F5 EMEA Webinar, November 2017

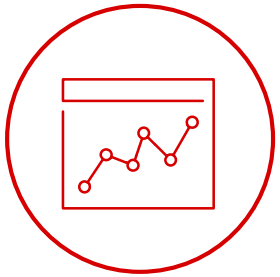
Defend Your Web Applications Against the OWASP Top 10 Security Risks

PRESENTED BY:

Speaker Name, Job Title

WE MAKE APPS GO FASTER. SMARTER. SAFER.

Application Security Is Business Continuity



Maintain and grow revenue

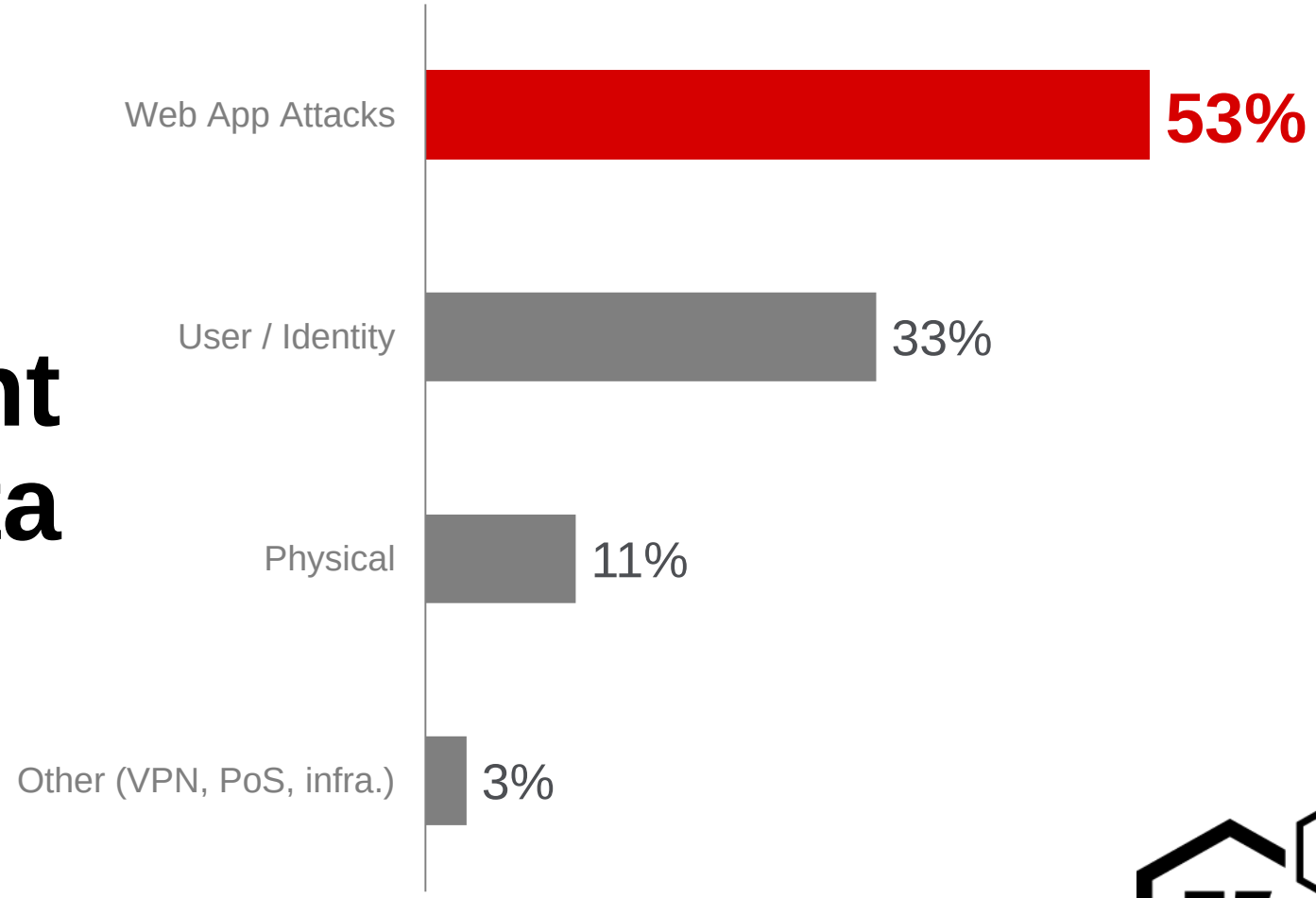
Identify industry threats

Protect assets and customers

GOAL

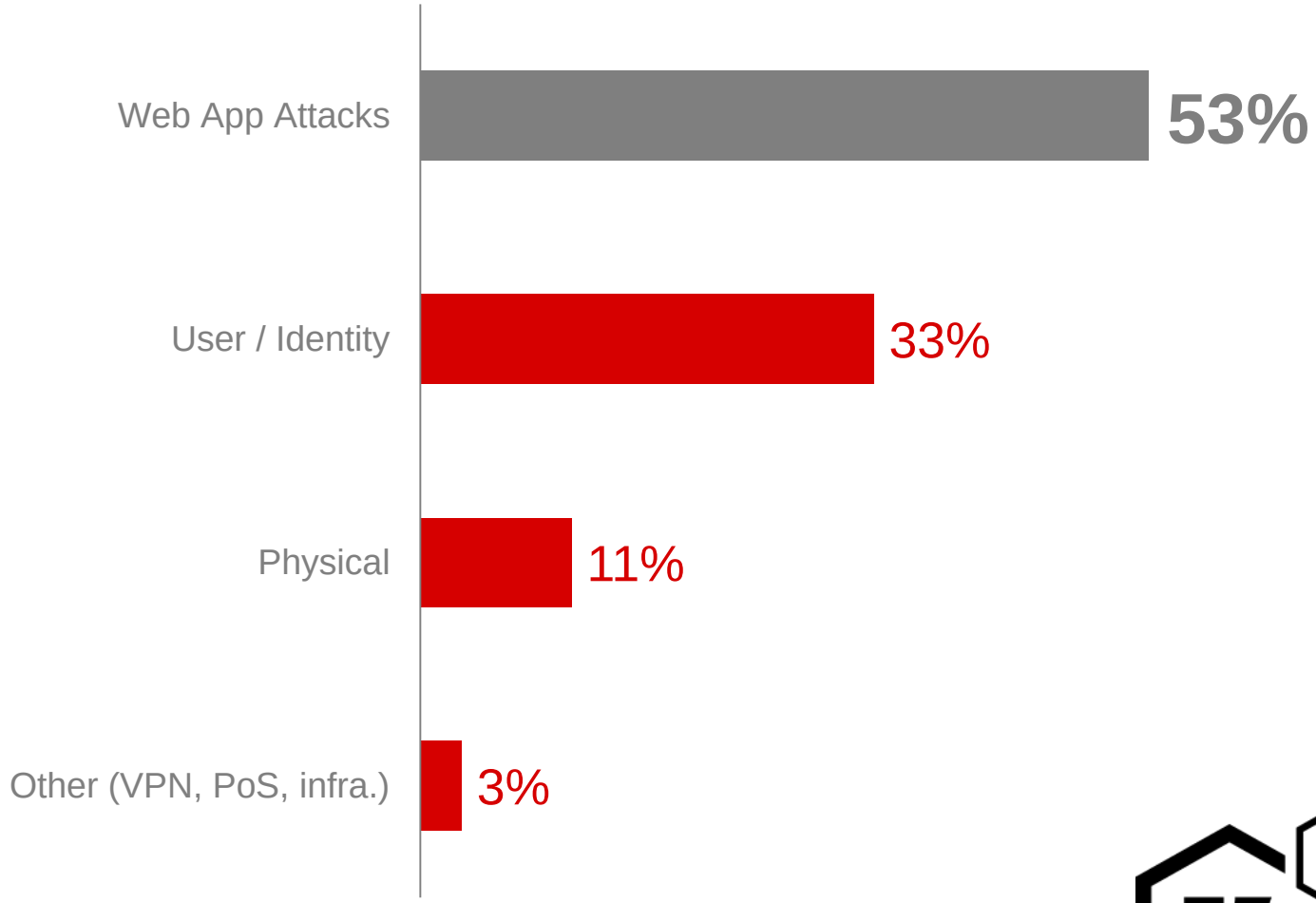
Defense in Depth

Web app attacks are the #1 single source entry point of successful data breaches



https://f5.com/labs

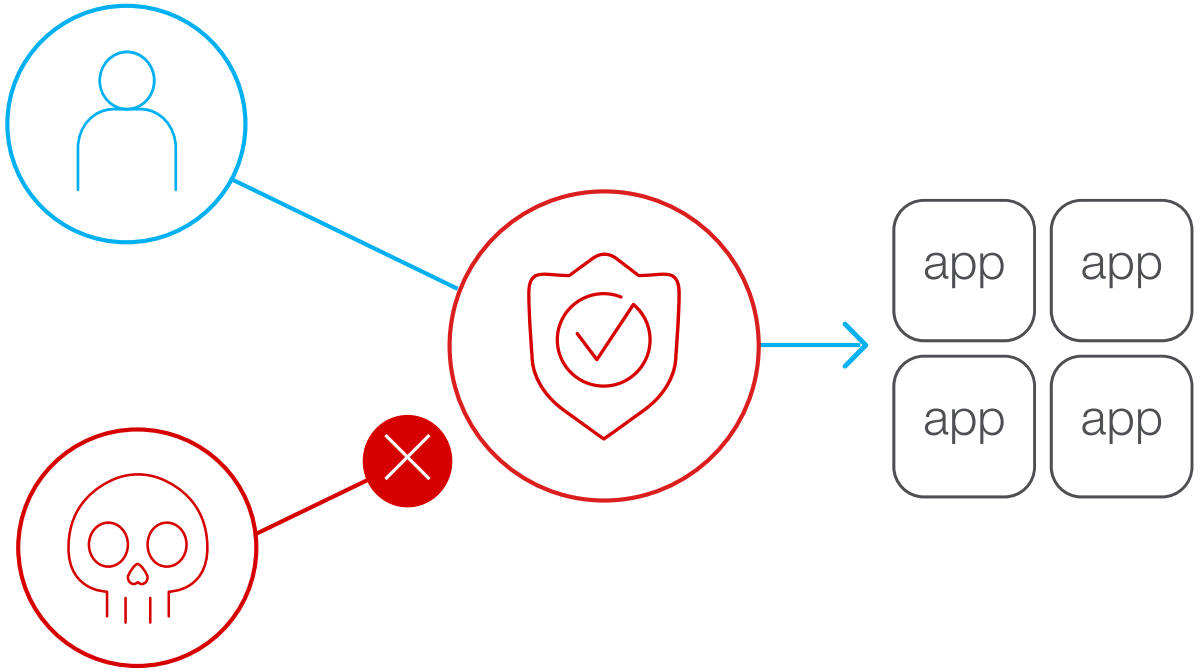
But we still have a lot of other exposure



https://f5.com/labs

WAF can stop attacks before they reach your apps

- Secure
- Expedient
- Effective

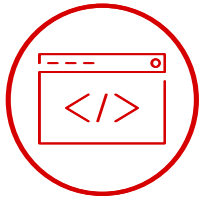


WAF Protections

Traditional WAF:

OWASP Top 10

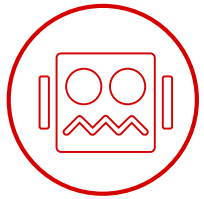
SSL/TLS Inspection



Scripting

Advanced WAF:

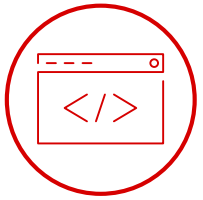
OWASP Top 10



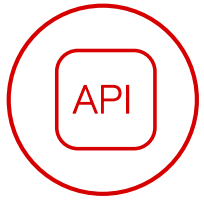
Malicious Bots

SSL/TLS Inspection

Credential Attacks



Scripting



API Attacks

The New OWASP Top 10

Open Web Application Security Project

What is the OWASP Top 10?

A broad consensus on the most critical web application security flaws

Most are very well known attack vectors that persist

Coverage is mandated by some regulatory requirements such as PCI DSS

What do all of these attack vectors have in common?

These are difficult and complex problems – despite being well known, remain pervasive

+

Building coverage into apps again and again is expensive – if you can find the expertise

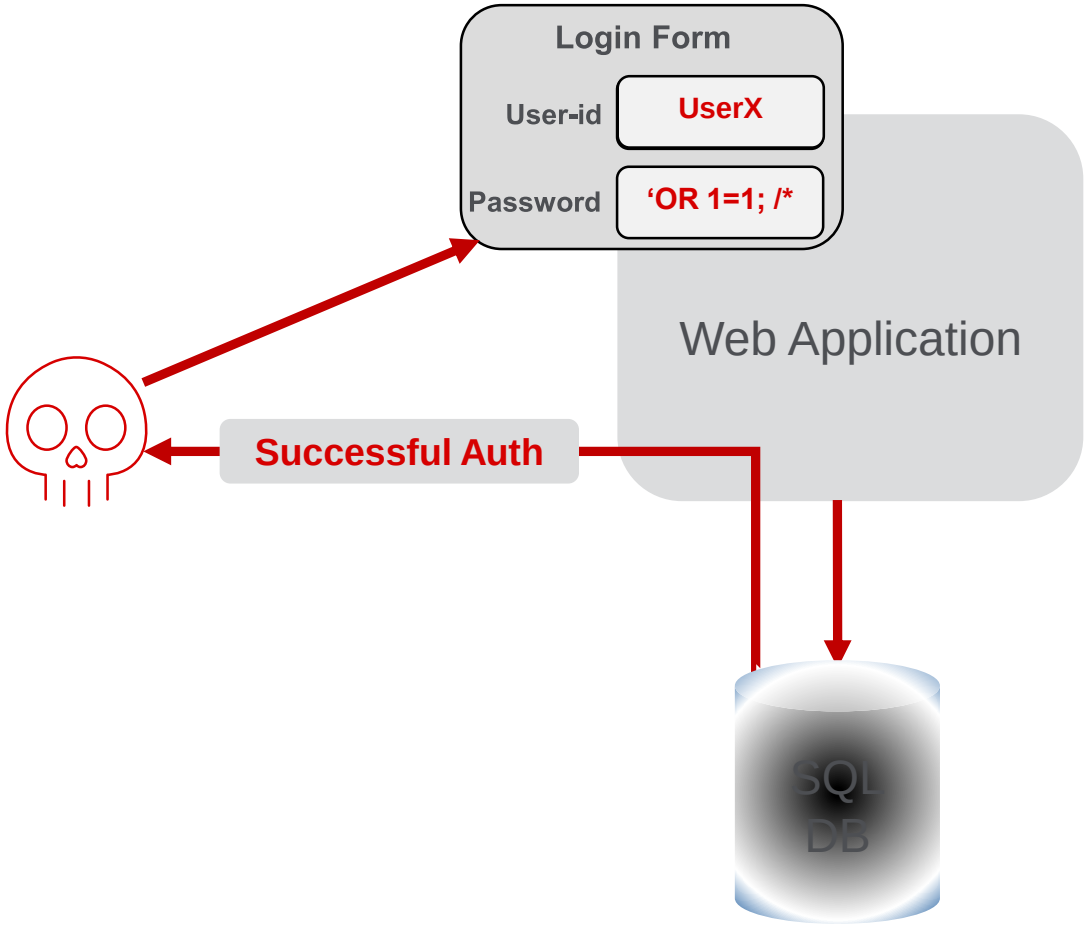
=

Common vulnerabilities remain difficult to defend against in any case

A1:2017 – Injection

RISK:

Insufficiently sanitised input data can lead to unauthorised command execution, data exfiltration, or data deletion



A1:2017 – Injection

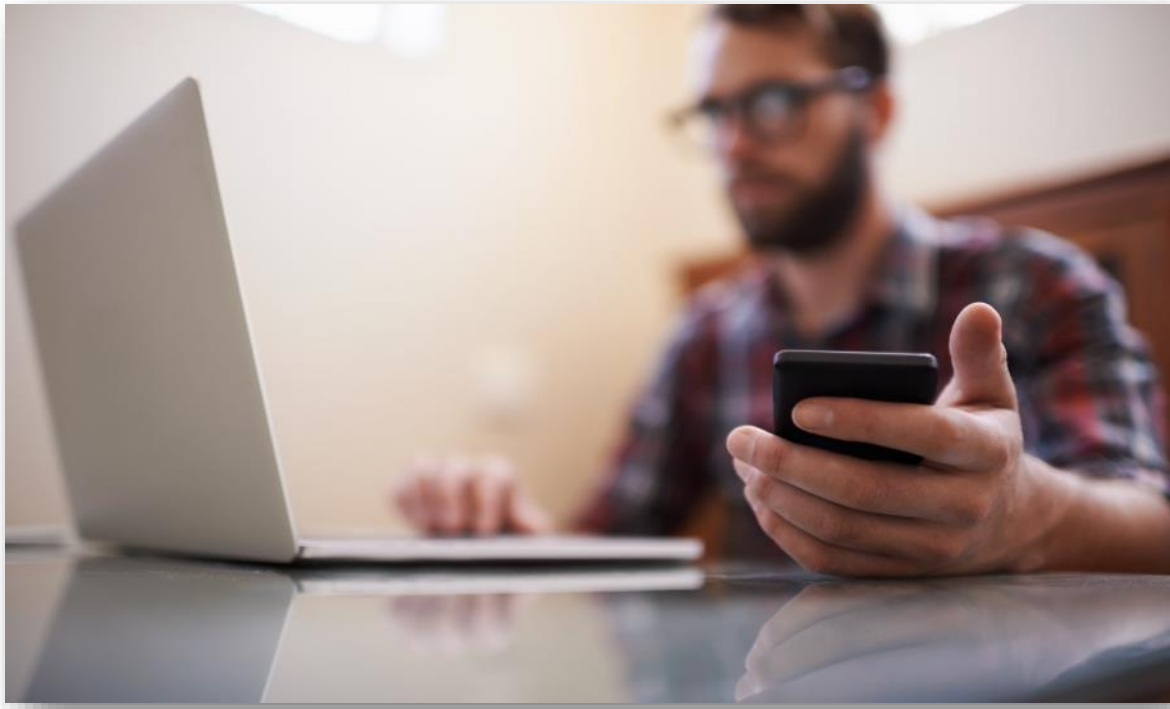
HOW TO MITIGATE

- Decrypt and inspect
- Block malicious payloads
- Enforce parameter values

A2:2017 – Broken Authentication

RISK:

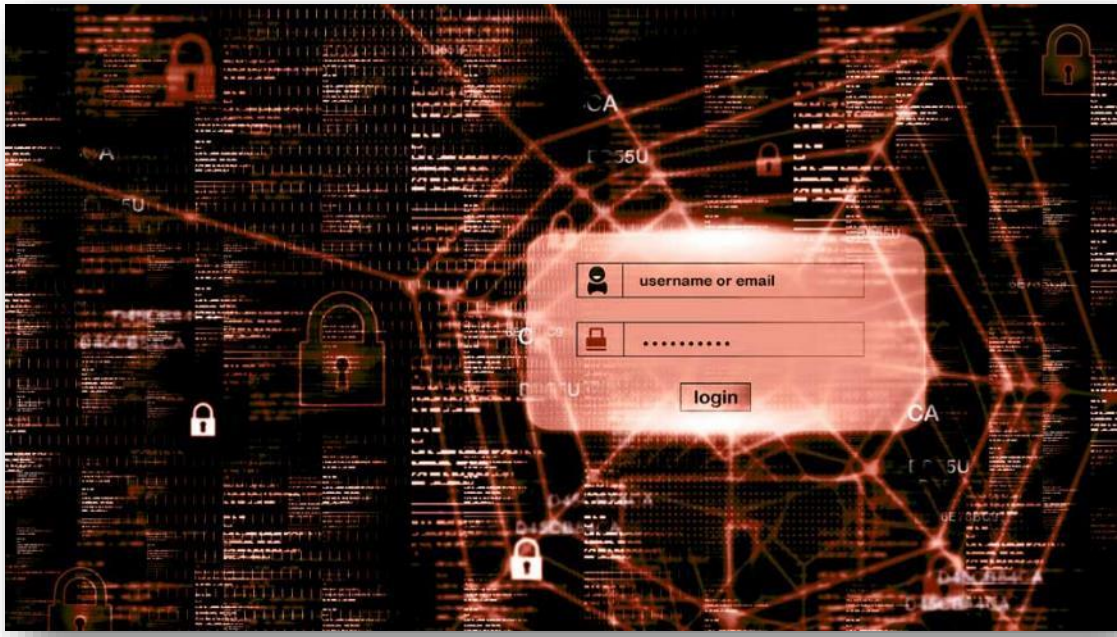
Compromised credentials, keys, session tokens, or other implementation flaws can allow attackers to assume other users' identities



A2:2017 – Broken Authentication

HOW TO MITIGATE

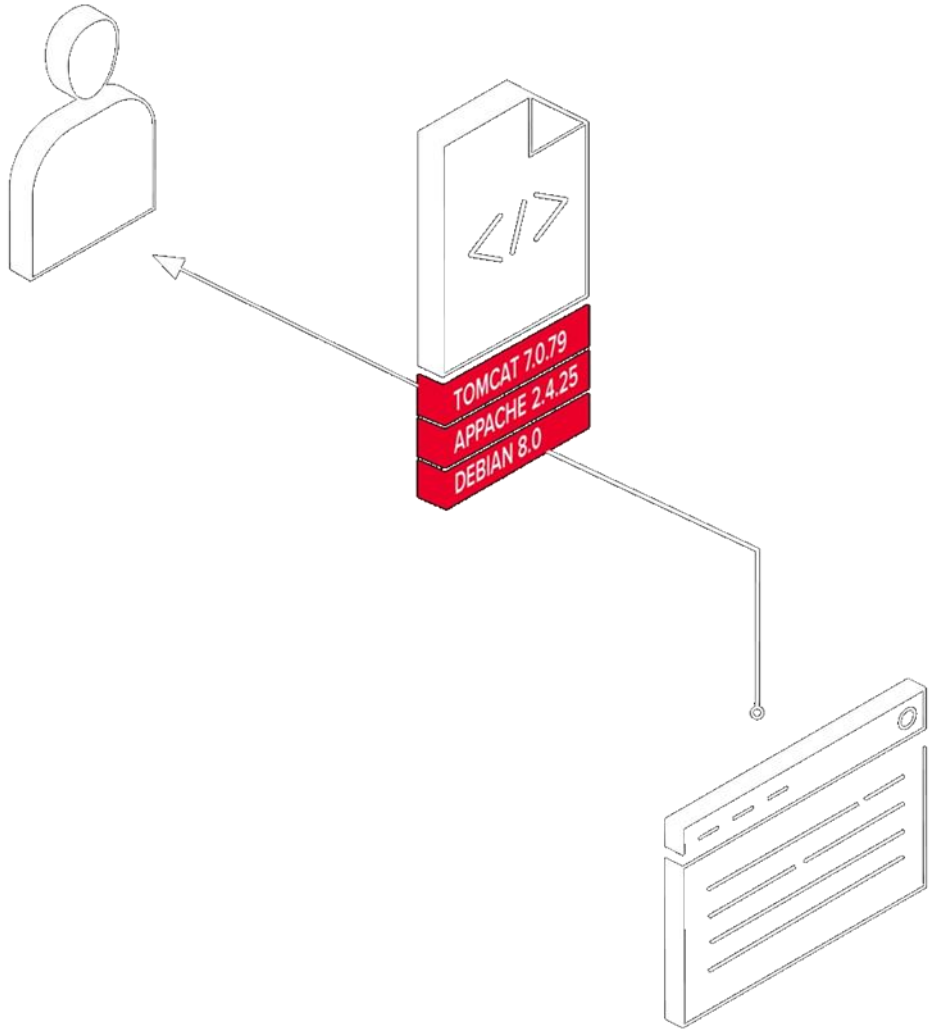
- Enable risk-based workflows
- Enforce session integrity
- Enforce site-wide TLS
- Monitor requests
- Leverage signed cookies



A3:2017 – Sensitive Data Exposure

RISK:

Unintentionally arming attackers with information critical to the execution of a breach, or just leaking the data they are after



A3:2017 – Sensitive Data Exposure

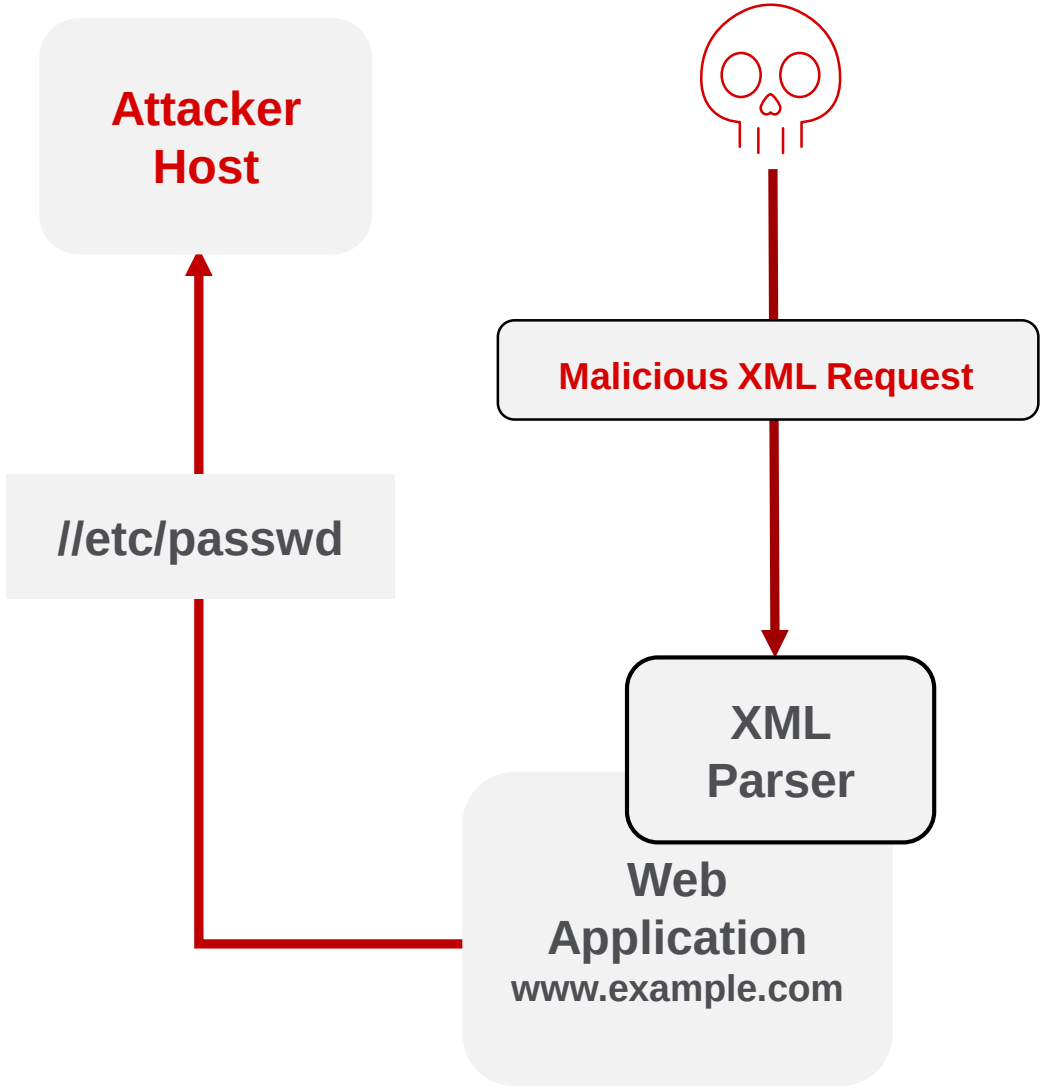
HOW TO MITIGATE

- Sanitise output
- Hide back-end software versions
- Manage login workflows
- Enforce usage of SSL
- Set the "secure" flag on all sensitive cookies

A4:2017 – XML External Entities (XXE)

RISK:

Apps that accept XML input can inadvertently allow external references and commands that cause XML processors to divulge data, execute code, and initiate outbound connections



A4:2017 – XML External Entities (XXE)

HOW TO MITIGATE

- Enforce HTTP standards
- Validate JSON, XML, and SOAP requests
- Choose APIs to expose
- Check for XML DoS and API attack signatures
- Limit outbound requests

A5:2017 – Broken Access Control

RISK:

Access to a sensitive object is insufficiently enforced resulting in it being retrievable by users whose credentials would not otherwise allow such access

A5:2017 – Broken Access Control

HOW TO MITIGATE

- Ensure that site content is only accessed by users with proper credentials
- Enforce flow control

A6:2017 – Security Misconfiguration

RISK:

Errors in documentation, lack of oversight, or mistakes made by administrators result in access policies or application protection systems not functioning

A6:2017 – Security Misconfiguration

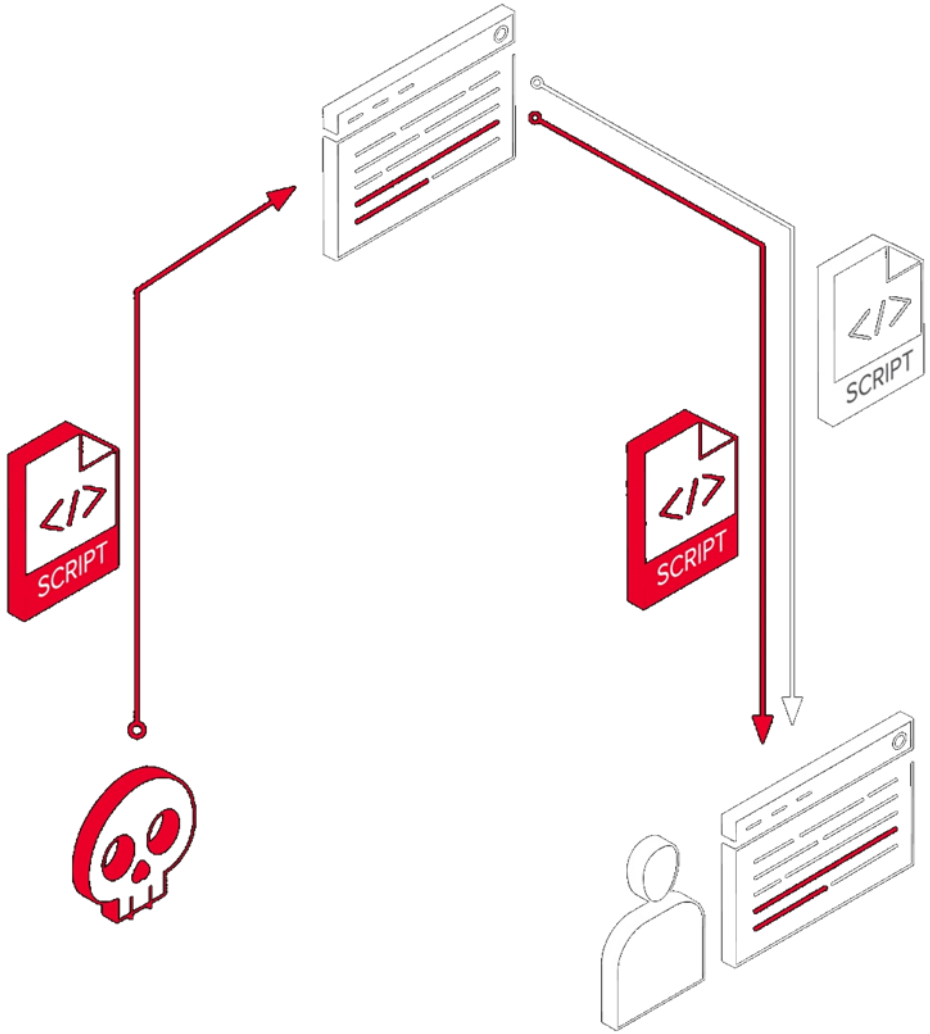
HOW TO MITIGATE

- Apply default deny controls
- Act as a central point of control
- Provide RFC enforcement
- Use whitelisting
- Enforce authenticated access

A7:2017 – Cross-Site Scripting (XSS)

RISK:

Attackers coerce web apps to store and serve, or reflect malicious code back to a victim to be executed within a site they're visiting



A7:2017 – Cross-Site Scripting (XSS)

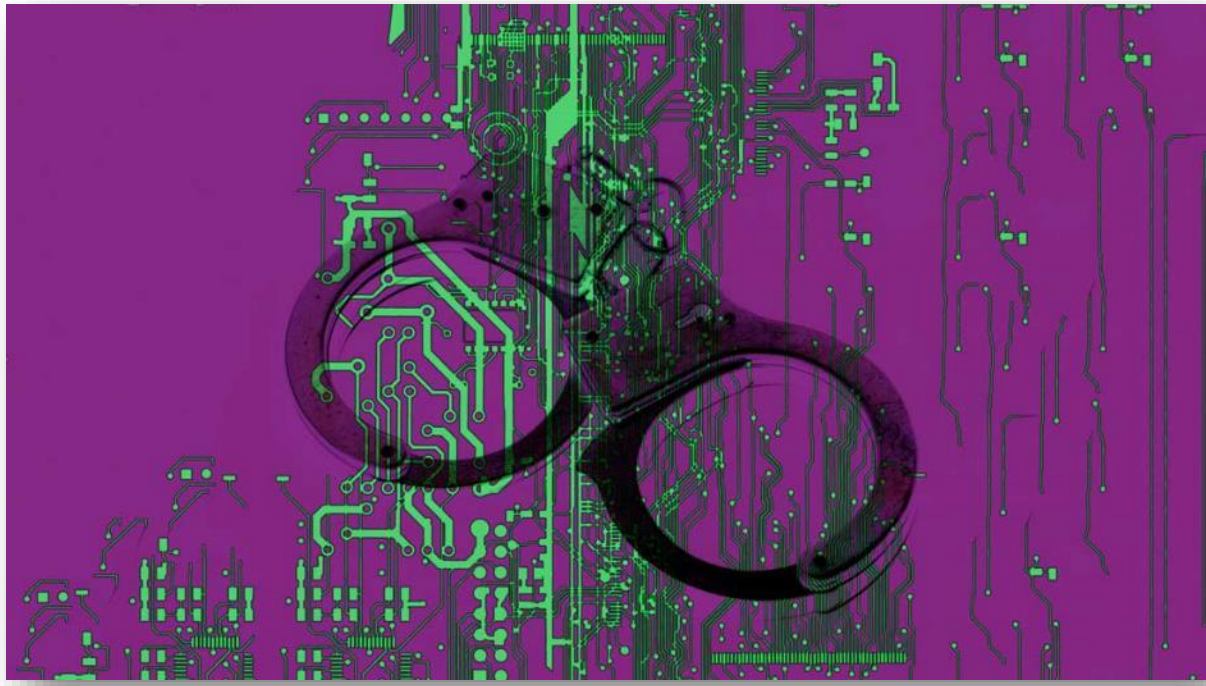
HOW TO MITIGATE

- Detect and filter
- Enforce accurate usage of URI metacharacters
- Enforce pre-defined parameter values

A8:2017 – Insecure Deserialisation

RISK:

Insufficient parsing of serialised data can result in unauthorised modification of tokens, bypass of access controls, remote code execution, and DoS



A8:2017 – Insecure Deserialisation

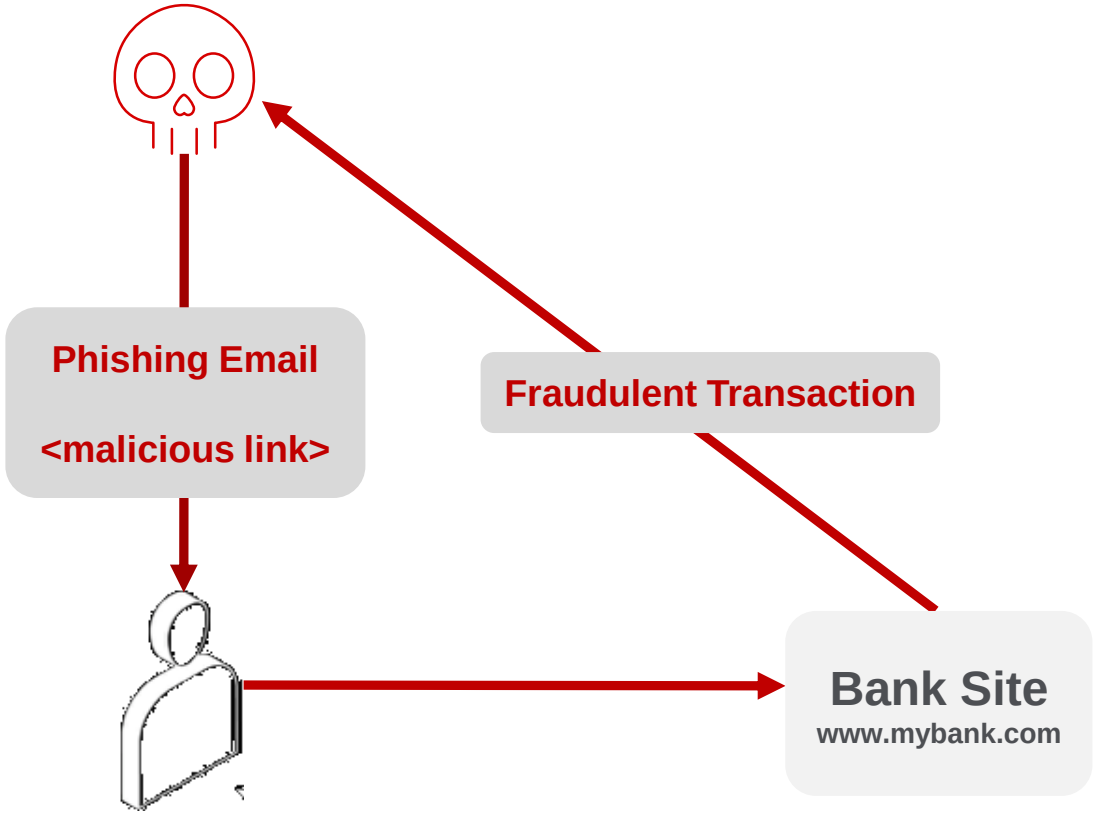
HOW TO MITIGATE

- Enforce HTTP controls
- Validate JSON, XML, and SOAP requests
- Check JSON parser output

A8: Cross-Site Request Forgery (from previous Top 10)

RISK:

A victim clicks on a link that was crafted by an attacker which piggybacks their credentials to initiate activity in a protected or sensitive application



A8: Cross-Site Request Forgery
(from previous Top 10)

HOW TO MITIGATE

- Add a random nonce to every URL that cannot be guessed by an attacker
- Limit application entry points

A9:2017 – Using Components with Known Vulnerabilities

RISK:

New vulnerabilities are discovered all of the time, you may need a particular version of something explicitly or there is risk in upgrading a dependency

A9:2017 – Using Components with Known Vulnerabilities

HOW TO MITIGATE

- Automatically **detect server software stacks**
- Analyse client request/ response against policy
- Provide **advanced programmability**
- Support “virtual patching”

A10:2017 – Insufficient Logging and Monitoring

RISK:

Log data and access information that is not collected or analysed will not help detect a breach, attackers rely on complacency and blind spots

A10:2017 – Insufficient Logging and Monitoring

HOW TO MITIGATE

- Implement a solution to protect web apps
- Enable logging at all layers
- Integrate with 3rd-party systems
- Support HTTP fluency
- Provide threat visibility

What do all of these items have in common?

These are difficult and complex problems – despite being well known, remain pervasive

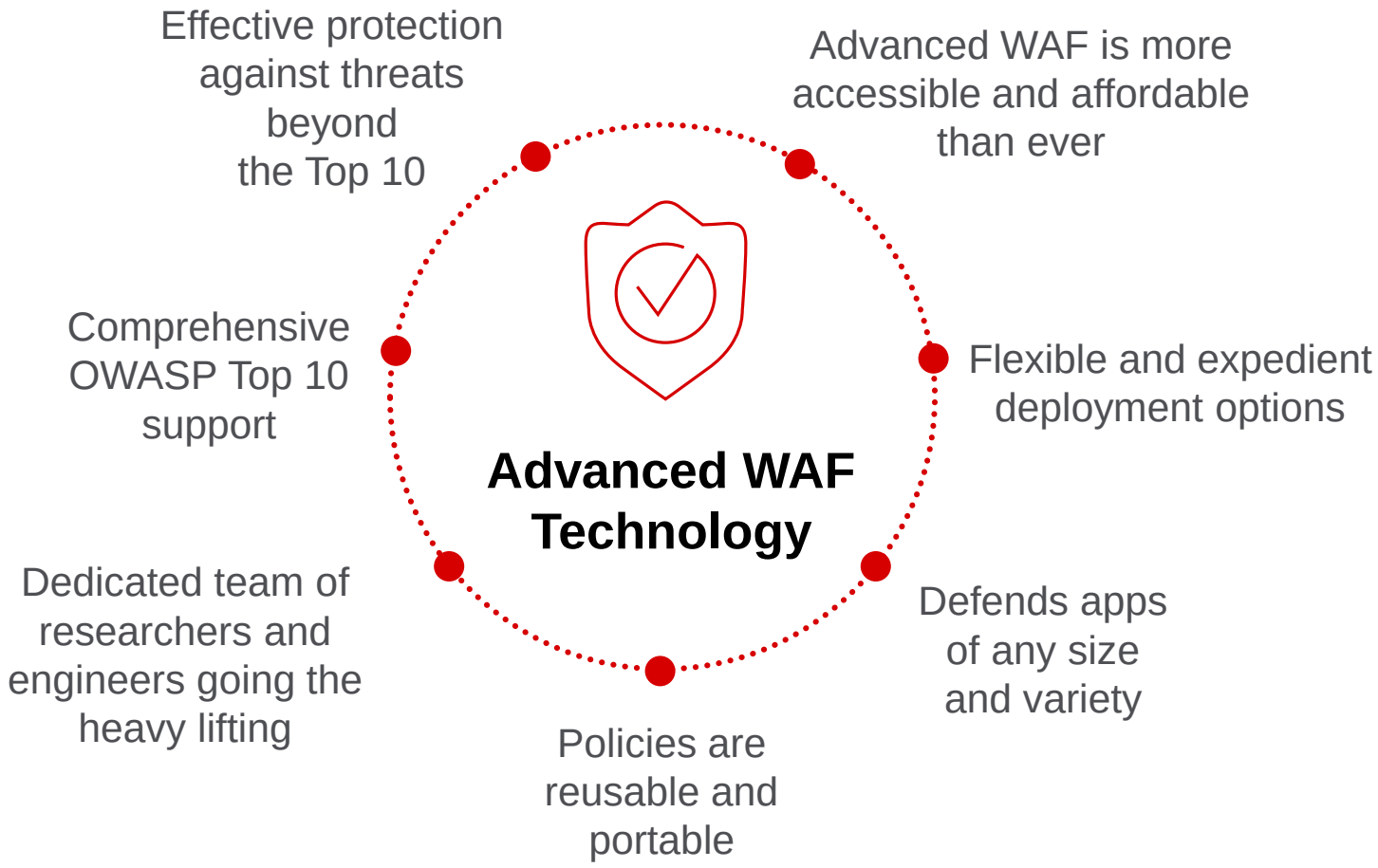
+

Building coverage into apps again and again is expensive – if you can find the expertise

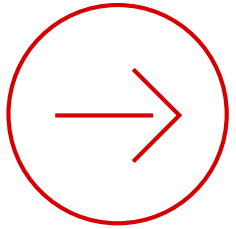
=

Common vulnerabilities remain difficult to defend against in any case

Here’s the good news



Beyond the Top 10



The OWASP Top 10 is just one piece of the app security puzzle

- Applications and vulnerabilities constantly change
- Good security tools can help reduce costs and give better business intel
- Go beyond vulnerabilities, and pursue a proactive security posture
- Push to make security part of the organisational culture where you work

Security programs are journeys of evolution

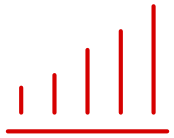
Ongoing diligence and constant refinement

Key Takeaways

These are complex concepts

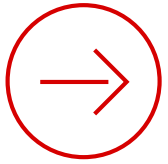
Difficult to defend against

Costly



Security programs are journeys of evolution

WAF is accessible and affordable



Protect against more than just the OWASP Top 10

WE MAKE APPS

FASTER. SMARTER. SAFER.