Building Your Own WAF as a Service and Forgetting about False Positives

Juan Berner

About me

- Lead security developer @Booking.com
- Twitter: @89berner
- medium.com/@89berner

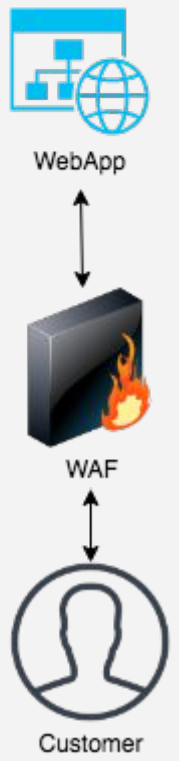
WAF?

- Web Application Firewall
- Mainly used to protect against Application Attacks
- SQLi, RCE, Protocol Violations, Rate Limiting ...

Deployment mode - Inline

- **Pros:**
  - **Traffic inspection**
  - **Ability to block**
  - **Transparent for web servers**

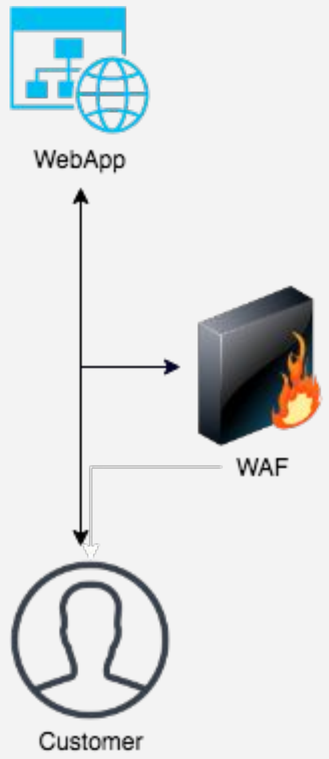
- **Cons:**
  - **Network placement**
  - **Latency**



Deployment mode - Out of band

- **Pros:**
  - **Traffic inspection**
  - **Transparent for web servers**
  - **Simpler network placement**

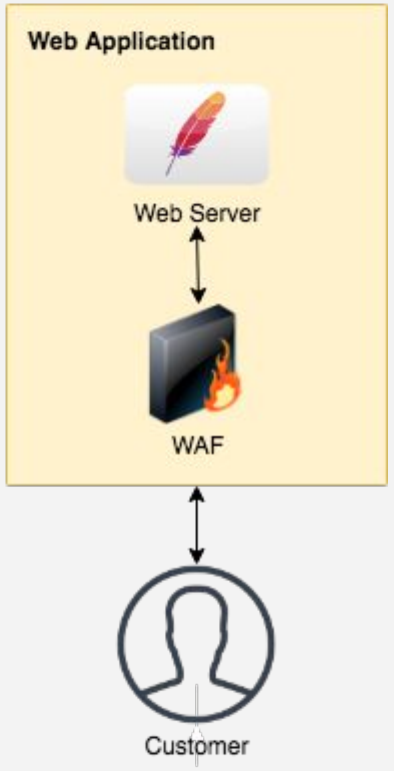
- **Cons:**
  - **Can’t block attacks**
  - **PFS**



Deployment mode - Agent

- **Pros:**
  - **Easier network placement**
  - **Simple to scale**

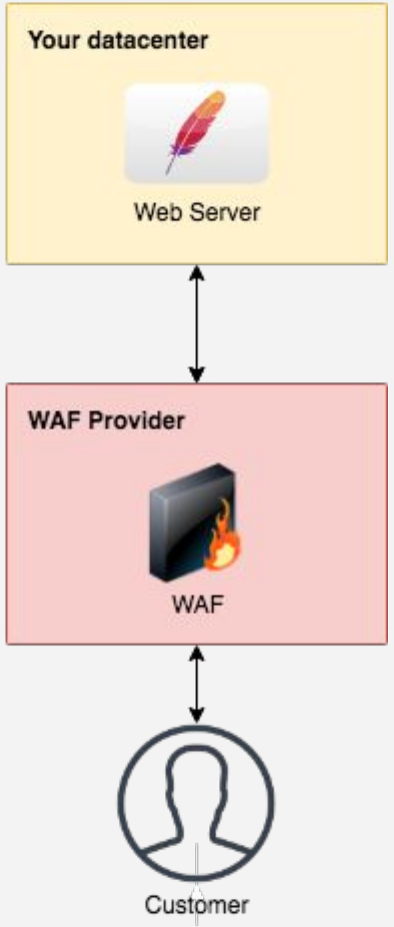
- **Cons:**
  - **More invasive on deployment environment**
  - **Can be less efficient on resource allocation**



Deployment mode - Cloud

- **Pros:**
  - **Simple to setup and scale**
  - **Network effect**

- **Cons:**
  - **Out of your control**
  - **Latency**



Caveats with typical WAF Solutions

- Network placement
- False positive rate
- Lack of control from developers

A challenging environment

- No acceptance for false positives
- Reluctance towards commercial appliances
- Blocking could only happen through the Application
- Latency would not be acceptable

Building the WAF as a Service

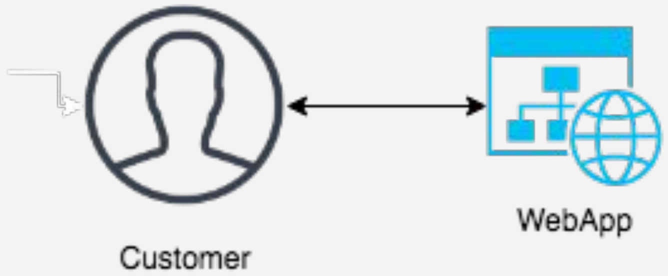
- Removes false positives by having an understanding of the application context
- No need for an appliance, just add an API call
- Blocking behaviour is decided by the application
- Ability to avoid latency for regular users

How could you build one?

- Open source components already exist
- Creating a log processing pipeline
- Building a WAF API
- Library for logs and calling API

Study case: Simple web application

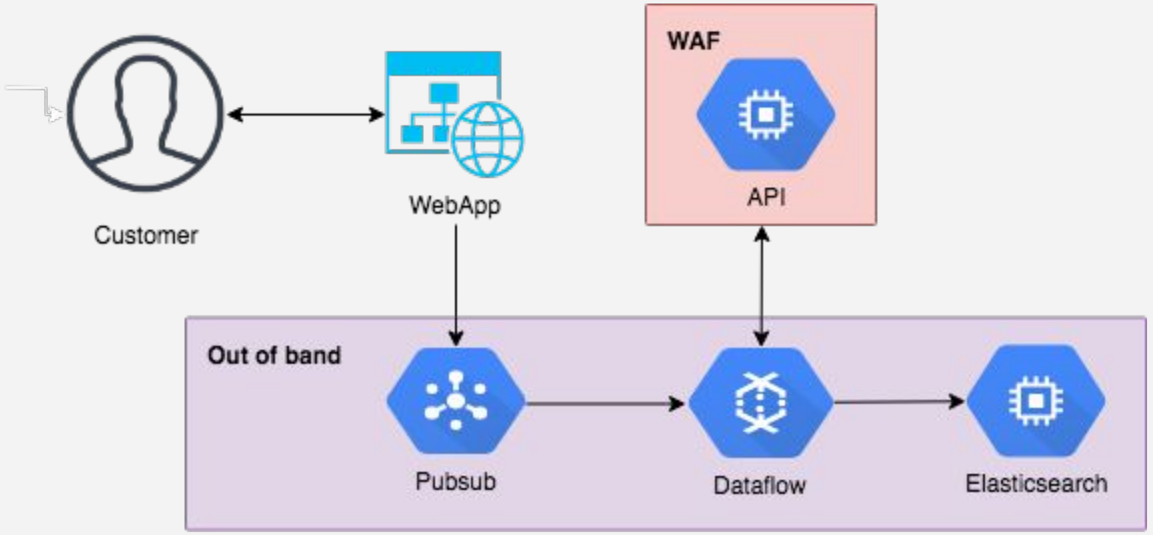
- Setup in Google Cloud
- Simple Flask Application
- Code available in github



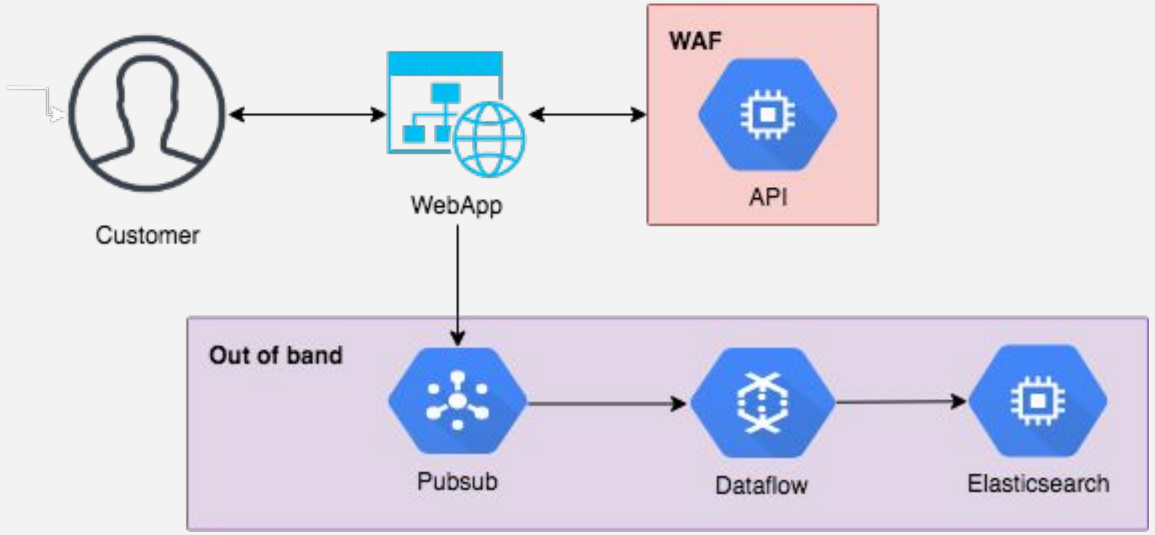
Deployment mode?

- **Let’s compare**

Out of band mode



Inline mode


Customer
WebApp
WAF
API
Out of band
Pubsub
Dataflow
Elasticsearch

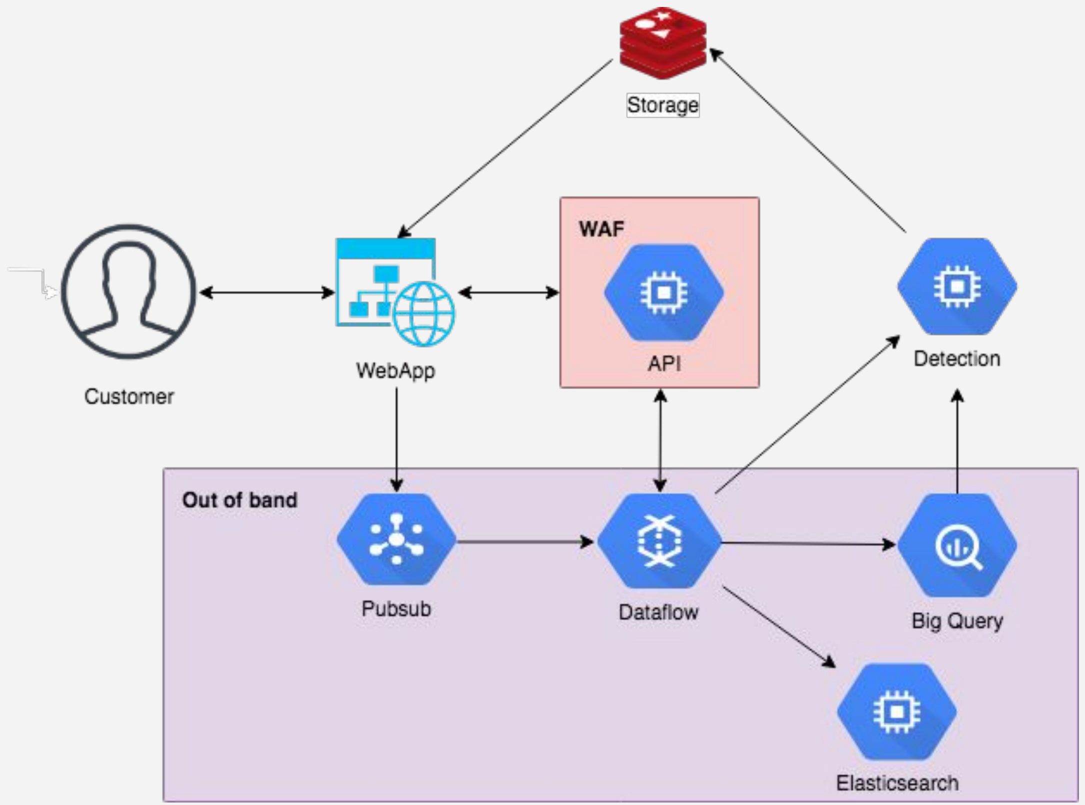
Every application is different

- Threat model
- FP tolerance
- Risk acceptance

Finding a middle ground

- Out of band mode removes latency concerns on users
- Inline mode provides security by blocking attacks
- Could we get the best of both worlds?

Hybrid mode


Storage
Customer
WebApp
WAF
API
Detection
Out of band
Pubsub
Dataflow
Big Query
Elasticsearch

Components - Web application

- Can decide which mode to work on
  - Inline
  - Out of band

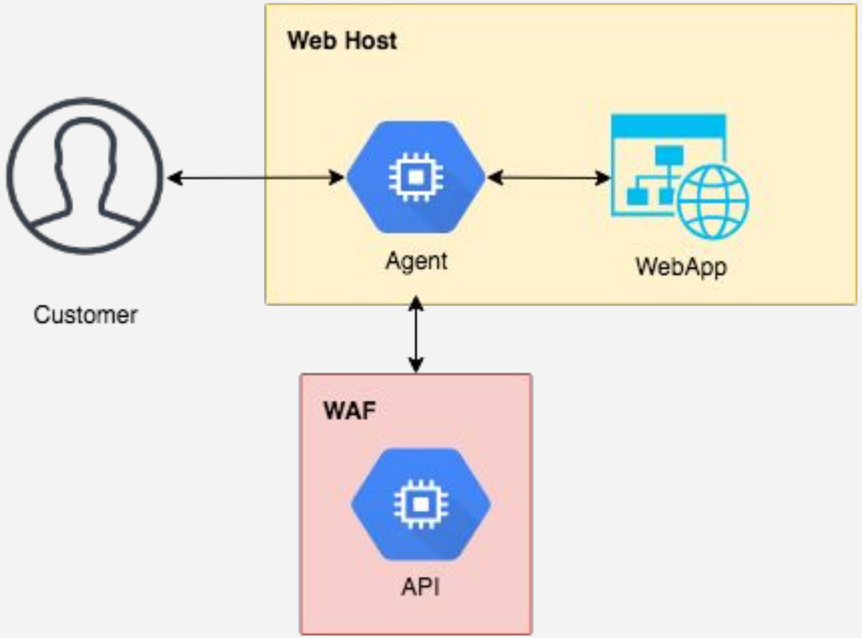
- Sends logs with partial request data encrypted

Example: Flask API

Components - Agent

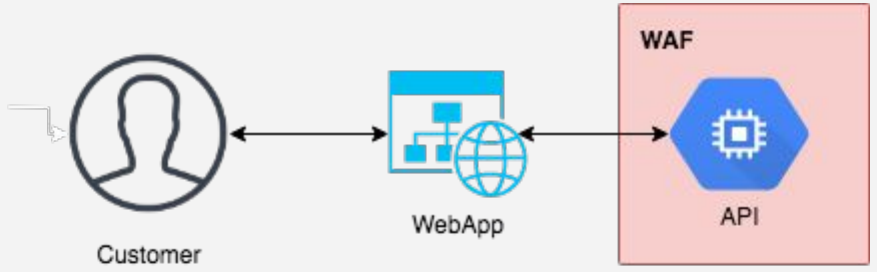
- Acts as a proxy to Web Application
- Minimal footprint
- Application agnostic
- Gets settings from application

Components - Agent



Components - Library

- Simpler to implement
- Will be tied to Application framework
- Inherent risks
- Strategy for this talk



Components - Historical database

- Historical activity
- Business value
- Patterns of behaviour for FP

Components - State store

- Allows to store configuration
- Ideally fast lookup for caching

Components - Log streaming

- Streaming pipeline
- Web requests are encapsulated and sent through it

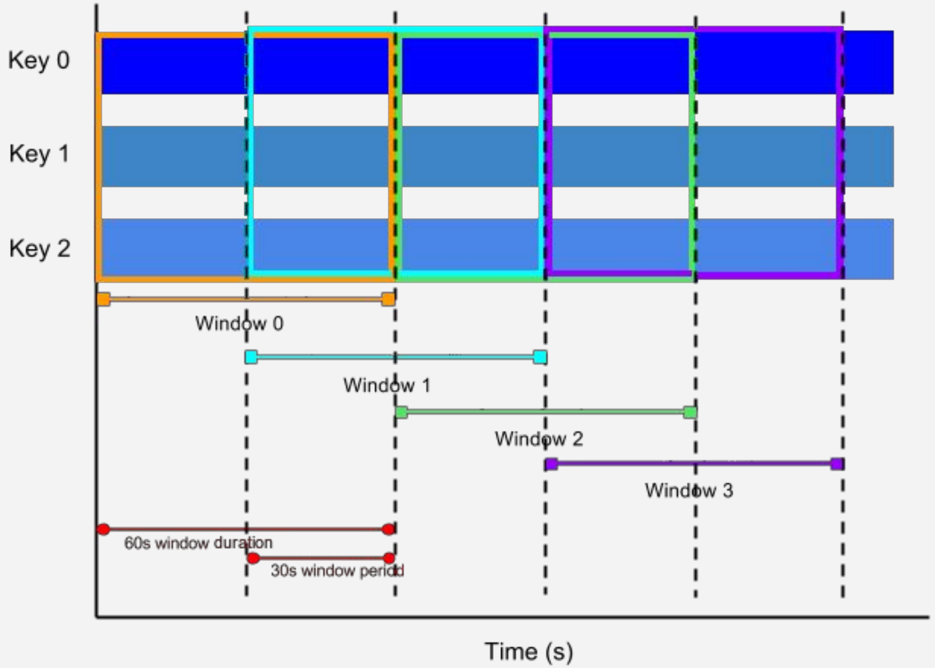
Google PusbSub

Components - Log processing

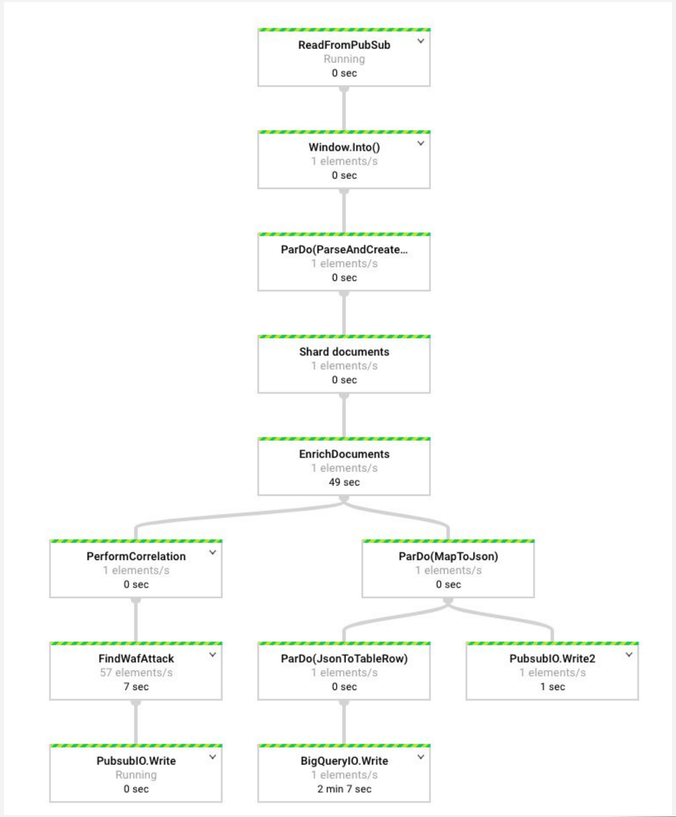
- Replays events not in line against WAF
- Calculates scores through windows of time

Google Dataflow

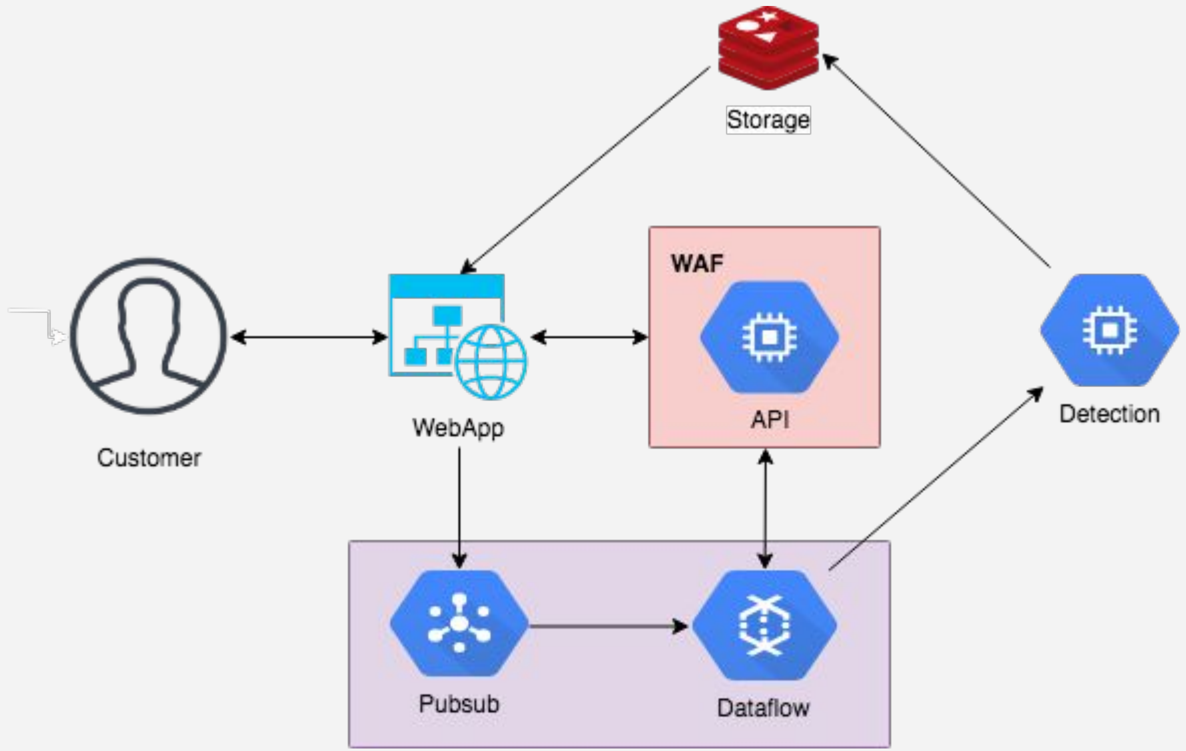
Components - Log processing



Components - Log processing



Components - Log processing



Components - WAF service

- Pluggable architecture
- Parallel nature of their components
- Applications can decide how to react

Components - WAF service

- Open source components
  - Modsecurity
  - Naxsi

Components - WAF service

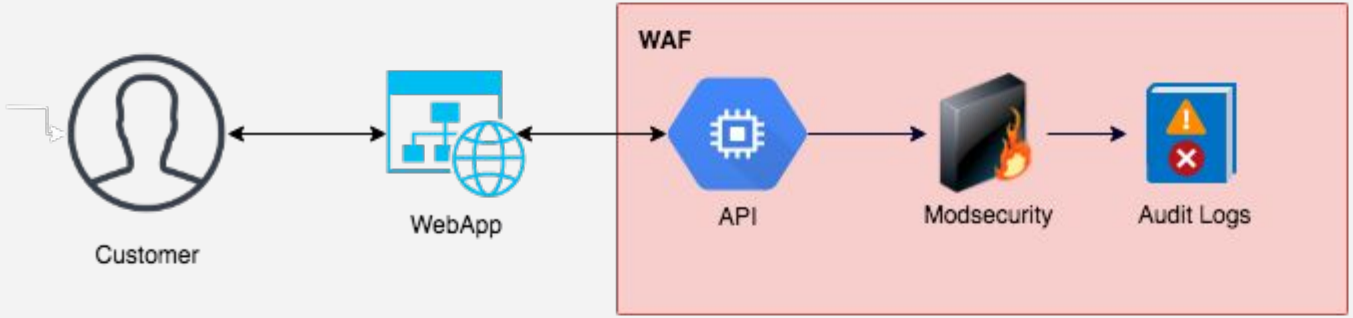
- Custom modules
  - Apply custom business logic
  - Implement simple services
    - Rate limiting
    - Rule engine for blocking
  - ML models

Components - WAF service

- Proprietary software or appliances
    - Reduced complexity of installation
    - Simple way of evaluation

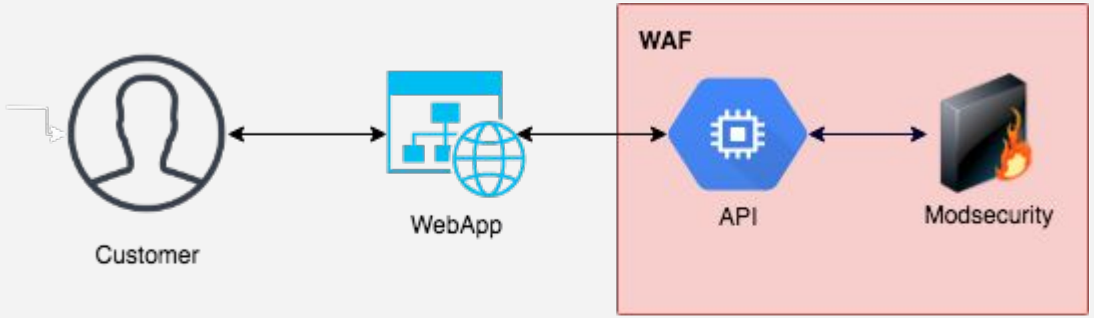
WAF service - Example: Modsecurity

- Could be made api driven through libModSecurity
- Can run on Apache HTTP Server or NGINX
- Results are written as logs



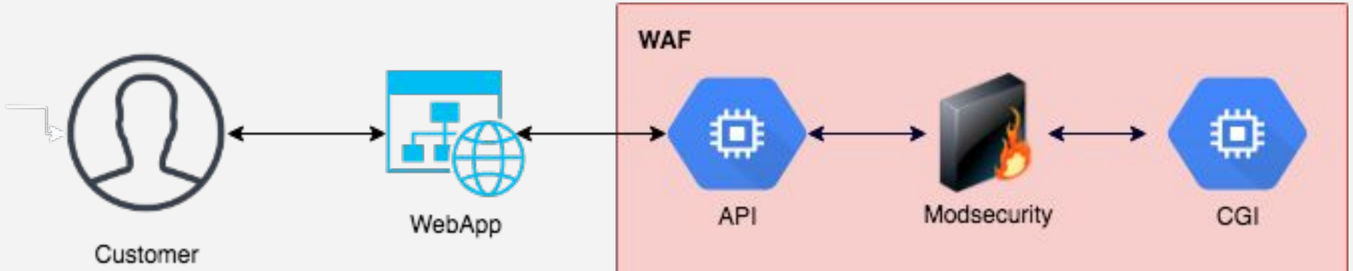
WAF service - Modsecurity as an API

- ***SecRule REMOTE_ADDR "@unconditionalMatch" "phase:4,id:999434,prepend: ...***

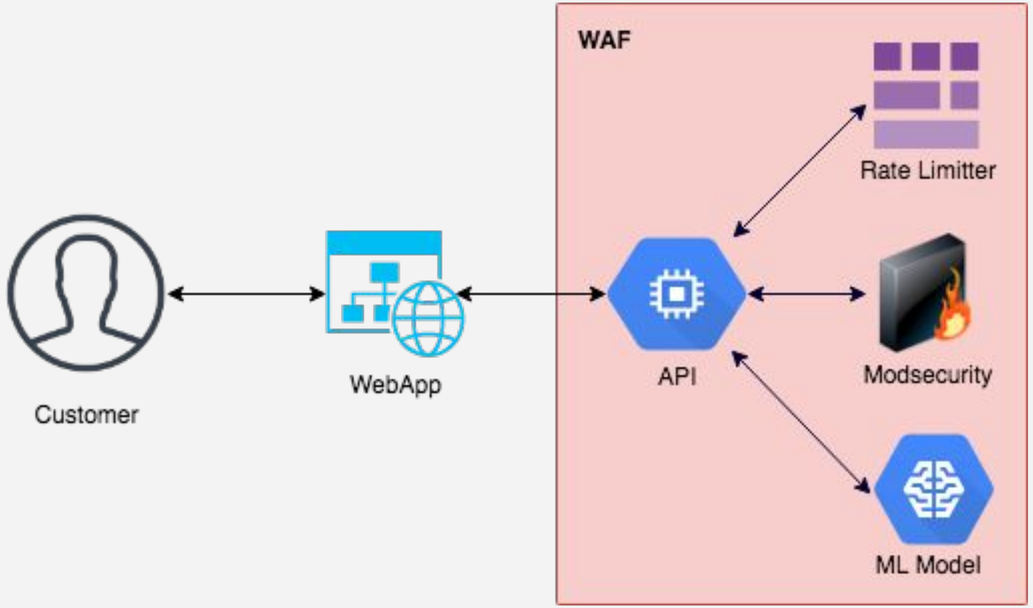


WAF service - Modsecurity as an API

- **Implementing response body analysis**
- **Body is sent to CGI for replay**



WAF service

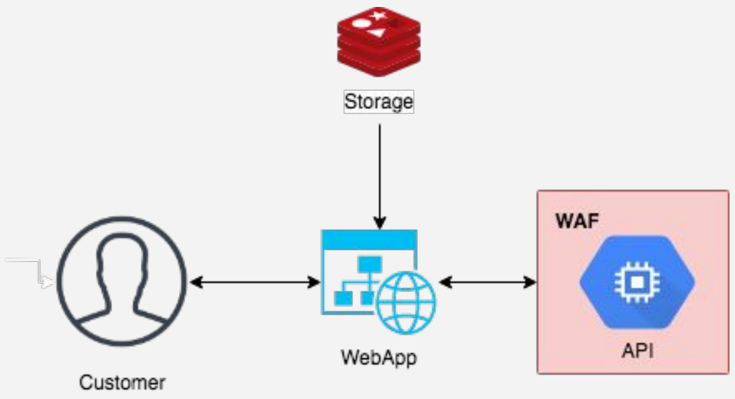

WAF
Rate Limitter
Customer
WebApp
API
Modsecurity
ML Model

How to block?

- We decide when to send traffic to the WAF
- Manually or automatically decided

Traffic routing

- Fingerprint based routing
  - Blocks based on scores
  - IP, client_id, combinations, 0day fingerprints..
  - Added automatically or manually



Traffic routing

- **Net block based routing**
  - **ISP**
  - **Hosting providers**
  - **Tor exit nodes / Proxies**

Traffic routing

- Virtual Patching
  - Always route particular vulnerable endpoints
  - Select for combination of parameters if needed
  - Example: website.com/?*vuln_param*=

FP rate management

- Detection FP vs blocking FP
- Key to allow blocking without impacting users
- Acceptable rate might change per application

FP rate management

- Business logic
  - How trustworthy is a user/ip?
  - Key business activity
  - What would be the impact on blocking them

FP rate management

- Historical Analysis
  - How normal is this type of request for this endpoint?
  - How does this user compare with others
  - How common are detection FP in this endpoint

FP rate management

- Context analysis
    - How many times have they triggered a FP
    - How many requests have they sent

FP rate management

- Example: Sleep(
  - *message="I sleep(1 or 2 days)"*
    - Might be detected as SQLI
    - Probability of FP is independent from each other

FP rate management

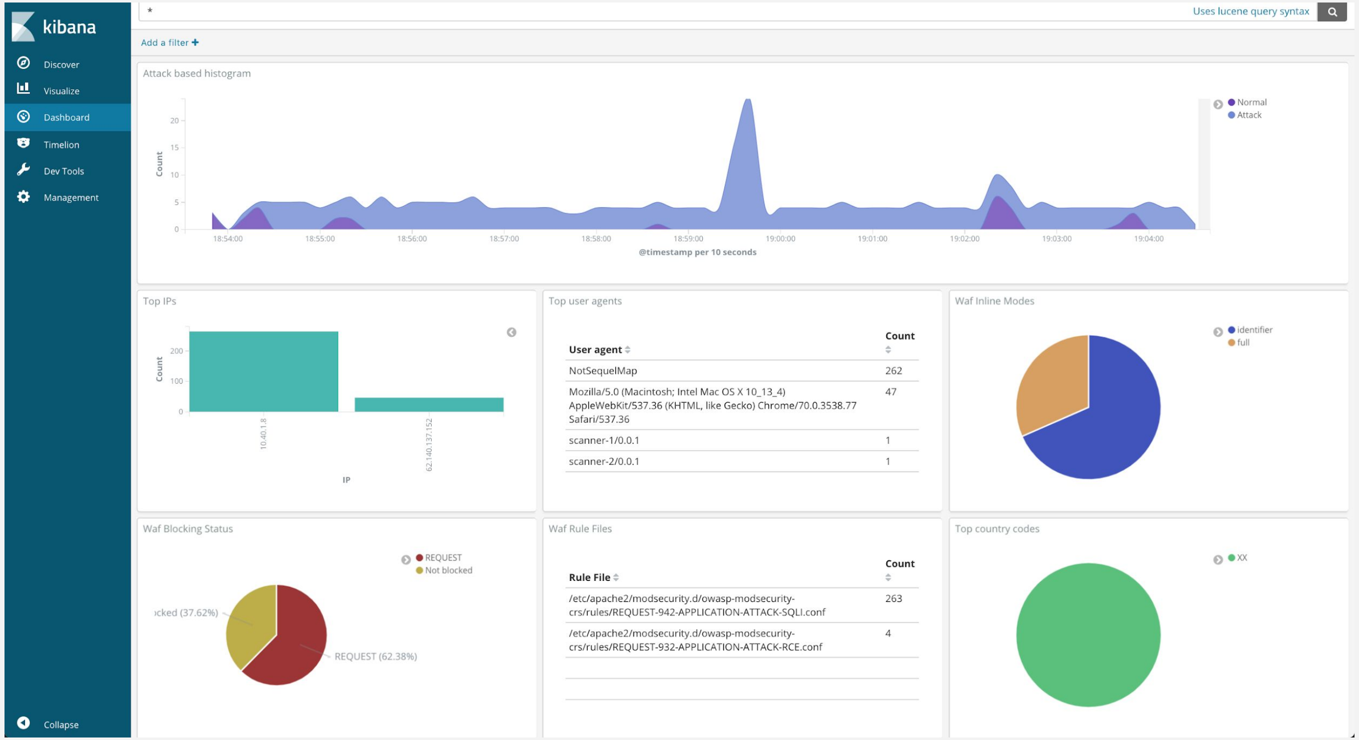
- Independant SQLI FP rate: 0.1%
- Our aim, 0.00001% (0.01^5) => Score 5
- Block can happen only for SQLis
- Aimed at attacks that need volume

Components - Visualisation

- Easily understand activity
- Visibility on attacks
- Performance metrics

Example: ELK

Components - Visualisation



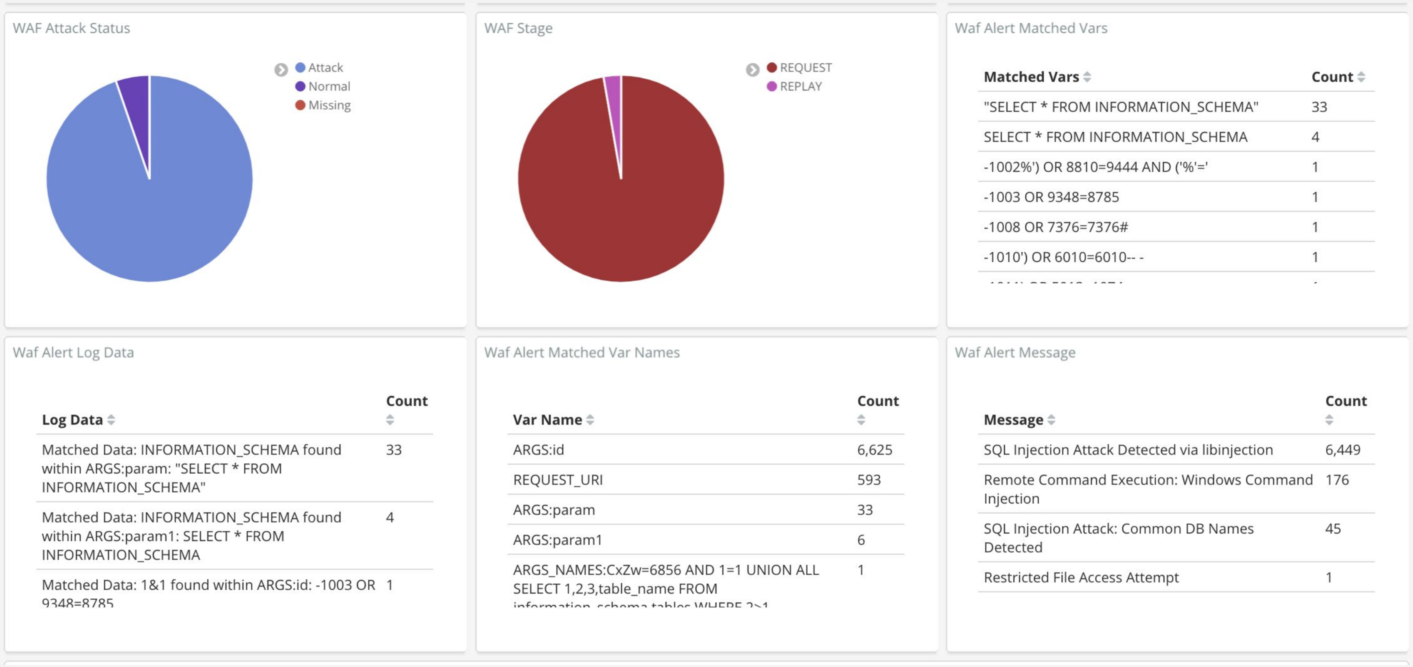
Top user agents

User agent	Count
NotSequelMap	262
Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_4) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/70.0.3538.77 Safari/537.36	47
scanner-1/0.0.1	1
scanner-2/0.0.1	1

Waf Rule Files

Rule File	Count
/etc/apache2/modsecurity.d/owasp-modsecurity-crs/rules/REQUEST-942-APPLICATION-ATTACK-SQLI.conf	263
/etc/apache2/modsecurity.d/owasp-modsecurity-crs/rules/REQUEST-932-APPLICATION-ATTACK-RCE.conf	4

Components - Visualisation



Waf Alert Matched Vars

Matched Vars	Count
"SELECT * FROM INFORMATION_SCHEMA"	33
SELECT * FROM INFORMATION_SCHEMA	4
-1002%') OR 8810=9444 AND ('%'='	1
-1003 OR 9348=8785	1
-1008 OR 7376=7376#	1
-1010') OR 6010=6010-- -	1

Waf Alert Log Data

Log Data	Count
Matched Data: INFORMATION_SCHEMA found within ARGS:param: "SELECT * FROM INFORMATION_SCHEMA"	33
Matched Data: INFORMATION_SCHEMA found within ARGS:param1: SELECT * FROM INFORMATION_SCHEMA	4
Matched Data: 1&1 found within ARGS:id: -1003 OR 9348=8785	1

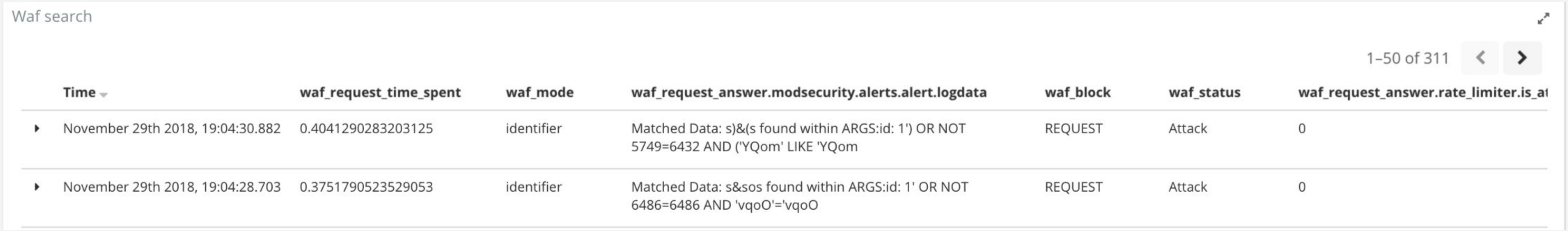
Waf Alert Matched Var Names

Var Name	Count
ARGS:id	6,625
REQUEST_URI	593
ARGS:param	33
ARGS:param1	6
ARGS_NAMES:CxZw=6856 AND 1=1 UNION ALL SELECT 1,2,3,table_name FROM information_schema.tables WHERE 2>1	1

Waf Alert Message

Message	Count
SQL Injection Attack Detected via libinjection	6,449
Remote Command Execution: Windows Command Injection	176
SQL Injection Attack: Common DB Names Detected	45
Restricted File Access Attempt	1

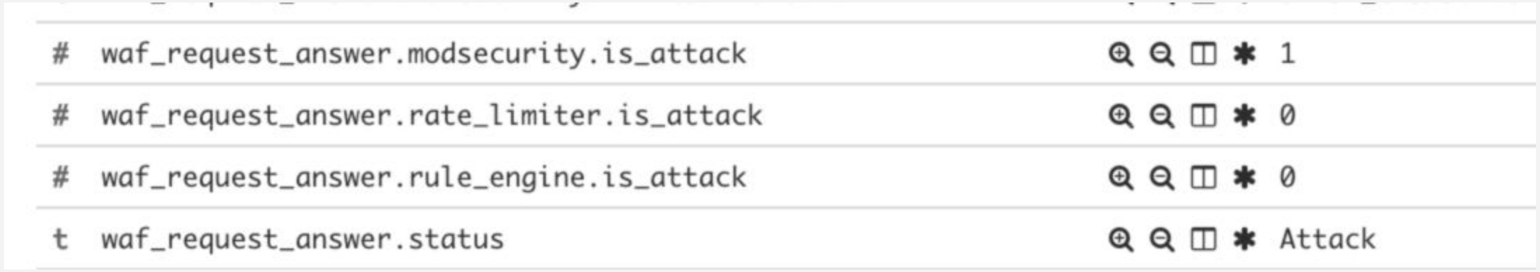
Components - Visualisation



Waf search

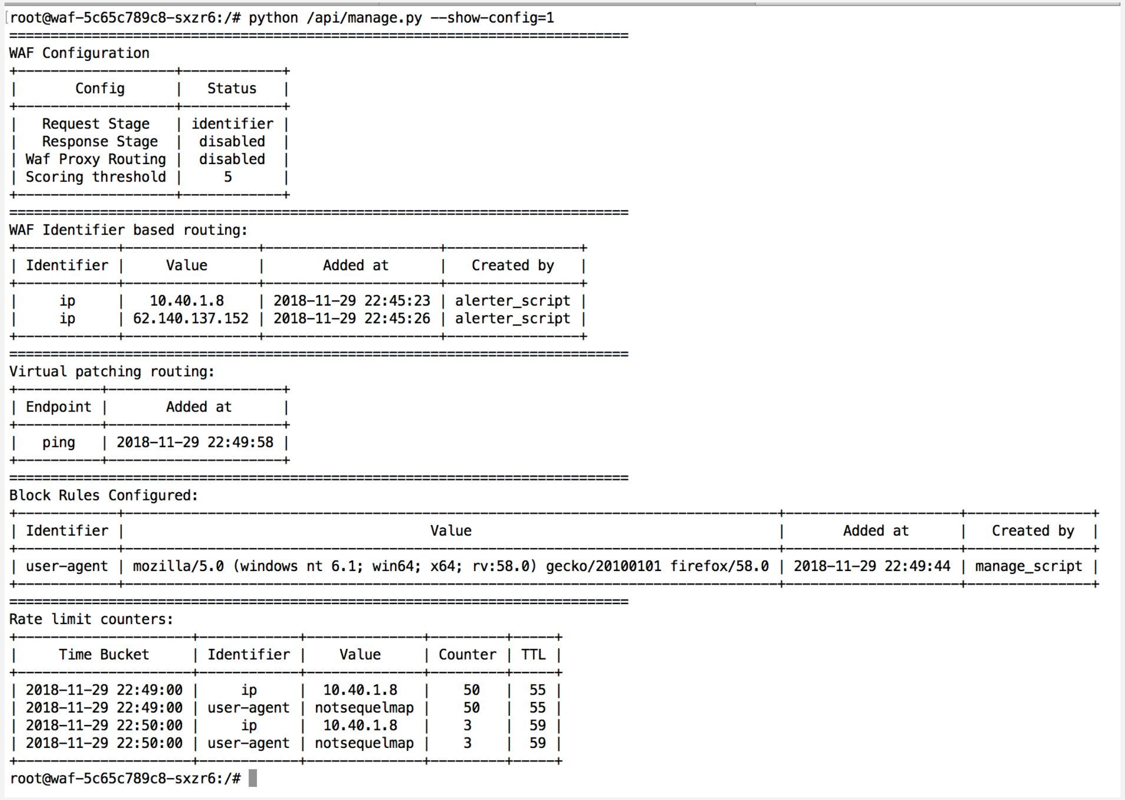
1–50 of 311

	Time	waf_request_time_spent	waf_mode	waf_request_answer.modsecurity.alerts.alert.logdata	waf_block	waf_status	waf_request_answer.rate_limiter.is_at
▸	November 29th 2018, 19:04:30.882	0.4041290283203125	identifier	Matched Data: s)&(s found within ARGS:id: 1') OR NOT 5749=6432 AND ('YQom' LIKE 'YQom	REQUEST	Attack	0
▸	November 29th 2018, 19:04:28.703	0.3751790523529053	identifier	Matched Data: s&sos found within ARGS:id: 1' OR NOT 6486=6486 AND 'vqoO'='vqoO	REQUEST	Attack	0



```
#  waf_request_answer.modsecurity.is_attack     1
#  waf_request_answer.rate_limiter.is_attack    0
#  waf_request_answer.rule_engine.is_attack     0
t  waf_request_answer.status                    Attack
```

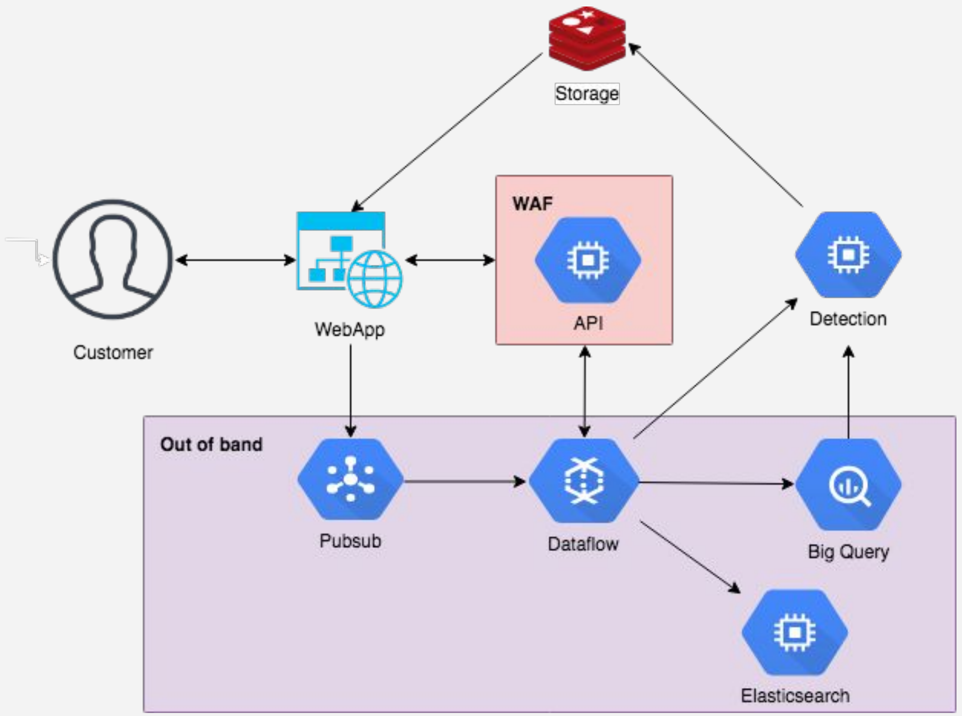
Components - Visualisation



```
root@waf-5c65c789c8-sxzr6:/# python /api/manage.py --show-config=1
===========================================================================
WAF Configuration
+-------------------+------------+
|       Config      |   Status   |
+-------------------+------------+
|   Request Stage   | identifier |
|   Response Stage  |  disabled  |
| Waf Proxy Routing |  disabled  |
| Scoring threshold |     5      |
+-------------------+------------+
===========================================================================
WAF Identifier based routing:
+------------+-----------------+---------------------+----------------+
| Identifier |      Value      |       Added at      |   Created by   |
+------------+-----------------+---------------------+----------------+
|     ip     |    10.40.1.8    | 2018-11-29 22:45:23 | alerter_script |
|     ip     | 62.140.137.152  | 2018-11-29 22:45:26 | alerter_script |
+------------+-----------------+---------------------+----------------+
===========================================================================
Virtual patching routing:
+----------+---------------------+
| Endpoint |       Added at      |
+----------+---------------------+
|   ping   | 2018-11-29 22:49:58 |
+----------+---------------------+
===========================================================================
Block Rules Configured:
+------------+-------------------------------------------------------------------------------+---------------------+---------------+
| Identifier |                                     Value                                     |       Added at      |   Created by  |
+------------+-------------------------------------------------------------------------------+---------------------+---------------+
| user-agent | mozilla/5.0 (windows nt 6.1; win64; x64; rv:58.0) gecko/20100101 firefox/58.0 | 2018-11-29 22:49:44 | manage_script |
+------------+-------------------------------------------------------------------------------+---------------------+---------------+
===========================================================================
Rate limit counters:
+---------------------+------------+--------------+---------+-----+
|     Time Bucket     | Identifier |    Value     | Counter | TTL |
+---------------------+------------+--------------+---------+-----+
| 2018-11-29 22:49:00 |     ip     |  10.40.1.8   |    50   |  55 |
| 2018-11-29 22:49:00 | user-agent | notsequelmap |    50   |  55 |
| 2018-11-29 22:50:00 |     ip     |  10.40.1.8   |    3    |  59 |
| 2018-11-29 22:50:00 | user-agent | notsequelmap |    3    |  59 |
+---------------------+------------+--------------+---------+-----+
root@waf-5c65c789c8-sxzr6:/#
```

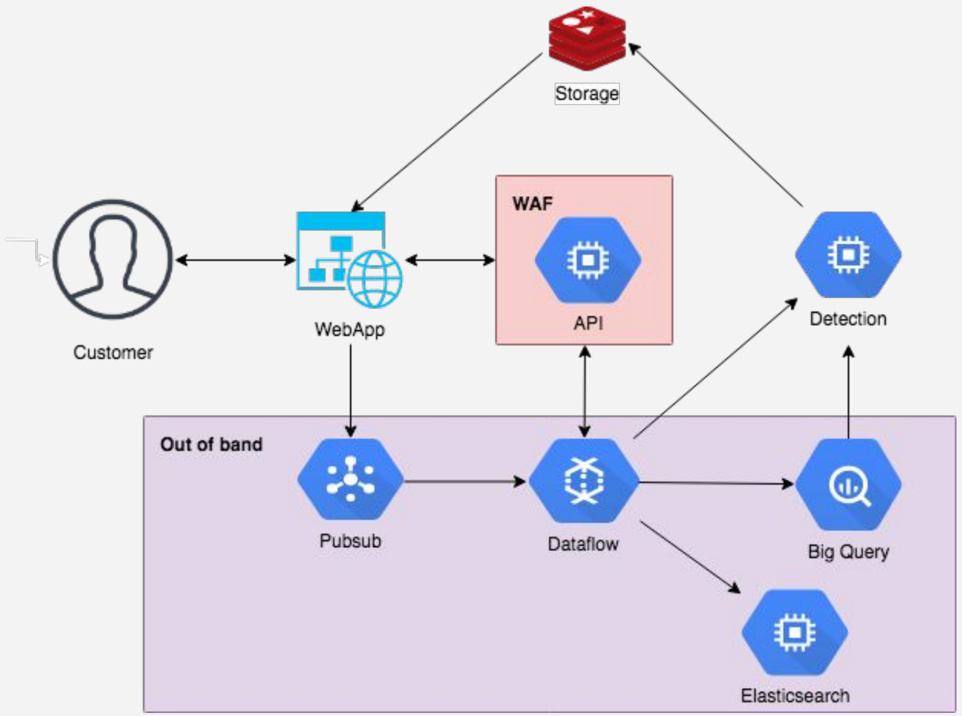
Hybrid mode

- **Benefits**
  - **Can reduce latency**
  - **Flexibility**
  - **FP rate can be decided**



Hybrid mode

- Caveats
  - Delayed response time for blocking
  - Complexity



ONE DOES NOT SIMPLY
PERFORM A LIVE DEMO
memegenerator.net

What now?

- Try it!
- https://github.com/89berner/waf-api-talk
- *git clone https://github.com/89berner/waf-api-talk && cd waf-api-talk; ./setup $YOUR_GCP_PROJECT*
- Questions?