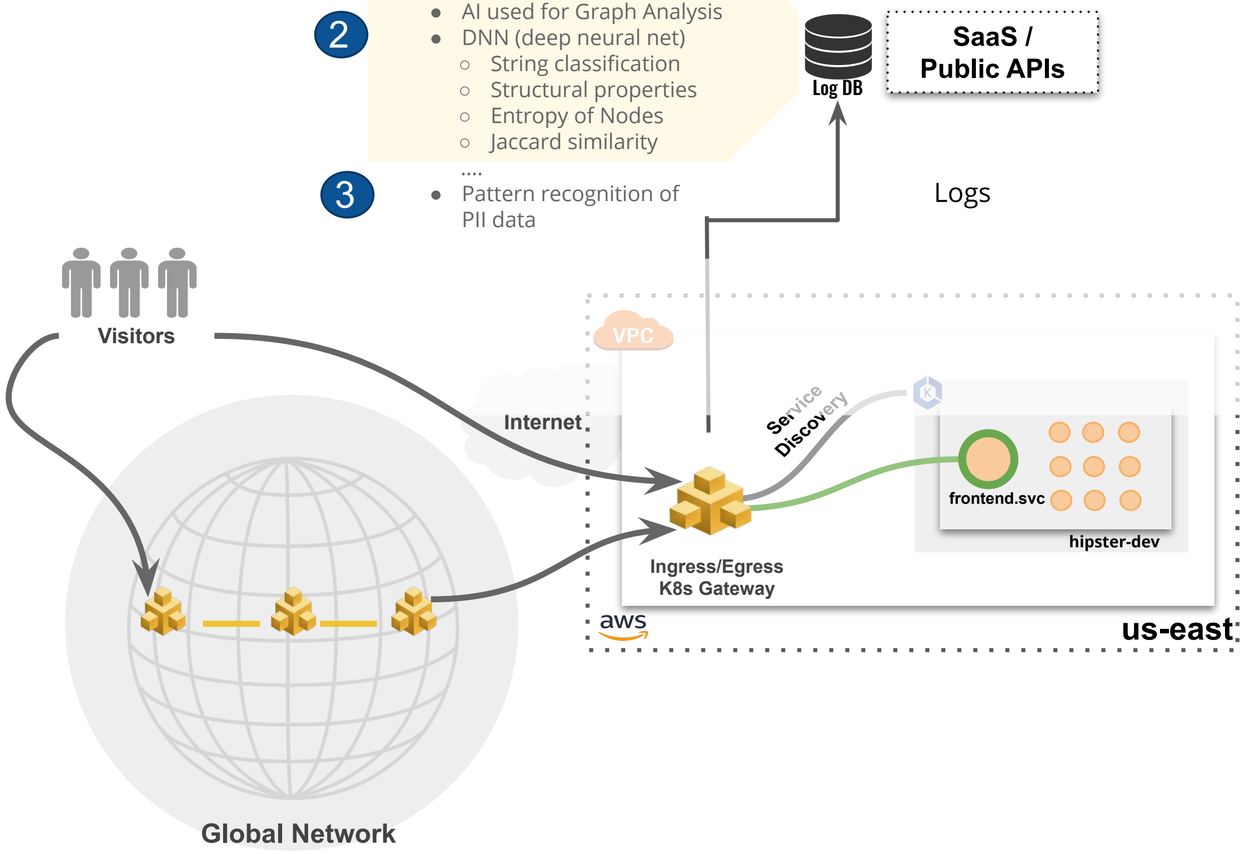
3 Analyze API schema,
Data shared in APIs response
Analyze if data is PII or not

1
- TimeSeries Analysis

2
- AI used for Graph Analysis
- DNN (deep neural net)
  - String classification
  - Structural properties
  - Entropy of Nodes
  - Jaccard similarity

….

3
- Pattern recognition of PII data

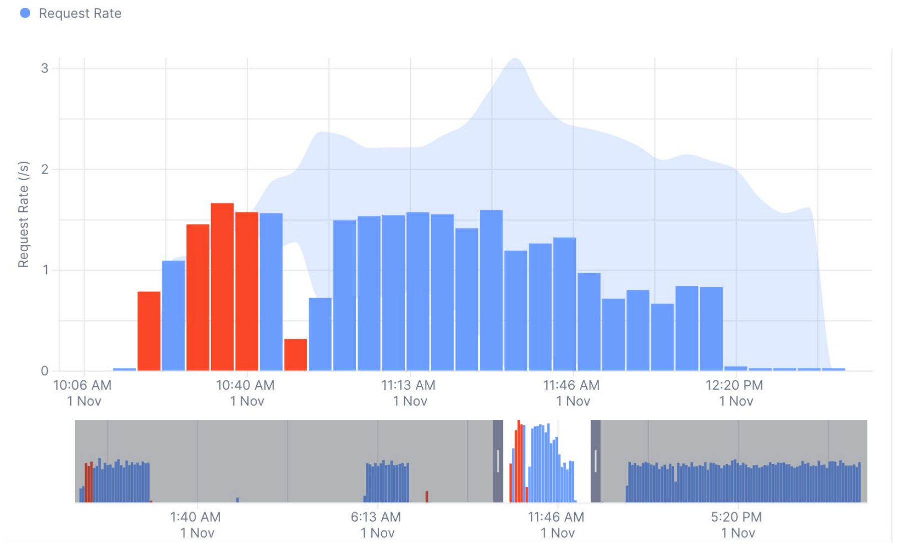


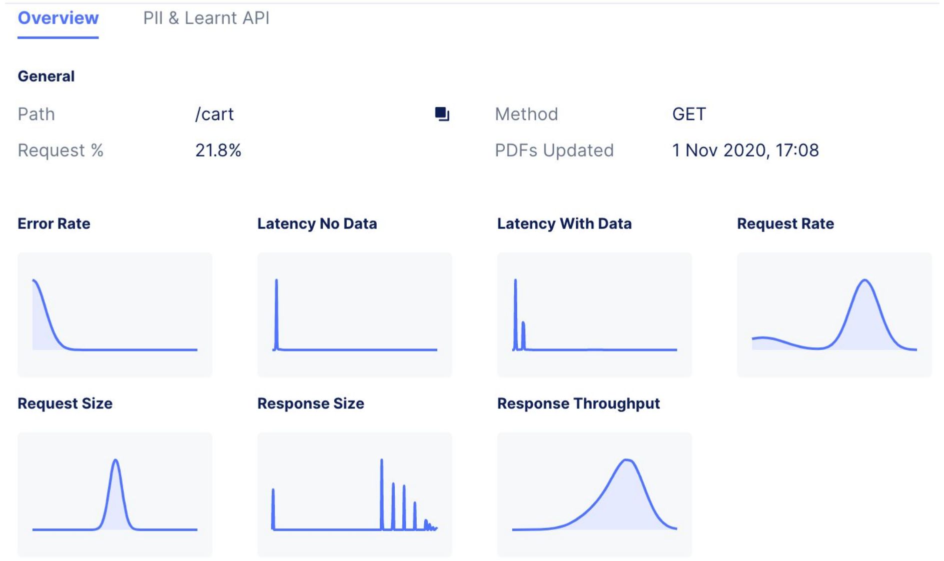
Analyze APP/API communication

Analyze

Baseline normal App-to-App communication patterns

Baseline normal API-to-API requests/response patterns





General

Path	/cart	Method	GET
Request %	21.8%	PDFs Updated	1 Nov 2020, 17:08

Volterra

Analyze APP/API data patterns

Analyze

Learn API Schema to determine data shared by API

```
"required": [
  "credit_card_expiration_month",
  "credit_card_cvv",
  "email",
  "credit_card_expiration_year",
  "city",
  "street_address",
  "zip_code",
  "state",
  "credit_card_number",
  "country"
],
"properties": {
  "credit_card_expiration_month": {
```

Identify what PII data is shared

PII

Field	Section	Type	Example
email	Request Body	Email	stage@ves.io test@ves.io someone@example.com

Determine data patterns to identify PII data

```
"credit_card_expiration_month": {
  "type": "string",
  "description": "Integer",
  "pattern": "-?\\d+"
},
```

```
"credit_card_number": {
  "type": "string",
  "description": "Integer",
  "pattern": "-?\\d+"
```

Volterra

Secure APP & API - Automated Policy Generation

Secure

DevOps

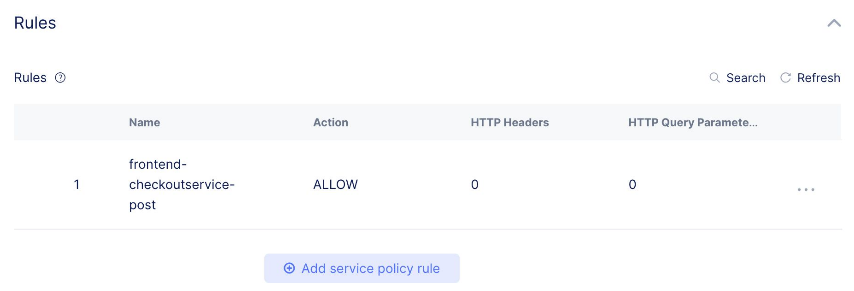
SecOps

1. Automatically generate policy based on App-to-App and Api-to-Api graph

```
Creating policy between public-frontend-post (client) ----> frontend (server)
allow POST http method for paths:
/cart/checkout
/cart
/setCurrency
```

2. Create API policy to block API response based on API Data pattern learnt

3. Create API policy to block API request based on PDF of Request Rate



Rules

Rules ⓘ

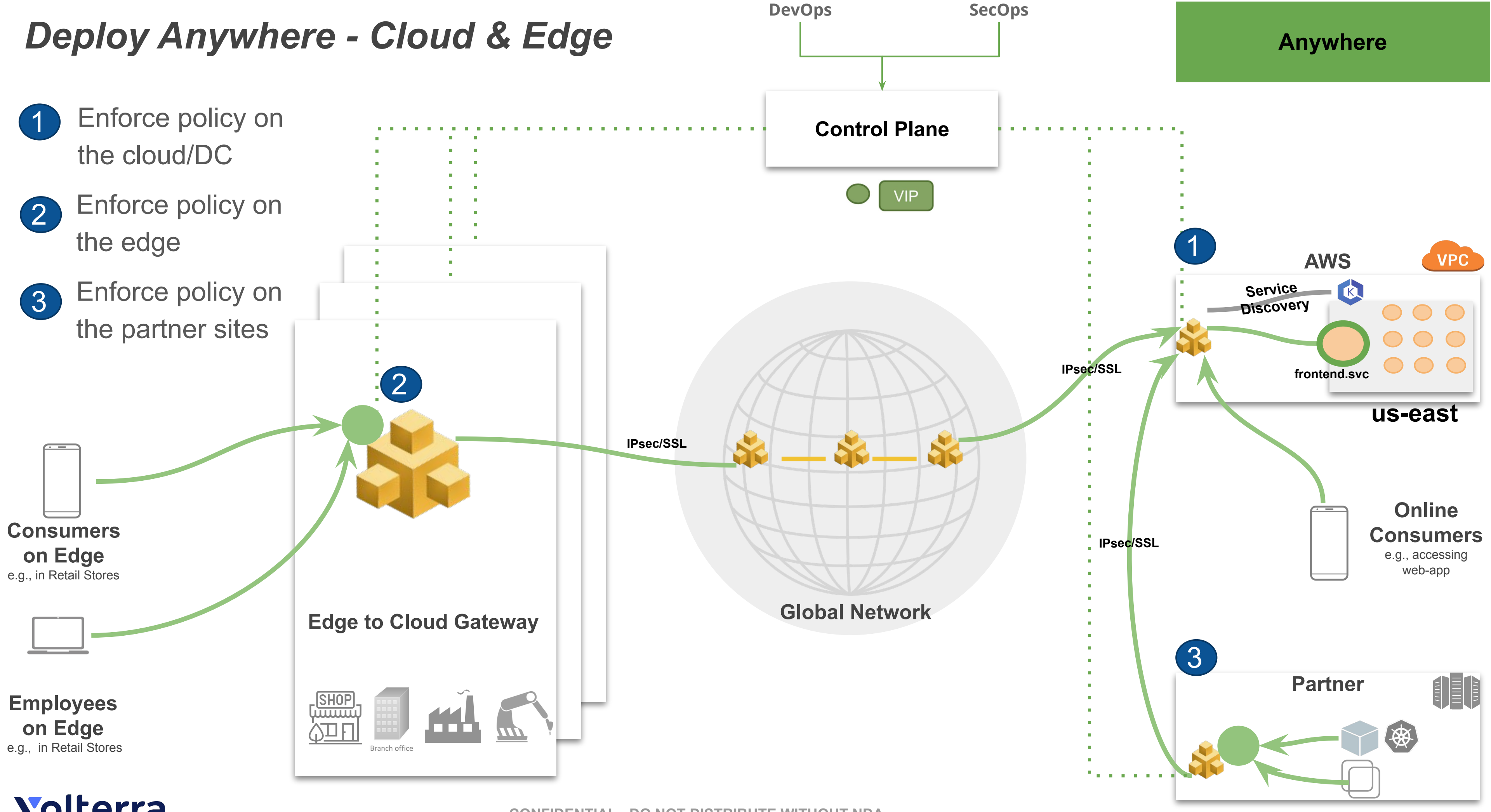
Search Refresh

	Name	Action	HTTP Headers	HTTP Query Paramete...	
1	frontend-checkoutservice-post	ALLOW	0	0	...

Volterra

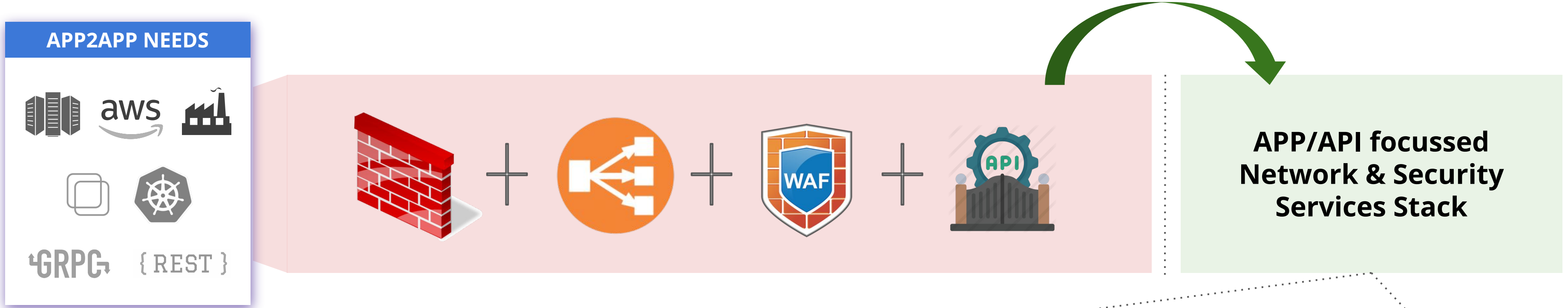
Deploy Anywhere - Cloud & Edge

1. Enforce policy on the cloud/DC
2. Enforce policy on the edge
3. Enforce policy on the partner sites



Volterra

A NEW APPROACH IS NEEDED FOR APP/API Security...



Discover

Analyze

Secure

Deployable Anywhere

Thank You!

Q&A

Volterra

Enable the Cloud for Everyone, Everywhere