doit

CONF42

No WAFs

Don’t use a Web Application Firewall, and when you should, anyway.

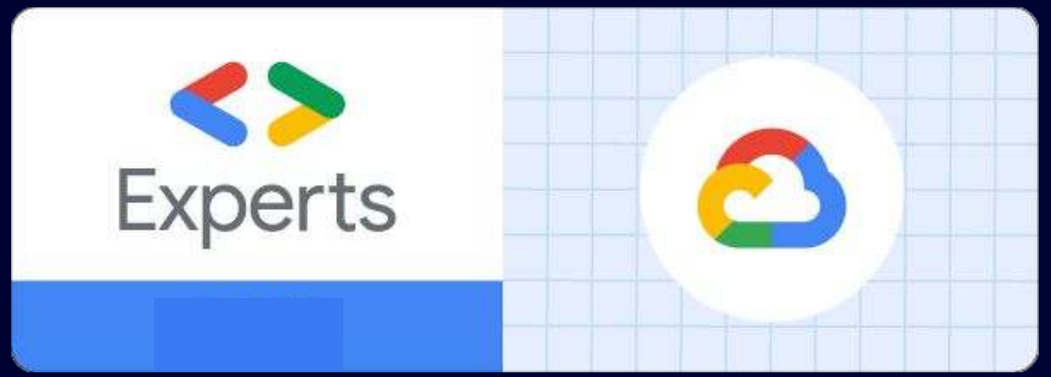
Joshua Fox

joshua@doit.com

Senior Cloud Architect


Google Cloud Certified
PROFESSIONAL
CLOUD ARCHITECT


aws
CERTIFIED
Solutions Architect
Professional


Experts

<<

doit

Azure

Google Cloud

aws

Article

No WAFs

No WAFs

Joshua Fox · Follow

Published in DoiT International · 6 min read · Apr 4, 2024

Don’t use a Web Application Firewall, and when you should, anyway

Your security team has just painted a grim picture of potential cyber threats, and you’re aware that your web application is a minefield of vulnerabilities. Fixing these security issues seems like a task that could take between forever and never.

Scenario

CISO warns about threats

Scenario

Quick! A solution!

Scenario

Get a WAF!

What is a WAF?

- A service that (tries to) protect your webapp
- Later we'll discuss architecture and specific functionality

Drivers for getting a WAF

Hacker Attack

Fire drill!

Penetration Test

- So, so easy to break in.
- The state of the art is … not good

Urgency

Quick! We need a solution! Now!
(And forever)

Expertise

No expertise now, … or ever

Outside Requirement/Audit

- Government
- Customer
- Partner
- Standards body

Security Blanket

Web Threats

OWASP Top Ten

Walkthrough: Cross Site Scripting

`nowaf.joshuafox.com`

Working vulnerability:

`withwaf.joshuafox.com`

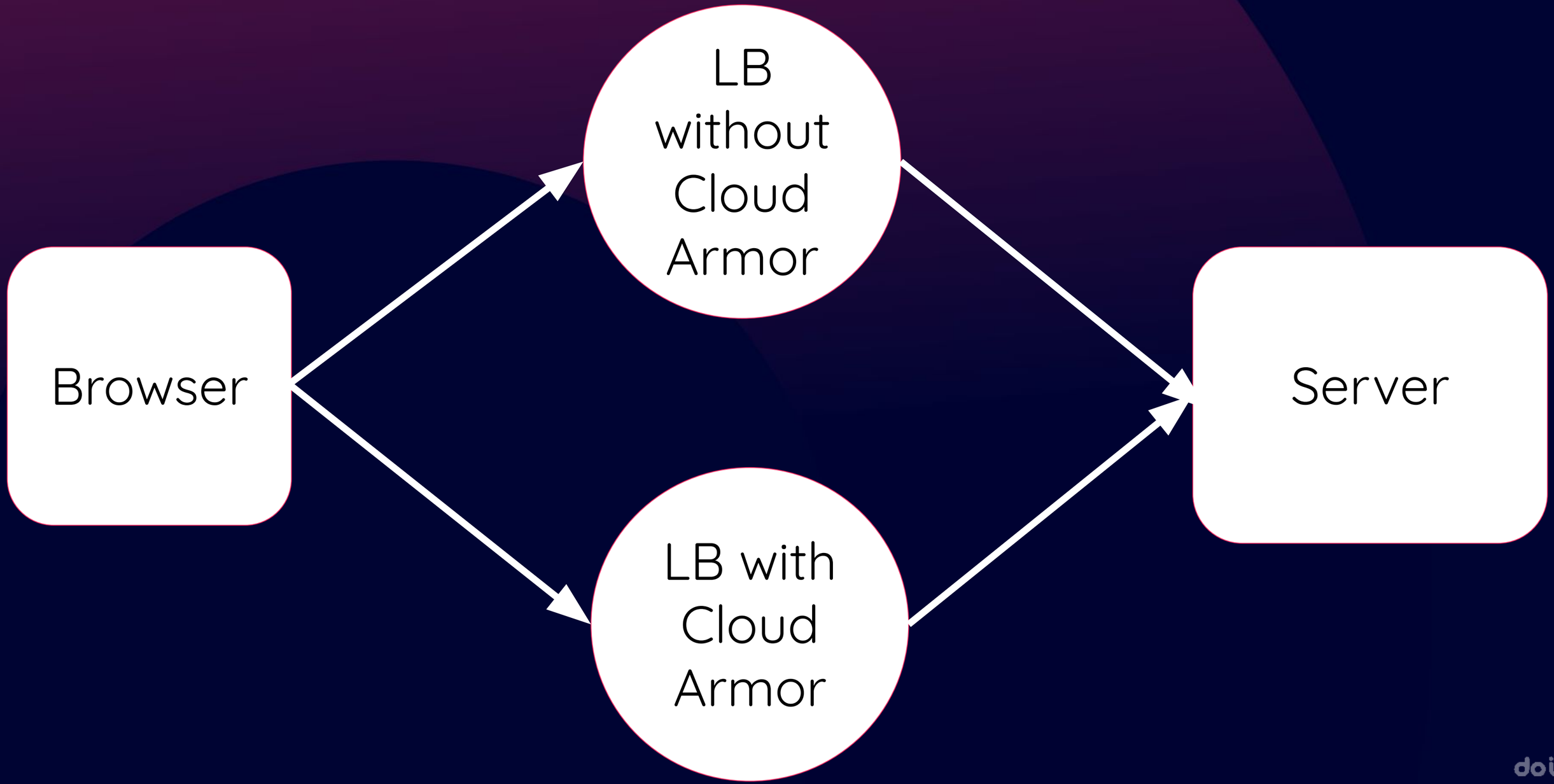
Without WAF

Try to create a game where my name is executable

```
<script>alert("HACKED YOU")</script>
```

(Actual hack ***steals your password***!)

Demo WAF Architecture



Make it Safe!

```
$('#chatLog')
..append(message.name))
```

WAF protects the broken one!

```
$('#chatLog')
  .append(DIV(null,
   message.name))
```

A simple chat message is executed

User-name is runnable JavaScript

```
<script>alert("HACKED YOU")</script>
```

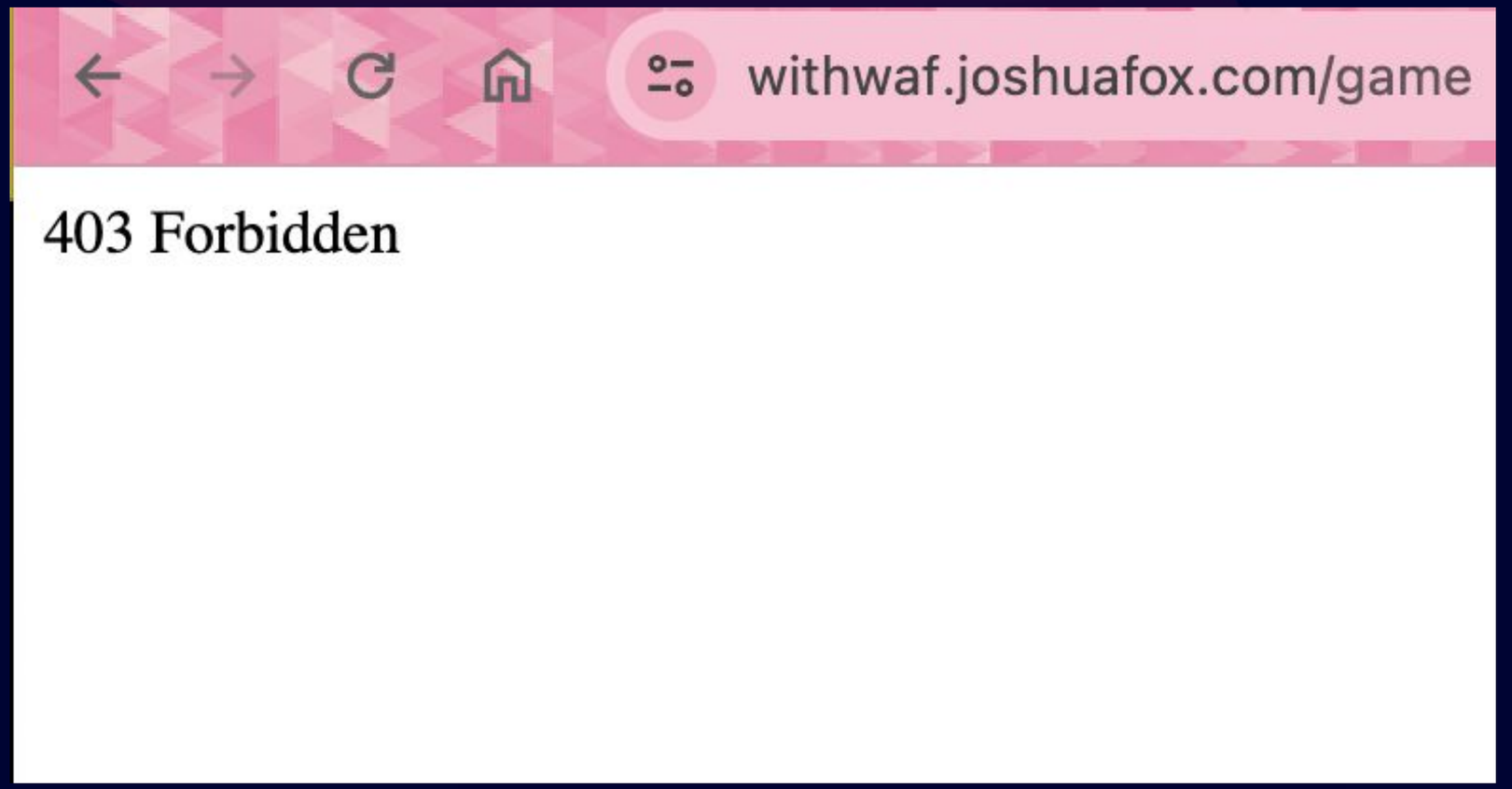
With WAF

Try to create a new game where my name is executable

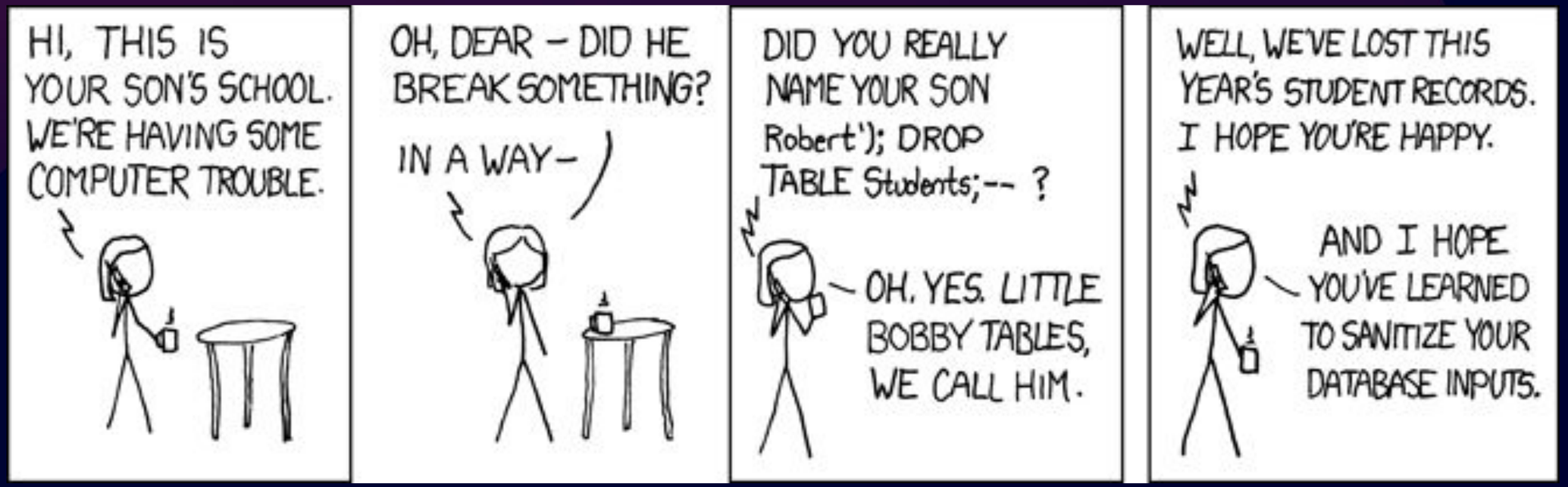
```
<script>alert("HACKED YOU")</script>
```

With WAF

- WAF catches and forbids my request
- Works? Crude weapon!!
- Don’t let it happen!



SQL Injection



SQL Injection

```
"INSERT INTO Students VALUES ('" + FNMName.Text + "',
'" + LName.Text + "')";

INSERT INTO Students VALUES ('Robert'); DROP TABLE
Students; --', 'Jones')
```

https://stackoverflow.com/questions/332365

DDOS

Distributed Denial of Service

Why Distributed?

Application-Level Threats

Broken Access Control

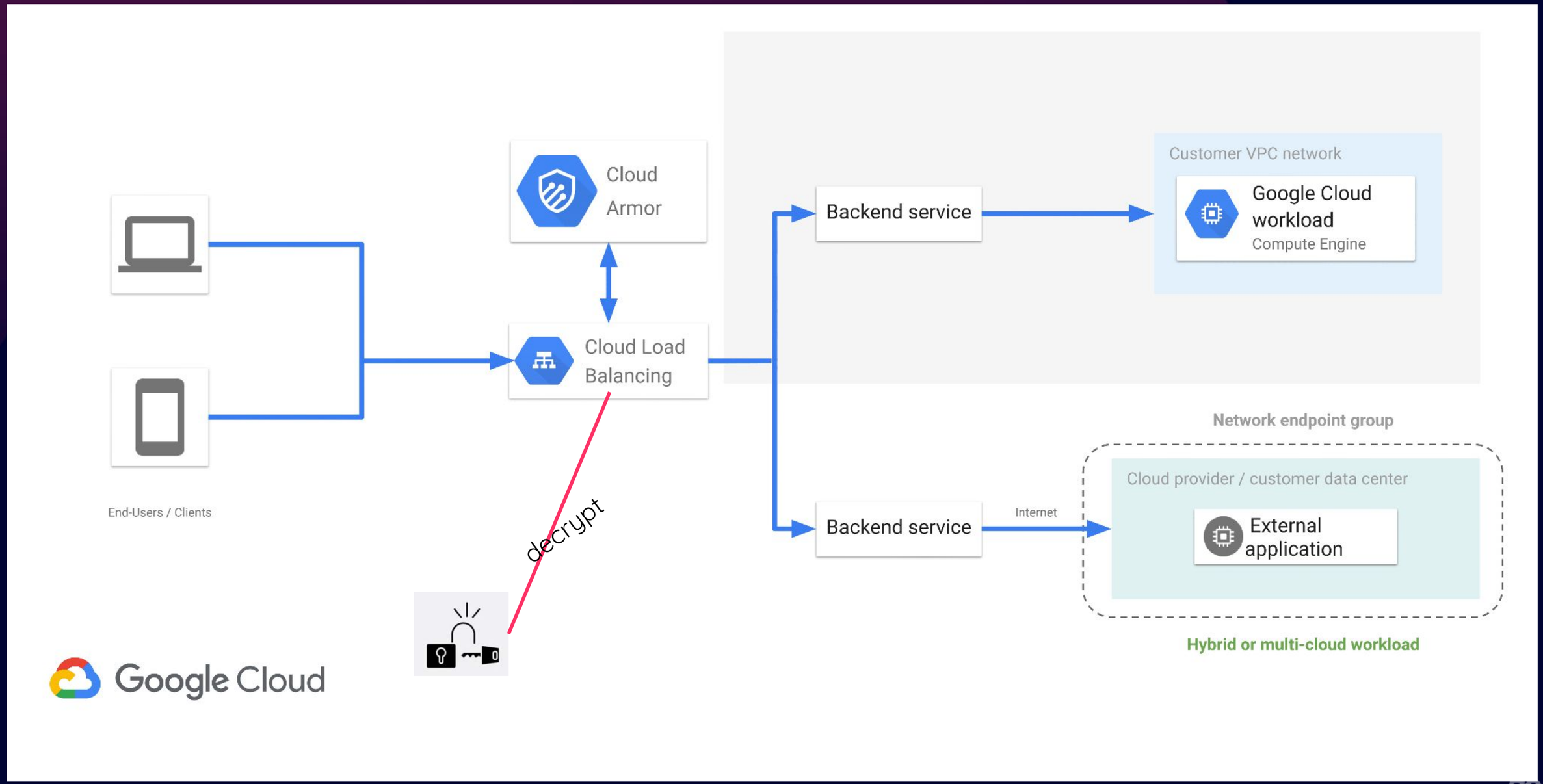
Incorrect Authorization

Authorization bypass, e.g. in search

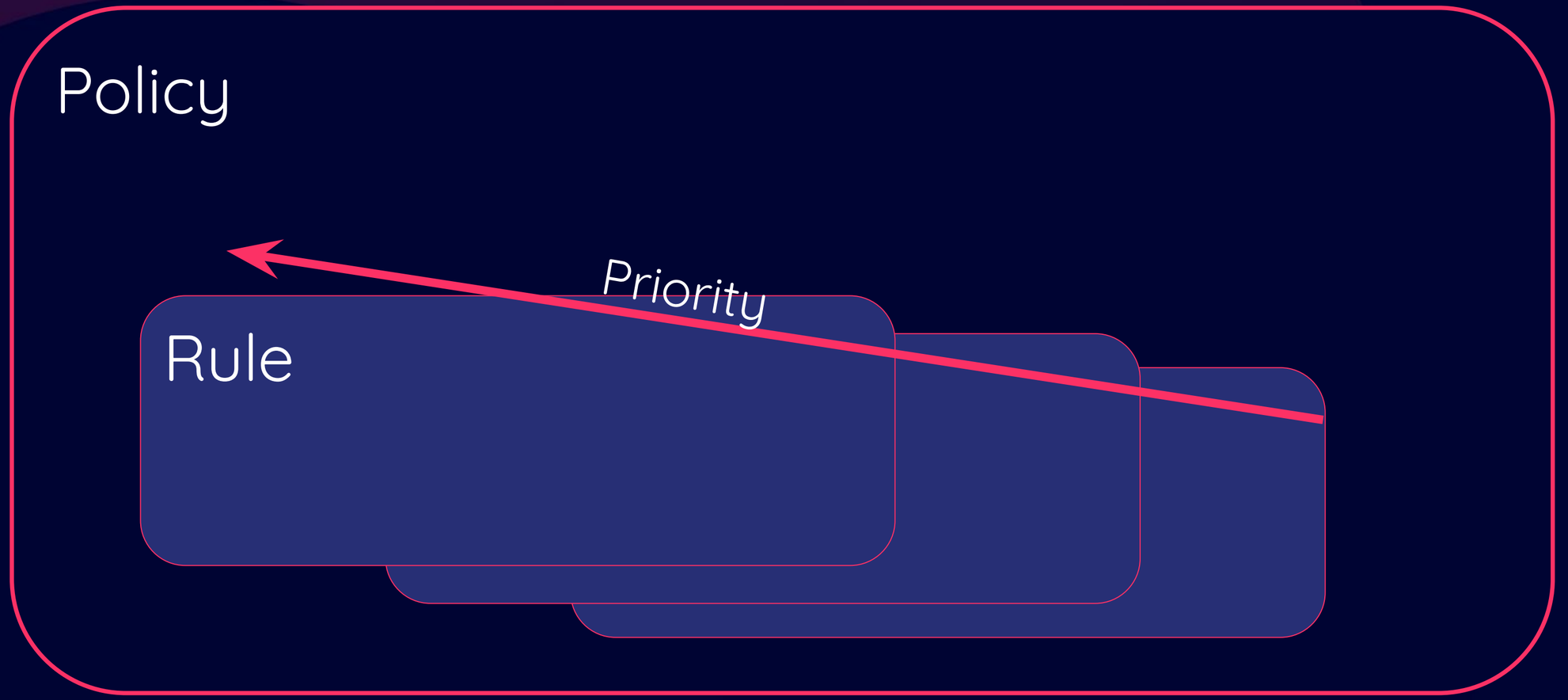
Toss in a WAF

How Cloud Armor works

Architecture



Policies and Rules



Rules

Types of Rules

- IP address blacklist/whitelist
  - (On the edge, compare to Firewall)
- Geography
- HTTP-content scan

Preconfigured Rules (Use these!)

Google Cloud Armor rule name	ModSecurity rule name
SQL injection	`sqli-v33-stable`
	`sqli-v33-canary`
Cross-site scripting	`xss-v33-stable`
	`xss-v33-canary`
Local file inclusion	`lfi-v33-stable`
	`lfi-v33-canary`

Sensitivity (Paranoia)

False positives and negatives

```
evaluatePreconfiguredWaf(
    'sqli-v33-stable',
    {'sensitivity': 2}
)
```

Standard Signatures

CRS 3.3 | CRS 3.0

Signature ID (Rule ID)	Sensitivity level	Description
owasp-crs-v030301-id942100-sqli	1	SQL Injection Attack Detected via libinjection
owasp-crs-v030301-id942140-sqli	1	SQL injection attack: Common DB Names Detected
owasp-crs-v030301-id942160-sqli	1	Detects blind SQLi tests using sleep() or benchmark()
owasp-crs-v030301-id942170-sqli	1	Detects SQL benchmark and sleep injection attempts including conditional queries
owasp-crs-v030301-id942190-sqli	1	Detects MSSQL code execution and information gathering attempts

Sample signature

```
# Regexp generated from util/regexp-assemble/regexp-942170.data using Regexp::Assemble.
# To rebuild the regexp:
#   cd util/regexp-assemble
#   ./regexp-assemble.pl regexp-942170.data
# Note that after assemble an outer bracket with an ignore case flag is added
# to the Regexp::Assemble output:
#   (?i:ASSEMBLE_OUTPUT)
#
SecRule REQUEST_COOKIES|!REQUEST_COOKIES:/__utm/|REQUEST_COOKIES_NAMES|
        ARGS_NAMES|ARGS|XML:/* "@rx
(?i:(?:select|;)\s+(?:benchmark|sleep|if)\s*?\(\s*?\(?\s*?\w+)" \
    "id:942170,\
    phase:2,\
    block,\
    capture,\
    t:none,t:urlDecodeUni,\
    msg:'Detects SQL benchmark and sleep injection attempts including conditional queries',\
    logdata:'Matched Data: %{TX.0} found within %{MATCHED_VAR_NAME}: %{MATCHED_VAR}',\
    tag:'application-multi',\
    tag:'language-multi',\
    tag:'platform-multi',\
    tag:'attack-sqli',\
….
    tag:'OWASP_AppSensor/CIE1',\
    tag:'PCI/6.5.2',\
    ver:'OWASP_CRS/3.2.0',\
    severity:'CRITICAL',\
    setvar:'tx.sql_injection_score=+%{tx.critical_anomaly_score}',\
```

Rule language

```
request.headers['user-agent'].
matches('(?i:wordpress)')
```

WAF won’t protect you!

Blocking Your Own App

WAF fights your own app

- Software Engineering Forum:
  - Requests disappear!

False positives

- If your app passes executable code on purpose.
- If it is full of text-fields and unvetted code
- Or just plain failure in imperfect rules

Job zero

A secure application

Secure your app

E.g.

- Escape all strings

```
<script>alert(1)</script>
```

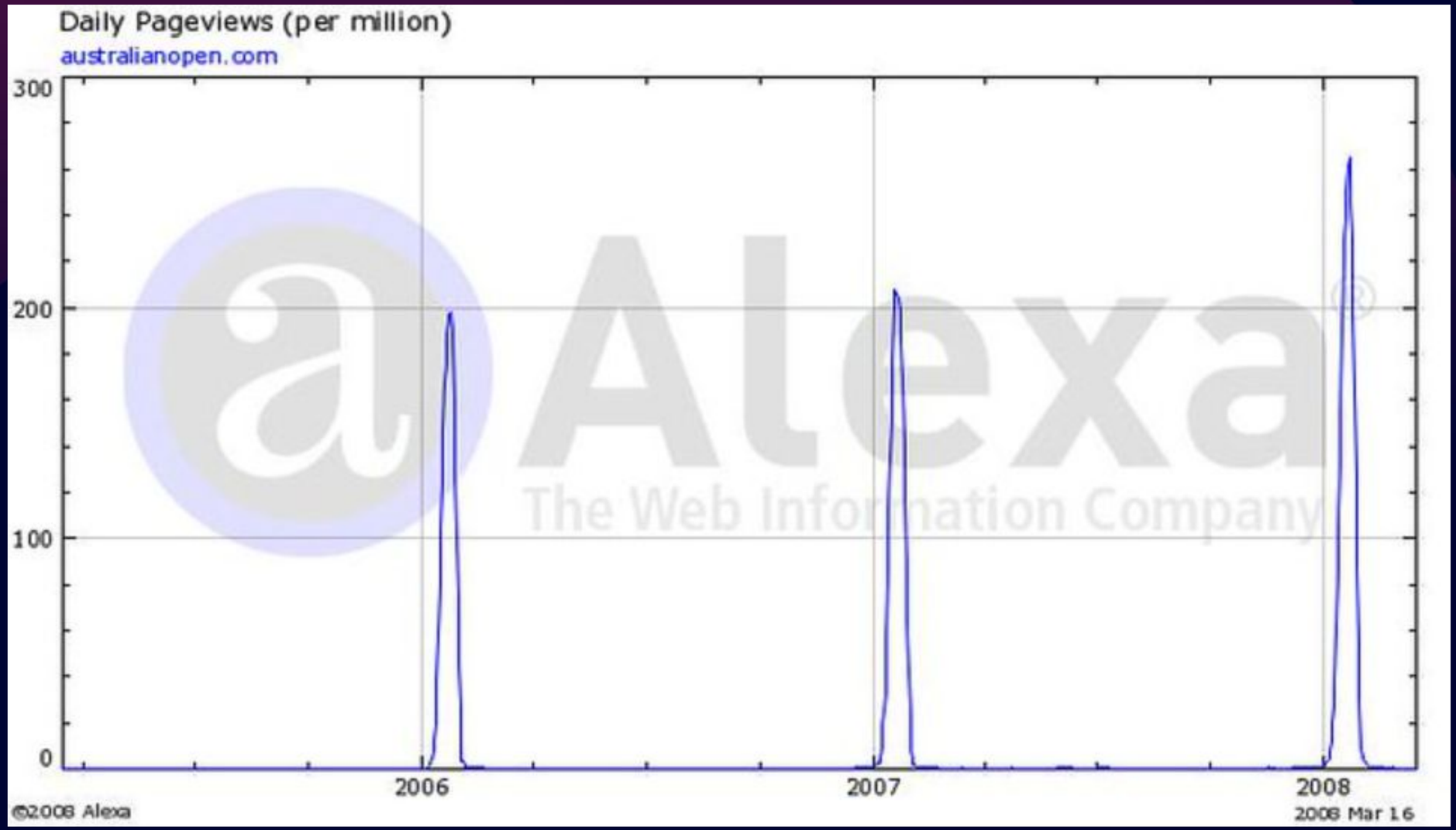
```
%3Cscript%3Ealert(1)%3C%2Fscript%3E
```

- Don't `dangerouslySetInnerHTML`
- Sanitize?

But the most important

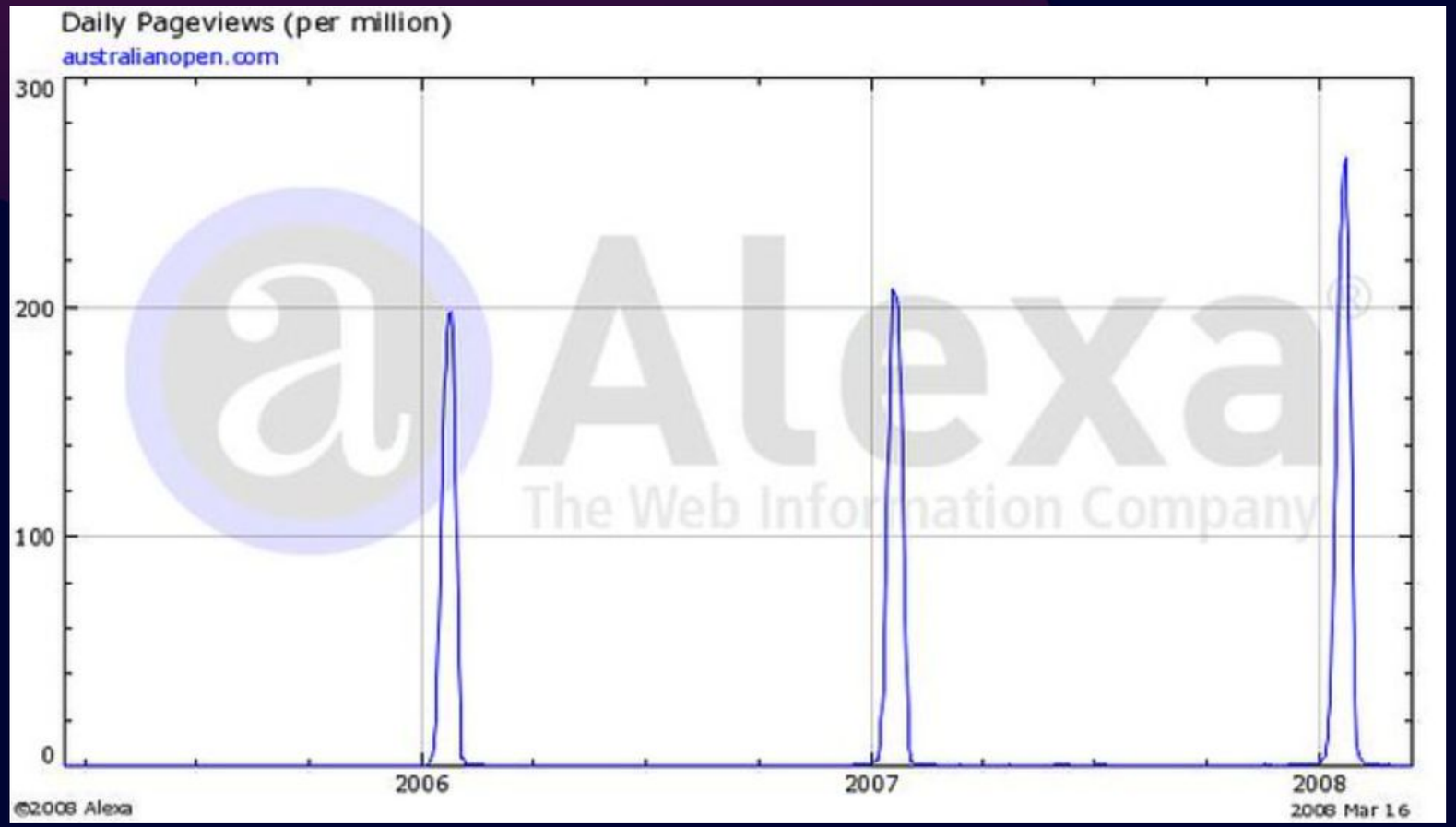
A security mindset

DDOS


Daily Pageviews (per million)
australianopen.com
300
200
100
0
2006
2007
2008
©2008 Alexa
2008 Mar 16
Alexa
The Web Information Company

DDOS False Positives

1. Plan in advance
2. Rely on advanced ML-driven features



IP Address

37.60.43.234

Also, address blocks

Geo


CHINA
AFGHANISTAN
PAKISTAN
INDIA
OMAN
LAOS
THAILAND
INDONESIA

Dry run

Preview

```
previewSecurityPolicy: {
configuredAction: "DENY"
name: "owasp-modsecurity-core-reuse-set"
…
```

Problem with Preview

Uncertainty

False Negatives

Letting the attacks through

Imperfect detection

- Regexes are limited as a language
- Regexes must be fast
- Only a few Kilobytes scanned

The worst: Broken Access Control

- Hundreds of pages: Which are open to
  - Unauthenticated, all authenticated, or wrong users
  - Read? Write?

WAF is helpless

Attackers shift

- Change IP addresses
- Change countries
- Change attacks
- Tiny string changes

Attackers are smart!

- Always scheming...
- WAFs have predefined rules
- Though ML can be flexible

Flexibility?

- Configure policies
- Configure rules
  - Exclude rule
  - Exclude field
- Choose sensitivity level
- Create rules

Flexibility not enough

- The experts already worked on the tough balance.
- You **don't** have a very special case that needs your own rules or config.
- And if you do, that requires some targeted work.

WAF adds risk!

Man-in-the-Middle

- WAF decrypts your HTTPS
- What if it has a bug?

Risk: Complacency

- Temporary becomes permanent
- Skills are not learned
- Chicken-wire fence

Risk to Performance

The WAF decrypts and analyzes every request

Pricing

- Basic: Pay for rules, policies, requests
- “Enterprise”: Monthly fee, also for resources, data

Expensive?

At long last...

What is it good for?

External Requirement

- Government
- Customer
- Partner
- Standards body

Third-Party Apps

- You have no access to code.
- But you still risk complacency.

Central Supervision

- Security Team handles new CVEs, e.g. log4J
- Temporary until product teams fix it
- DENY/ALLOW/LOG for monitoring tool as well as protection-tuning

When you know for sure what you want to block

The one go-to feature

DDOS

Consider advanced services

- Human team
- Attack visibility
- Third-party named IP address lists
- Threat Intelligence
- Adaptive Protection

If you're going to do it, do it now

No fire drills

Prefer your Cloud's WAF

- Google Cloud Armor or AWS Shield
- The HTTPS is probably being decrypted anyway.
- Pay-as-you-go, especially for your ***long*** ramp-up period

Minuses of a WAF

- Complacency
  - Resources and skills
- False positives
- False negatives
- Added risk and slowness

Plusses of a WAF

- Easier than DIY
- No need to change code
- Centrally-controllable
- Features that you can’t provide
  - Adaptive protection
  - DDOS

Conclusion

- Security is job zero
- Security is in your app
- You can't hand off responsibility
- Use WAF where relevant

We're hiring!

joshua@doit.com

Slides

https://bit.ly/waf-dont

doit