04.01.2021

Web Application Security Architectures

UNITED SECURITY PROVIDERS

Christoph Schulthess

Gastvortrag FHNW

Who’s speaking

Christoph Schulthess
Manager Application Security

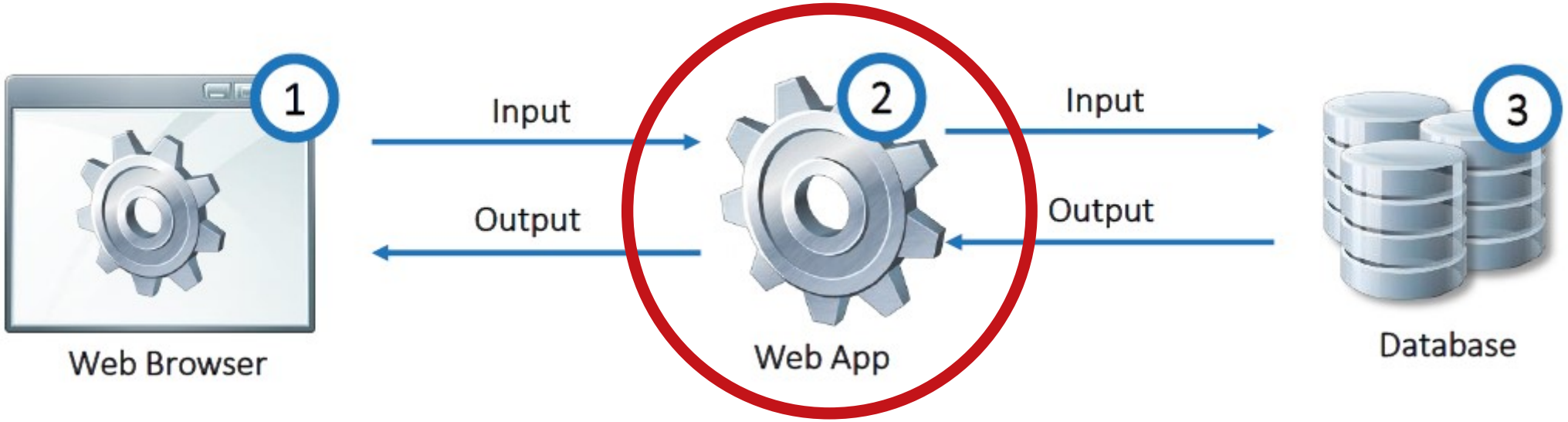
United Security Providers AG
eMail: christoph.schulthess@u-s-p.ch
Web: www.united-security-providers.ch

Agenda

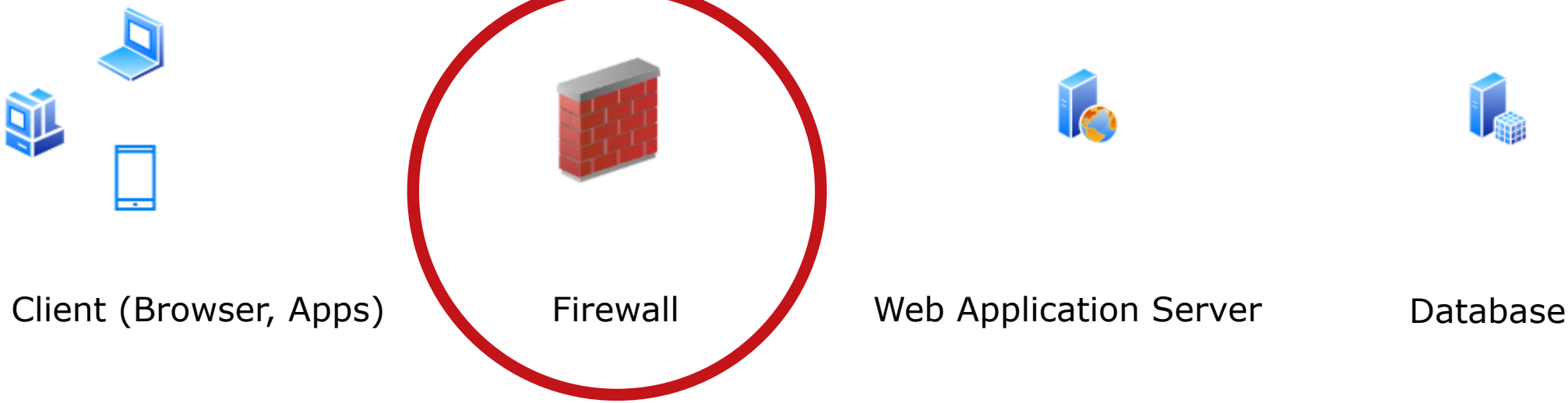
1. **What**’s a WAF
2. **Why** is this relevant and important
3. **How** can you adapt

1. **What**’s a WAF

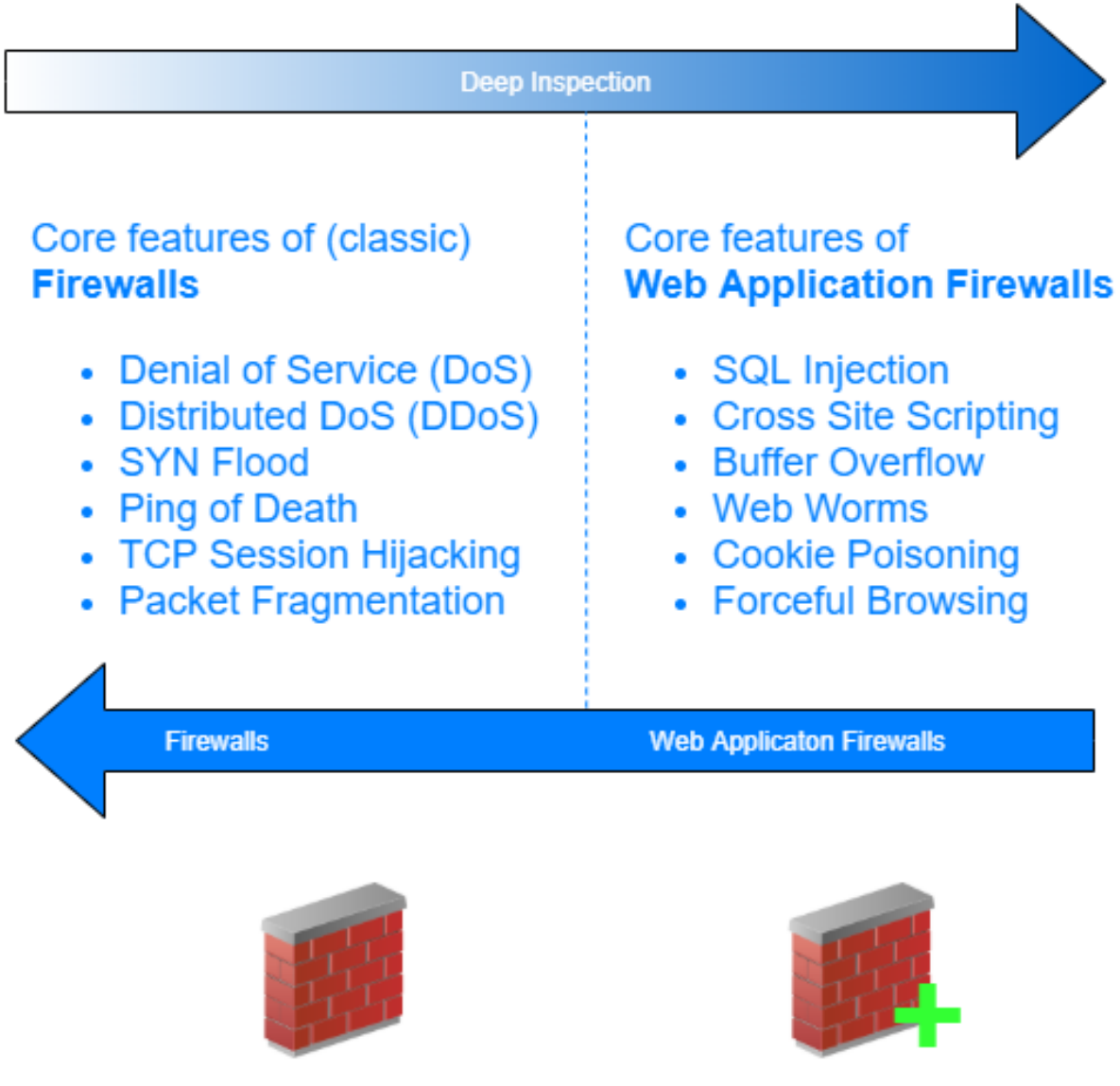
3 tier architecture



(Web Application) Firewall..?



(L3/L4 Firewall vs. WAF)



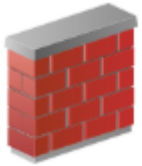
OWASP Top10 Web Application Security Risks

1. Injection
2. Broken Authentication
3. Sensitive Data Exposure
4. XML External Entities (XXE)
5. Broken Access Contro
6. Security Misconfiguration
7. Cross-Site Scripting (XSS)
8. Insecure Deserialization
9. Using Components with Known Vulnerabilities
10. Insufficient Logging & Monitoring

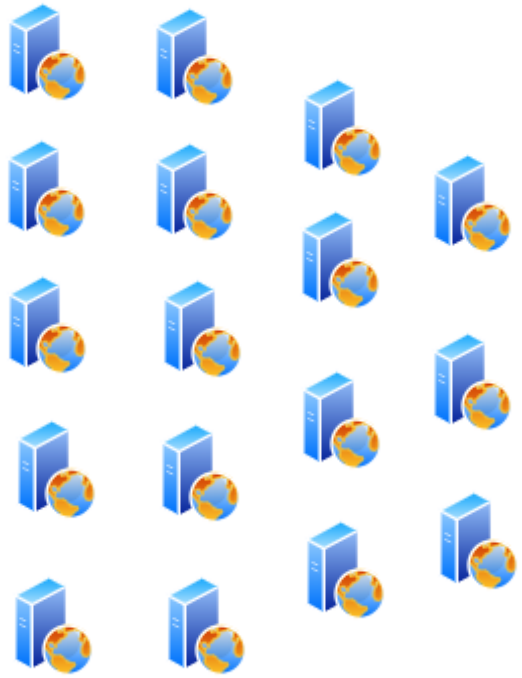
2. Why is this relevant and important

Architecture View

Client (Browser, Apps)



Firewall



√ XSS Protection

Web Application Server

Database

Web Application Firewall

Description according to OWASP (Open Web Application Security Project):

*"A web application firewall (WAF) is an **application firewall** for **HTTP** applications. It applies a set of **rules** to an HTTP conversation. Generally, these rules cover common attacks such as cross-site scripting (XSS) and SQL injection.*

*While proxies generally protect clients, **WAFs protect servers**. A WAF is deployed to protect a specific web application or set of web applications. A WAF can be considered a **reverse proxy**.*

*WAFs may come in the form of an appliance, server plugin, or filter, and may be **customized** to an application. The effort to perform this customization can be significant and needs to be maintained as the application is modified"*

(Source: https://www.owasp.org/index.php/Web_Application_Firewall)

➔ It's **not** a network firewall on layer 3 or 4 but a **gateway**.

Web Application Firewall

Description according to OWASP (Open Web Application Security Project):

*"A web application firewall (WAF) is an **application firewall** for **HTTP** applications. It applies a set of **rules** to an HTTP conversation. Generally, these rules cover common attacks such as cross-site scripting (XSS) and SQL injection.*

*While proxies generally protect clients, **WAFs protect servers**. A WAF is deployed to protect a specific web application or set of web applications. A WAF can be considered a **reverse proxy**.*

*WAFs may come in the form of an appliance, server plugin, or filter, and may be **customized** to an application. The effort to perform this customization can be significant and needs to be maintained as the application is modified"*

(Source: https://www.owasp.org/index.php/Web_Application_Firewall)

➔ It's **not** a network firewall on layer 3 or 4 but a **gateway**.

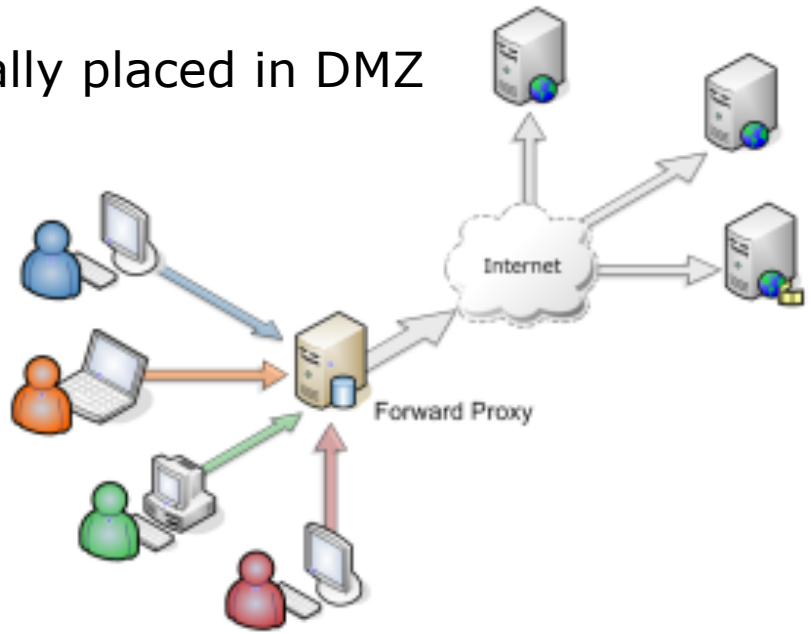
Reverse Proxy

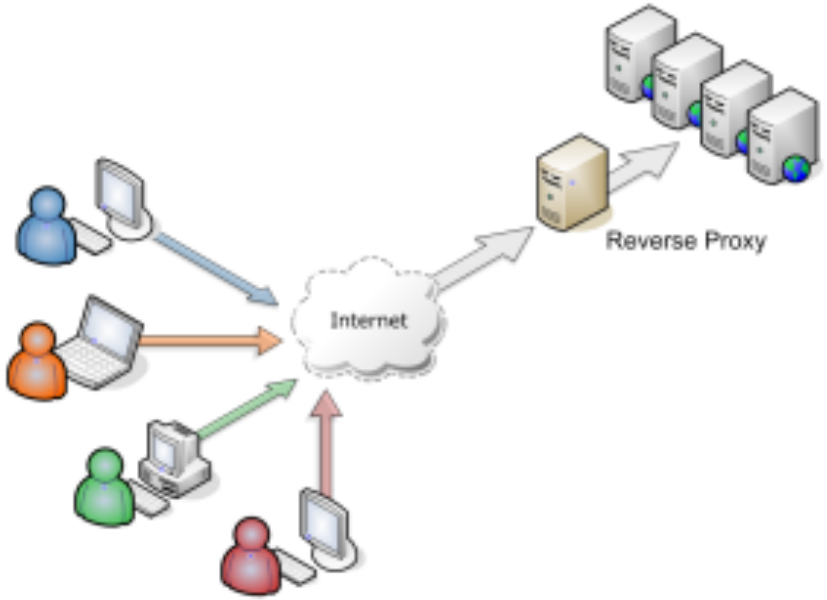
- **Forward proxy** acts on behalf of the client
- **Reverse proxy** acts on behalf of the server

Typical functions of a reverse proxy

- SSL/TLS Termination/Offloading
- Caching/Acceleration
- Load distribution

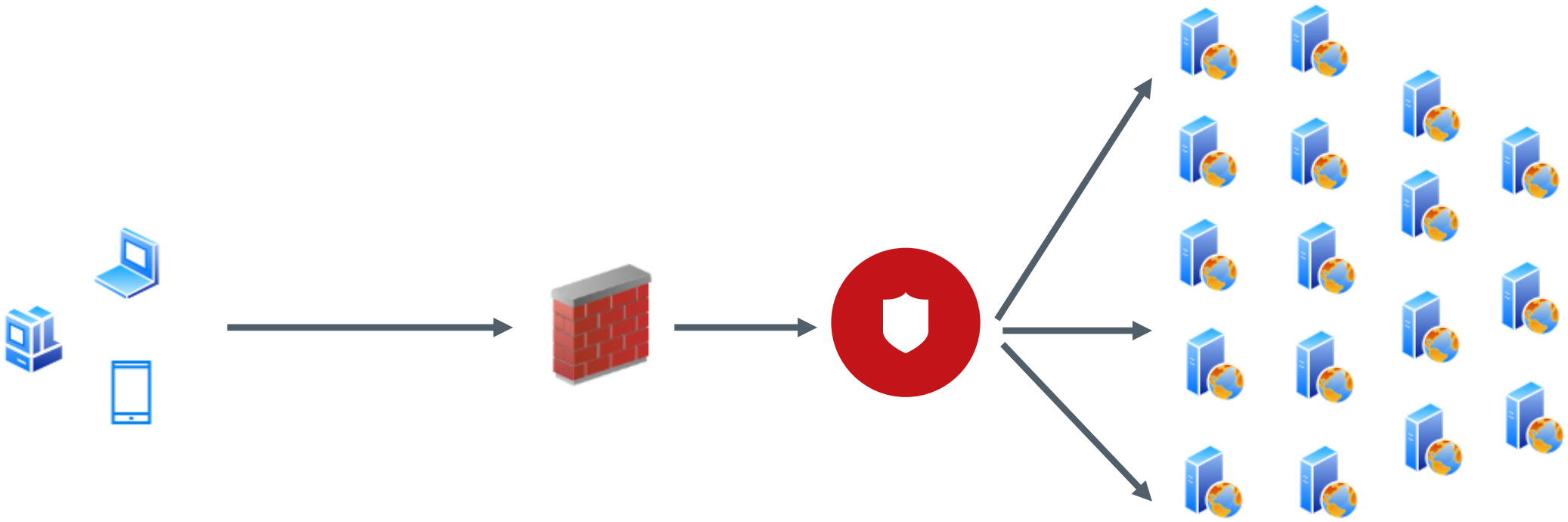
Usually placed in DMZ





(Source: https://i.stack.imgur.com/0qpxZ.png)

Architecture View



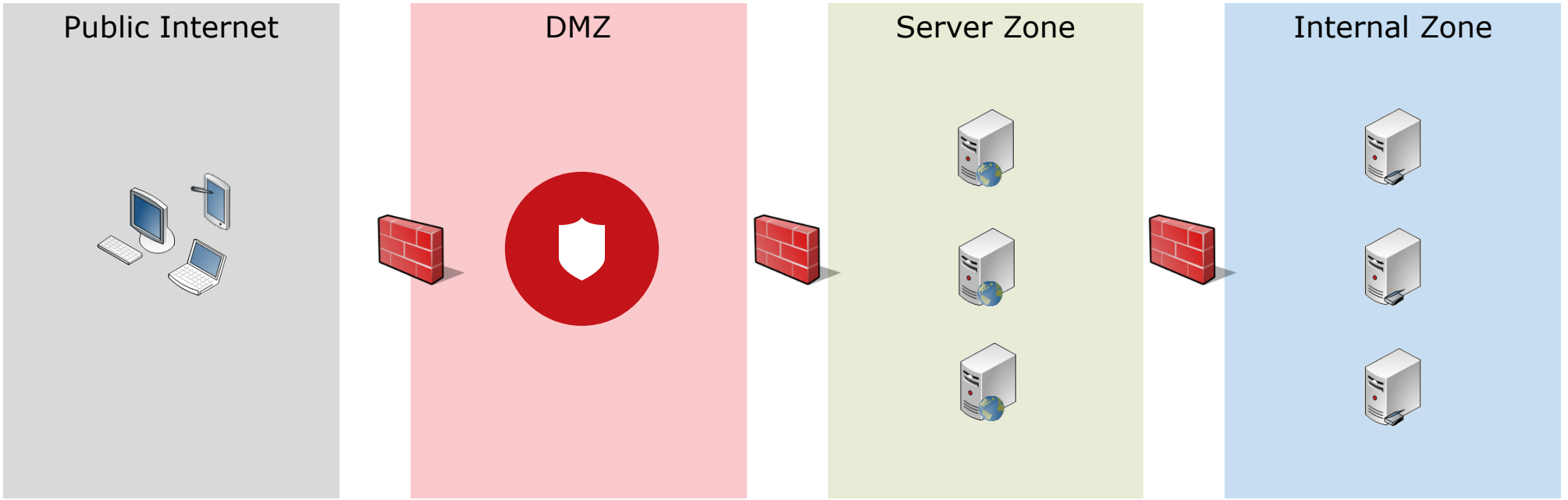
Client (Browser, Apps)

Firewall

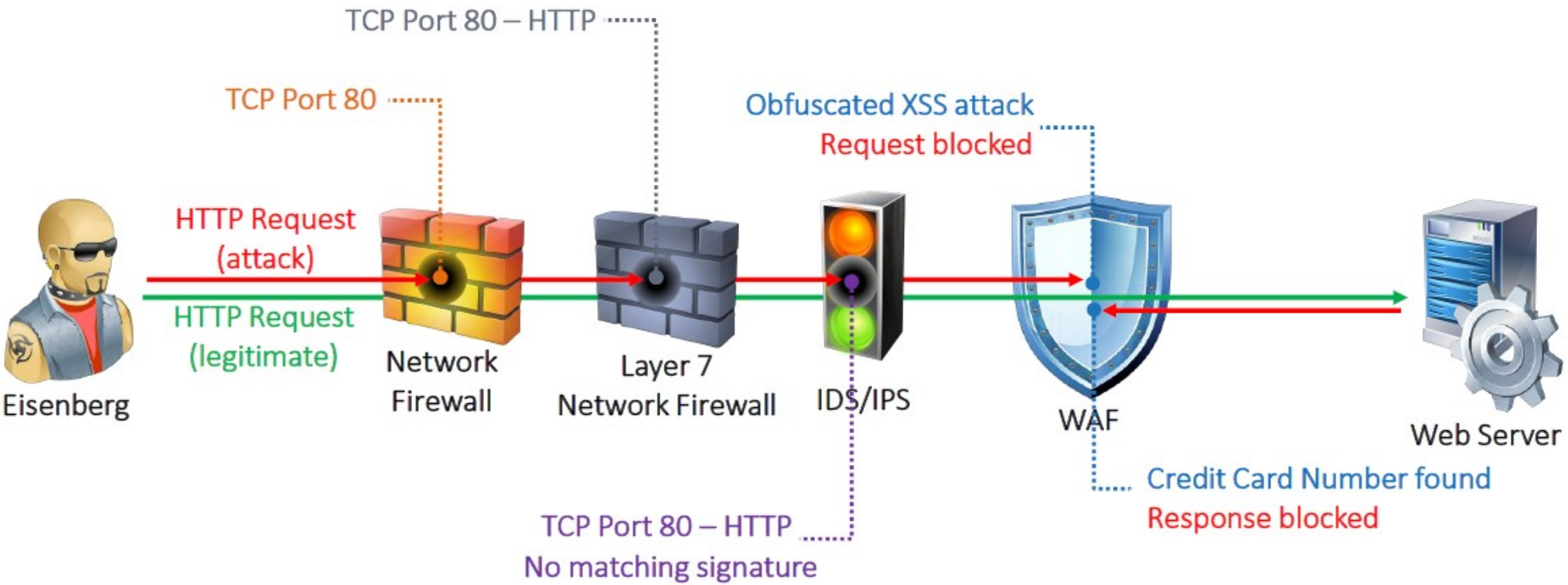
Web Application Firewall
+
Reverse Proxy

Web Application Server

Architecture View – more detailed view



Attack Example



Web Application Firewall

Description according to OWASP (Open Web Application Security Project):

*"A web application firewall (WAF) is an **application firewall** for **HTTP** applications. It applies a set of **rules** to an HTTP conversation. Generally, these rules cover common attacks such as cross-site scripting (XSS) and SQL injection.*

*While proxies generally protect clients, **WAFs protect servers**. A WAF is deployed to protect a specific web application or set of web applications. A WAF can be considered a **reverse proxy**.*

*WAFs may come in the form of an appliance, server plugin, or filter, and may be **customized** to an application. The effort to perform this customization can be significant and needs to be maintained as the application is modified"*

(Source: https://www.owasp.org/index.php/Web_Application_Firewall)

➔ It's **not** a network firewall on layer 3 or 4 but a **gateway**.

- The OWASP ModSecurity CRS Project's goal is to provide an easily "pluggable" set of generic attack detection rules that provide a base level of protection for any web application.
- Consider the Web Application Firewall Evaluation Criteria Project (WAFEC) to help evaluate commercial and open source web application firewalls.

(Source: https://www.owasp.org/index.php/Web_Application_Firewall)

Web Application Firewall

Description according to OWASP (Open Web Application Security Project):

*"A web application firewall (WAF) is an **application firewall** for **HTTP** applications. It applies a set of **rules** to an HTTP conversation. Generally, these rules cover common attacks such as cross-site scripting (XSS) and SQL injection.*

*While proxies generally protect clients, **WAFs protect servers**. A WAF is deployed to protect a specific web application or set of web applications. A WAF can be considered a **reverse proxy**.*

*WAFs may come in the form of an appliance, server plugin, or filter, and may be **customized** to an application. The effort to perform this customization can be significant and needs to be maintained as the application is modified"*

(Source: https://www.owasp.org/index.php/Web_Application_Firewall)

➔ It's **not** a network firewall on layer 3 or 4 but a **gateway**.

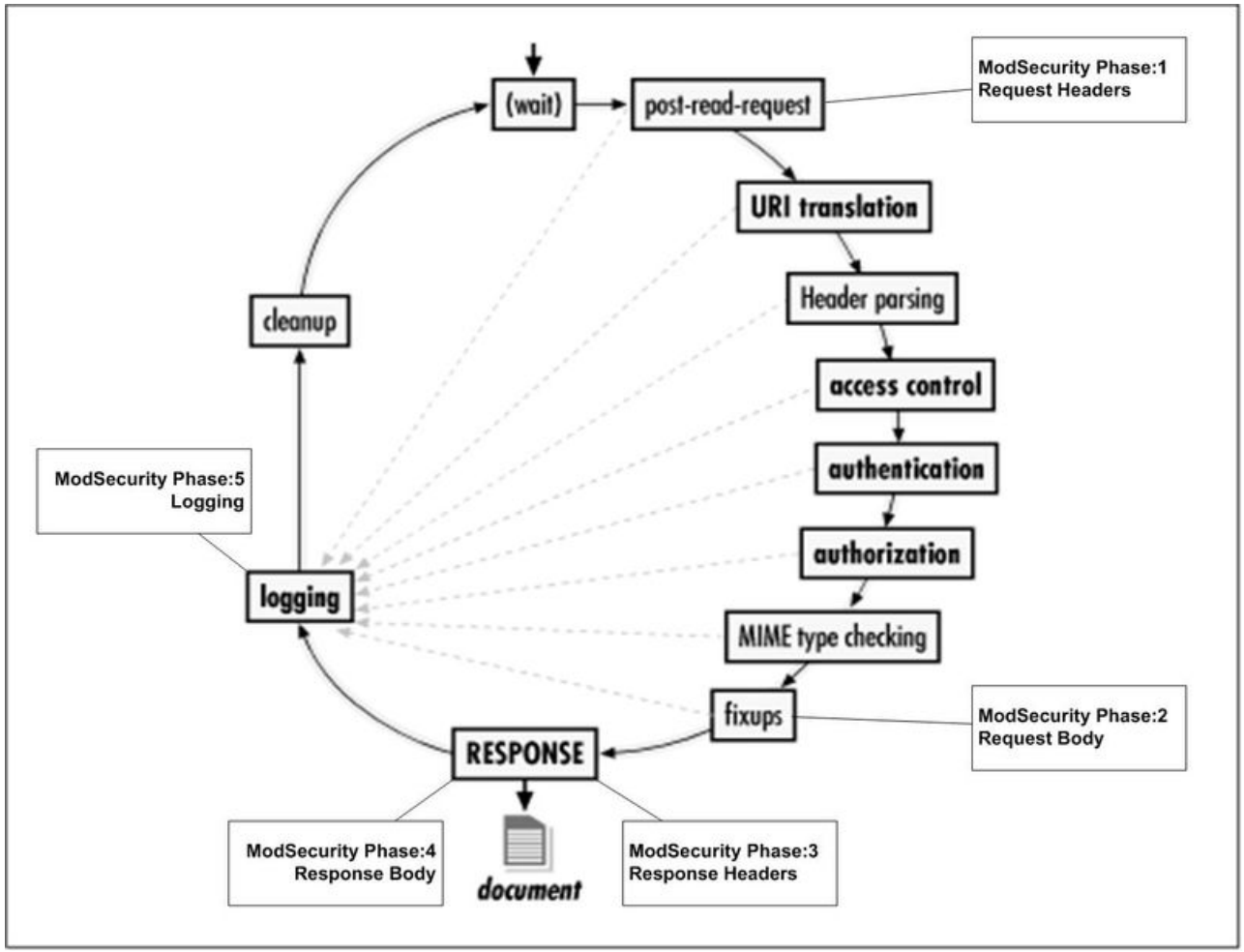
- The OWASP ModSecurity CRS Project's goal is to provide an easily "pluggable" set of generic attack detection rules that provide a base level of protection for any web application.
- Consider the Web Application Firewall Evaluation Criteria Project (WAFEC) to help evaluate commercial and open source web application firewalls.

(Source: https://www.owasp.org/index.php/Web_Application_Firewall)

ModSecurity

"Open Source Web Application Firewall"

- Open Source (Apache License 2.0)
- Engine
- Rules/Ruleset
- Traditional vs. Scoring mode
- OWASP Core Rule Set (CRS)



(Source: https://camo.githubusercontent.com/8c6d225b0c7fbb4a8a6ac1a98ead9a1d57d22fbf/687474703a2f2f7777772e6d6f647365637572697479792e6f72672f6772617068696373732f6d6f647365632d726f746174696f6e2e6a7067)

ModSecurity

Example of a ModSecurity Rule from CRS 3.2 (SQL Injection)

```
SecRule REQUEST_COOKIES|!REQUEST_COOKIES:/__utm/|REQUEST_COOKIES_NAMES|ARGS_NAMES|ARGS|XML:/* "@rx (?i)union.*?select.*?from" \
    "id:942270,\
    phase:2,\
    block,\
    capture,\
    t:none,t:urlDecodeUni,\
    msg:'Looking for basic sql injection. Common attack string for mysql, oracle and others.',\
    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:'paranoia-level/1',\
    tag:'OWASP_CRS',\
    tag:'OWASP_CRS/WEB_ATTACK/SQL_INJECTION',\
    tag:'WASCTC/WASC-19',\
    tag:'OWASP_TOP_10/A1',\
    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}',\
    setvar:'tx.anomaly_score_pl1=+%{tx.critical_anomaly_score}'"
```

Web Application Firewall

Description according to OWASP (Open Web Application Security Project):

*"A web application firewall (WAF) is an **application firewall** for **HTTP** applications. It applies a set of **rules** to an HTTP conversation. Generally, these rules cover common attacks such as cross-site scripting (XSS) and SQL injection.*

*While proxies generally protect clients, **WAFs protect servers**. A WAF is deployed to protect a specific web application or set of web applications. A WAF can be considered a **reverse proxy**.*

*WAFs may come in the form of an appliance, server plugin, or filter, and may be **customized** to an application. The effort to perform this customization can be significant and needs to be maintained as the application is modified"*

(Source: https://www.owasp.org/index.php/Web_Application_Firewall)

➔ It's **not** a network firewall on layer 3 or 4 but a **gateway**.

- The OWASP ModSecurity CRS Project's goal is to provide an easily "pluggable" set of generic attack detection rules that provide a base level of protection for any web application.
- Consider the Web Application Firewall Evaluation Criteria Project (WAFEC) to help evaluate commercial and open source web application firewalls.

(Source: https://www.owasp.org/index.php/Web_Application_Firewall)

WAFEC

Joined project between the Web Application Security Consortium (WASC) and OWASP.

Detailed guideline with categories

- Deployment Architecture
- HTTP Support
- Detection Techniques
- Protection Techniques
- Logging
- Reporting
- Management
- Performance
- XML

(Source: http://projects.webappsec.org/w/page/13246983/WAFEC%201%20HTML%20Version)

Web Application Firewall

Description according to (Open Web Application Security Project):

*"A web application firewall (WAF) is an **application firewall** for **HTTP** applications. It applies a set of **rules** to an HTTP conversation. Generally, these rules cover common attacks such as cross-site scripting (XSS) and SQL injection.*

*While proxies generally protect clients, **WAFs protect servers**. A WAF is deployed to protect a specific web application or set of web applications. A WAF can be considered a **reverse proxy**.*

*WAFs may come in the form of an appliance, server plugin, or filter, and may be **customized** to an application. The effort to perform this customization can be significant and needs to be maintained as the application is modified"*

(Source: https://www.owasp.org/index.php/Web_Application_Firewall)

➔ It's **not** a network firewall on layer 3 or 4 but a **gateway**.

- The OWASP ModSecurity CRS Project's goal is to provide an easily "pluggable" set of generic attack detection rules that provide a base level of protection for any web application.*
- Consider the Web Application Firewall Evaluation Criteria Project (WAFEC) to help evaluate commercial and open source web application firewalls.

"fully fledged"

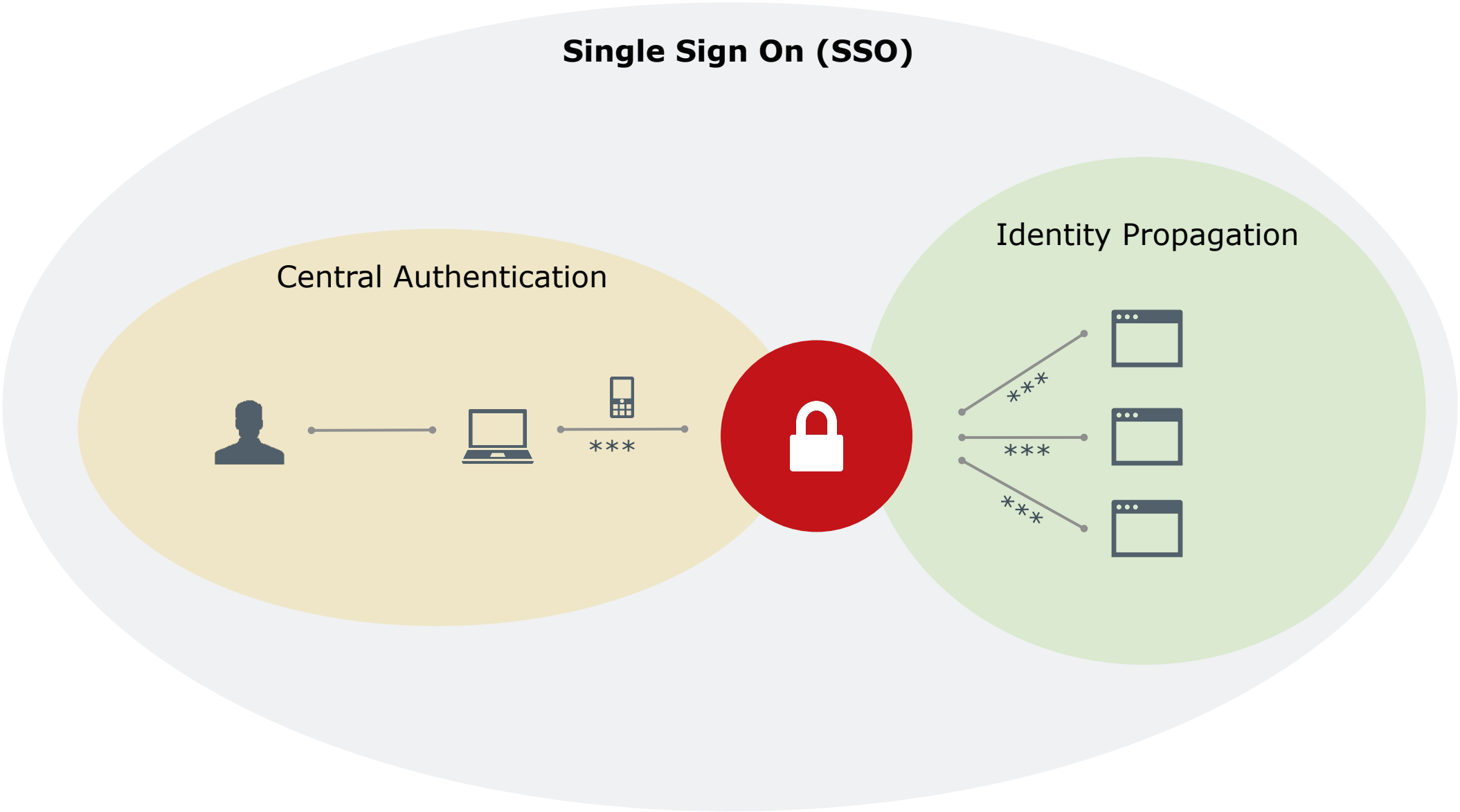
(Source: https://www.owasp.org/index.php/Web_Application_Firewall)

* Apache/ModSecurity Tutorials https://www.netnea.com/cms/apache-tutorials/

Added Value of a “fully fledged” WAF

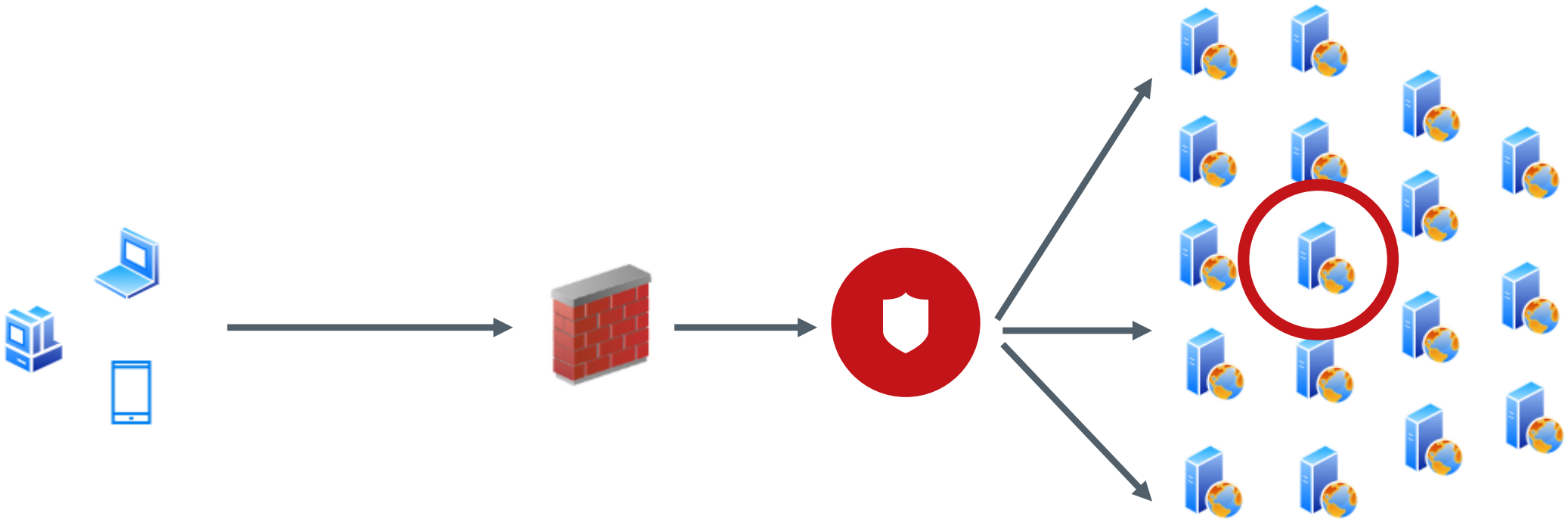
- Centralized authentication
  - 2FA if required
  - Federation support (SAML, OpenID Connect, OAuth)
- Session Management / Session Store
- Identity Propagation
- Policy-based authorization
- Single Sign On (SSO)
- …

Added Value of a “fully fledged” WAF



3. How can you adapt

Dos and Don’ts



Client (Browser, Apps)

Firewall

Web Application Firewall
+
Reverse Proxy

Web Application Server

9 Dos and Don’ts

```
POST /webapp/start?action=login HTTP/1.1
Host: www.myapp.ch
User-Agent: Mozilla/1.42
Content-Type: application/x-www-form-urlencoded
…
Cookie: Secure_WAF_Cookie: kd4iib5d6d…
EvilHeader: <script>attack</script>

userid=alice&password=12345&evilparam=evilvalue
```

HTTP Request

```
POST /webapp/start?action=login HTTP/1.1
Host: srv99.localdomain
User-Agent: Mozilla/1.42
Content-Type: application/x-www-form-
urlencoded
…
Cookie: PHPSESSID=2k3kd8f7gkv7y

userid=alice&password=12345
```

HTTP Request

1 Follow standard HTTP specs (RFC)

Client

WAF

Webserver

HTTP Response

```
HTTP/1.1 200 OK
Date: Mon, 1 Jan 2020 06:30:32 GMT
Server: dummy
Content-Type: text/html;charsest=UTF-8
…


<html>
<head</head>
  <body>
    <a href=https://www.myapp.ch/link>click me</a>
  </body>
</html>
```

HTTP Response

```
HTTP/1.1 200 OK
Date: Mon, 1 Jan 2020 06:30:32 GMT
Server: Apache/2.0.55 (Debian) PHP/5.1.2-1
mod_ssl/2.0.55 OpenSSL/0.9.8b
Content-Type: text/html;charsest=UTF-8
…

<html>
<head</head>
  <body>
    <a href=http://srv99.localdomain/link>click me</a>
  </body>
</html>
```

9 Dos and Don’ts

```
POST /webapp/start?action=login HTTP/1.1
Host: www.myapp.ch
User-Agent: Mozilla/1.42
Content-Type: application/x-www-form-urlencoded
…
Cookie: Secure_WAF_Cookie: kd4iib5d6d…
EvilHeader: <script>attack</script>

userid=alice&password=12345&evilparam=evilvalue
```

HTTP Request

```
POST /webapp/start?action=login HTTP/1.1
Host: srv99.localdomain
User-Agent: Mozilla/1.42
Content-Type: application/x-www-form-
urlencoded
…
Cookie: PHPSESSID=2k3kd8f7gkv7y

userid=alice&password=12345
```

HTTP Request

2

Don’t create links or cookies in the browser

Client

WAF

Webserver

HTTP Response

```
HTTP/1.1 200 OK
Date: Mon, 1 Jan 2020 06:30:32 GMT
Server: dummy
Content-Type: text/html;charsest=UTF-8
…


<html>
<head</head>
  <body>
    <a href=https://www.myapp.ch/link>click me</a>
  </body>
</html>
```

HTTP Response

```
HTTP/1.1 200 OK
Date: Mon, 1 Jan 2020 06:30:32 GMT
Server: Apache/2.0.55 (Debian) PHP/5.1.2-1
mod_ssl/2.0.55 OpenSSL/0.9.8b
Content-Type: text/html;charsest=UTF-8
…

<html>
<head</head>
  <body>
    <a href=http://srv99.localdomain/link>click me</a>
  </body>
</html>
```

9 Dos and Don’ts

Client

```
POST /webapp/start?action=login HTTP/1.1
Host: www.myapp.ch
User-Agent: Mozilla/1.42
Content-Type: application/x-www-form-urlencoded
…
Cookie: Secure_WAF_Cookie: kd4iib5d6d…
EvilHeader: <script>attack</script>

userid=alice&password=12345&evilparam=evilvalue
```

HTTP Request

WAF

```
POST /webapp/start?action=login HTTP/1.1
Host: srv99.localdomain
User-Agent: Mozilla/1.42
Content-Type: application/x-www-form-urlencoded
…
Cookie: PHPSESSID=2k3kd8f7gkv7y

userid=alice&password=12345
```

HTTP Request

Webserver

3

Ensure ways for identity propagation: e.g., header, NTLM, Kerberos

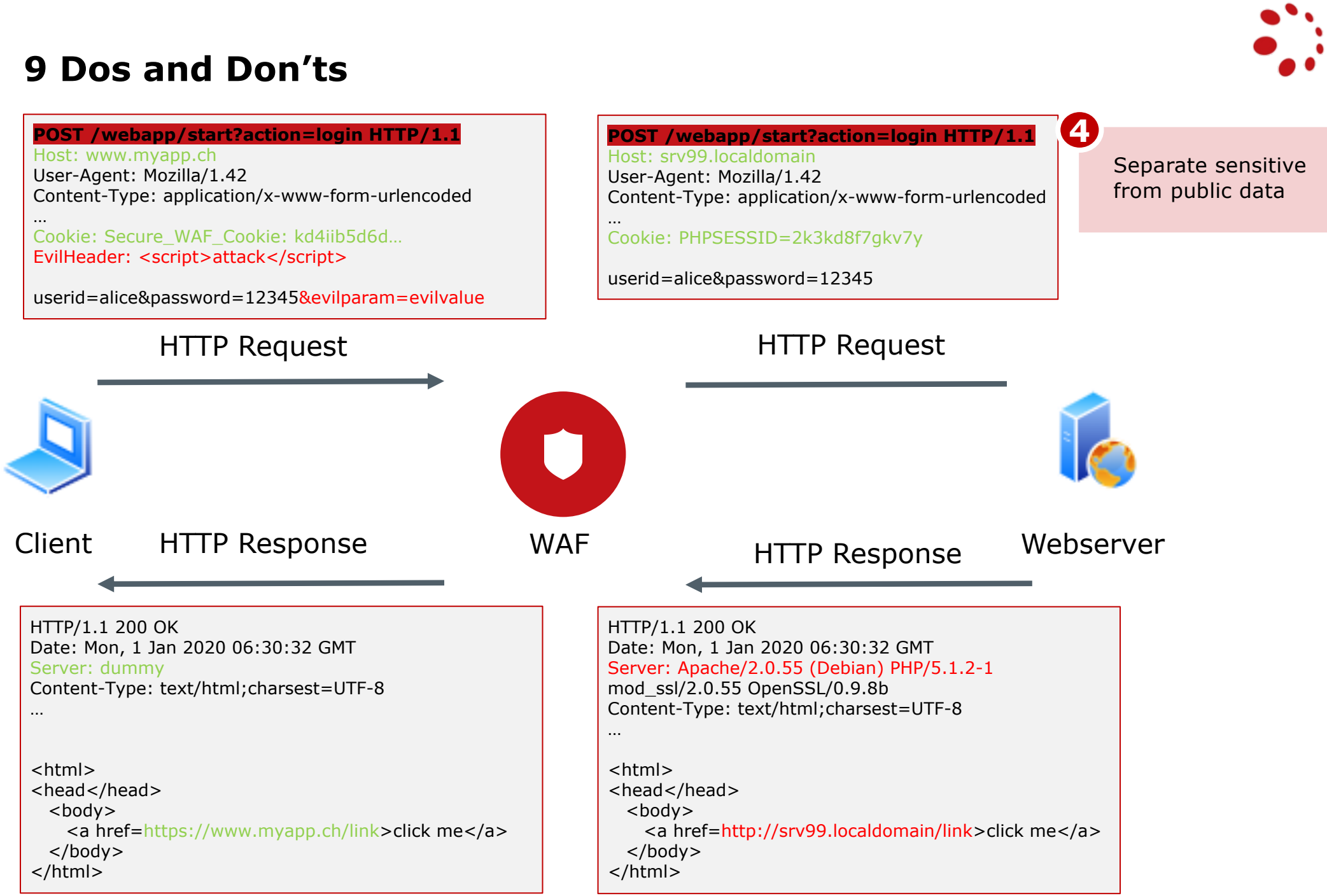
HTTP Response

```
HTTP/1.1 200 OK
Date: Mon, 1 Jan 2020 06:30:32 GMT
Server: dummy
Content-Type: text/html;charsest=UTF-8
…


<html>
<head</head>
  <body>
    <a href=https://www.myapp.ch/link>click me</a>
  </body>
</html>
```

HTTP Response

```
HTTP/1.1 200 OK
Date: Mon, 1 Jan 2020 06:30:32 GMT
Server: Apache/2.0.55 (Debian) PHP/5.1.2-1
mod_ssl/2.0.55 OpenSSL/0.9.8b
Content-Type: text/html;charsest=UTF-8
…

<html>
<head</head>
  <body>
    <a href=http://srv99.localdomain/link>click me</a>
  </body>
</html>
```

9 Dos and Don’ts



9 Dos and Don’ts

```
POST /webapp/start?action=login HTTP/1.1
Host: www.myapp.ch
User-Agent: Mozilla/1.42
Content-Type: application/x-www-form-urlencoded
…
Cookie: Secure_WAF_Cookie: kd4iib5d6d…
EvilHeader: <script>attack</script>

userid=alice&password=12345&evilparam=evilvalue
```

HTTP Request

```
POST /webapp/start?action=login HTTP/1.1
Host: srv99.localdomain
User-Agent: Mozilla/1.42
Content-Type: application/x-www-form-urlencoded
…
Cookie: PHPSESSID=2k3kd8f7gkv7y

userid=alice&password=12345
```

HTTP Request

5 Use correct MIME type headers

Client

WAF

Webserver

HTTP Response

```
HTTP/1.1 200 OK
Date: Mon, 1 Jan 2020 06:30:32 GMT
Server: dummy
Content-Type: text/html;charsest=UTF-8
…


<html>
<head</head>
  <body>
    <a href=https://www.myapp.ch/link>click me</a>
  </body>
</html>
```

HTTP Response

```
HTTP/1.1 200 OK
Date: Mon, 1 Jan 2020 06:30:32 GMT
Server: Apache/2.0.55 (Debian) PHP/5.1.2-1
mod_ssl/2.0.55 OpenSSL/0.9.8b
Content-Type: text/html;charsest=UTF-8
…

<html>
<head</head>
  <body>
    <a href=http://srv99.localdomain/link>click me</a>
  </body>
</html>
```

9 Dos and Don’ts

```
POST /webapp/start?action=login HTTP/1.1
Host: www.myapp.ch
User-Agent: Mozilla/1.42
Content-Type: application/x-www-form-urlencoded
…
Cookie: Secure_WAF_Cookie: kd4iib5d6d…
EvilHeader: <script>attack</script>

userid=alice&password=12345&evilparam=evilvalue
```

HTTP Request

```
POST /webapp/start?action=login HTTP/1.1
Host: srv99.localdomain
User-Agent: Mozilla/1.42
Content-Type: application/x-www-form-
urlencoded
…
Cookie: PHPSESSID=2k3kd8f7gkv7y

userid=alice&password=12345
```

HTTP Request

6 Use relative paths

7 Don’t use <base href= …> tags

Client

WAF

Webserver

HTTP Response

```
HTTP/1.1 200 OK
Date: Mon, 1 Jan 2020 06:30:32 GMT
Server: dummy
Content-Type: text/html;charsest=UTF-8
…


<html>
<head</head>
  <body>
    <a href=https://www.myapp.ch/link>click
me</a>
  </body>
</html>
```

HTTP Response

```
HTTP/1.1 200 OK
Date: Mon, 1 Jan 2020 06:30:32 GMT
Server: Apache/2.0.55 (Debian) PHP/5.1.2-1
mod_ssl/2.0.55 OpenSSL/0.9.8b
Content-Type: text/html;charsest=UTF-8
…

<html>
<head</head>
  <body>
    <a href=http://srv99.localdomain/link>click
me</a>
  </body>
</html>
```

9 Dos and Don’ts

```
POST /webapp/start?action=login HTTP/1.1
Host: www.myapp.ch
User-Agent: Mozilla/1.42
Content-Type: application/x-www-form-urlencoded
…
Cookie: Secure_WAF_Cookie: kd4iib5d6d…
EvilHeader: <script>attack</script>

userid=alice&password=12345&evilparam=evilvalue
```

HTTP Request

```
POST /webapp/start?action=login HTTP/1.1
Host: srv99.localdomain
User-Agent: Mozilla/1.42
Content-Type: application/x-www-form-
urlencoded
…
Cookie: PHPSESSID=2k3kd8f7gkv7y

userid=alice&password=12345
```

HTTP Request

8 Return proper HTTP error codes

9 Trigger proper relogin for asynchronous requests

Client

WAF

Webserver

HTTP Response

```
HTTP/1.1 200 OK
Date: Mon, 1 Jan 2020 06:30:32 GMT
Server: dummy
Content-Type: text/html;charsest=UTF-8
…


<html>
<head</head>
  <body>
    <a href=https://www.myapp.ch/link>click me</a>
  </body>
</html>
```

HTTP Response

```
HTTP/1.1 200 OK
Date: Mon, 1 Jan 2020 06:30:32 GMT
Server: Apache/2.0.55 (Debian) PHP/5.1.2-1
mod_ssl/2.0.55 OpenSSL/0.9.8b
Content-Type: text/html;charsest=UTF-8
…

<html>
<head</head>
  <body>
    <a href=http://srv99.localdomain/link>click me</a>
  </body>
</html>
```

9 Dos and Don’ts

1. Follow standard HTTP specs (RFC)
2. Don’t create links or cookies in the browser
3. Ensure ways for identity propagation: e.g. header, NTLM, Kerberos
4. Separate sensitive from public data
5. Use current MIME type headers
6. Use relative paths
7. Don’t use <base href=…> tags
8. Return proper HTTP error codes
9. Trigger proper relogin for asynchronous requests

Recap – final architecture proposal

UNITED SECURITY PROVIDERS

Thanks for joining

https://united-security-providers.ch