



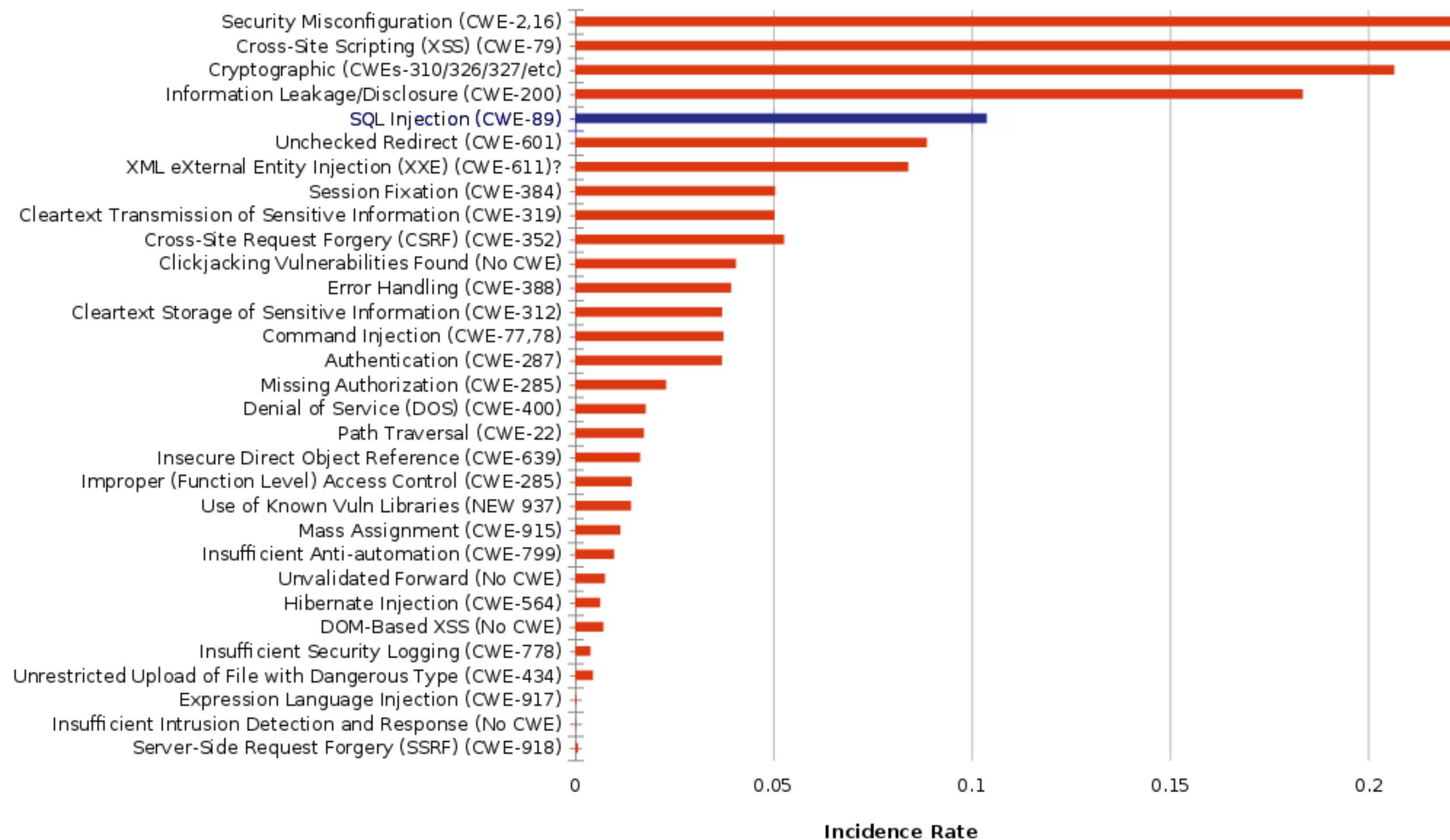
OWASP TOP TEN 2017

WHAT'S NEW?

WHAT IS THIS...

OWASP TOP TEN?

Incidence Rate per CWEs



source: Data collected by the OWASP project - see last slide for a link

THE NEW TOP TEN

SQL INJECTION

SENSITIVE DATA EXPOSURE

SECURITY MISCONFIGURATION

VULNERABLE COMPONENTS

CROSS SITE SCRIPTING

XML EXTERNAL ENTITIES

INSUFFICIENT LOGGING & MONITORING

BROKEN AUTHENTICATION

INSECURE DESERIALISATION

BROKEN ACCESS CONTROL

| OWASP Top 10 - 2013 | ➔ | OWASP Top 10 - 2017 |
|--|---|--|
| A1 – Injection | ➔ | A1:2017-Injection |
| A2 – Broken Authentication and Session Management | ➔ | A2:2017-Broken Authentication |
| A3 – Cross-Site Scripting (XSS) | ➡ | A3:2017-Sensitive Data Exposure |
| A4 – Insecure Direct Object References [Merged+A7] | U | A4:2017-XML External Entities (XXE) [NEW] |
| A5 – Security Misconfiguration | ➡ | A5:2017-Broken Access Control [Merged] |
| A6 – Sensitive Data Exposure | ↗ | A6:2017-Security Misconfiguration |
| A7 – Missing Function Level Access Contr [Merged+A4] | U | A7:2017-Cross-Site Scripting (XSS) |
| A8 – Cross-Site Request Forgery (CSRF) | ☒ | A8:2017-Insecure Deserialization [NEW, Community] |
| A9 – Using Components with Known Vulnerabilities | ➔ | A9:2017-Using Components with Known Vulnerabilities |
| A10 – Unvalidated Redirects and Forwards | ☒ | A10:2017-Insufficient Logging&Monitoring [NEW,Comm.] |

**THE TOP TEN IS ONLY THE
BEGINNING...**

Smart people, probably

BROKEN AUTHENTICATION

Systems with broken authentication will allow attackers to impersonate your users and administrators

WHAT TO LOOK FOR

- ▶ Permits Automated attacks - brute force or dict.
- ▶ Exposure of session IDs
- ▶ Weak password hashing / cleartext storage
- ▶ Allow weak passwords (e.g admin 12345)
- ▶ Doesn't rotate session IDs
- ▶ Missing or ineffective Multi-factor Auth. (MFA)

>_ DEMO TIME

XML EXTERNAL ENTITIES

XXE Flaws allow attackers to extract data, execute remote HTTP requests from the vulnerable server and perform denial of service attacks

WHAT TO LOOK FOR

- ▶ Applications that accept XML upload
- ▶ Relies on Document Type Declaration Validation
- ▶ Relies on External Entity Resolution
- ▶ Use of SAML for federated identity
- ▶ SOAP < 1.2 that use XML entities

BASIC IDEA

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE foo [
  <!ELEMENT foo ANY >
  <!ENTITY bar "BAA BAA BLACK SHEEP" >
]>
<foo> &bar; </foo>
```

RESULT

```
<foo> BAA BAA BLACK SHEEP </foo>
```

WHAT IS VULNERABLE?

SAFE BY DEFAULT

- ▶ LINQ TO XML
- ▶ XDocument
- ▶ XmlDictionaryReader
- ▶ XmlDocument 4.5.2+
- ▶ XmlNodeReader
- ▶ XmlReader
- ▶ XmlTextReader 4.5.2+
- ▶ XPathNavigator 4.5.2+
- ▶ XslCompiledTransform

UNSAFE BY DEFAULT

- ▶ XmlDocument < 4.5.2
- ▶ XmlTextReader < 4.5.2
- ▶ XPathNavigator < 4.5.2

source: OWASP XXE Cheat sheet -
see last slide for link

> DEMO TIME

WHAT TO LOOK FOR

```
var xDoc = new XmlDocument();  
var settings = new XmlReaderSettings() jim, 3 days ago · removed unus  
{  
    MaxCharactersFromEntities = 0,  
    MaxCharactersInDocument = 0,  
    DtdProcessing = DtdProcessing.Parse,  
    XmlResolver = new XmlUrlResolver()  
};  
  
using (var reader = XmlReader.Create(bankFile.OpenReadStream(), settings))  
{
```

PREVENTION

- ▶ Avoid use of DTD Parsing and expansion
- ▶ Turn off entity expansion
- ▶ Use XML Schema validation server side
- ▶ Use JSON
- ▶ Minimise what you serialise

INSECURE DESERIALIZATION

Applications that have to deserialise objects from untrusted sources risk being subjected to object tampering and malicious payloads

WHAT TO LOOK FOR

- ▶ System calls that take input from deserialised object are very bad
- ▶ Applications that change their state based on a deserialised object
- ▶ Pay attention to Remote Procedure Calls, JSON with no signatures, HTTP Cookies etc.

HOW TO ADDRESS

- ▶ Avoid deserialised objects from untrusted sources!
- ▶ Use integrity checks, Hashed Message Authentication Codes, digital signatures etc.
- ▶ Strict Type checking
- ▶ Log log log... a single user constantly deserialising things for a few hours? Hmm...

> DEMO TIME

INSUFFICIENT LOGGING AND MONITORING

- ▶ Lack of centralisation & correlation
- ▶ High value events, or transactions aren't logged
- ▶ Insufficient detail in error messages
- ▶ local storage
- ▶ No alerts or escalation processes
- ▶ "Sink-holed" unhandled exceptions

HOW TO FIX

- ▶ No excuses anymore for not having a centralised logging server
- ▶ Think about potential points of abuse, high value targets (logins etc) log these things!
- ▶ Unhandled exceptions should be alerted and actioned
- ▶ Log -> Monitor -> Alert -> Escalate

GRAYLOG

- ▶ Centralised server, can deploy on prem, or on IaaS
- ▶ Free to use, pay for support, open source
- ▶ Integrates with LDAP
- ▶ REST API
- ▶ GELF Protocol over UDP/TCP
- ▶ Easy integration with NLog or Serilog
- ▶ Nice & simple alerting system
- ▶ Elastic search under the hood

Streams

You can route incoming messages into streams by applying rules against them. If a message matches all rules of a stream it is routed into it. A message can be routed into multiple streams. You can for example create a stream that contains all SSH logins and configure to be alerted whenever there are more logins than usual. Read more about streams in the [documentation](#).

[Create Stream](#)

Take a look at the [Graylog stream dashboards](#) for wall-mounted displays or other integrations.

Successful user logins

All successful user logins

0 messages/second, Must match all of the 2 configured stream rule(s). [Show stream rules](#)

[Edit rules](#)[Manage outputs](#)[Manage alerts](#)[Pause stream](#)[More actions ▾](#)

Failed user logins

All failed user logins

0 messages/second, Must match all of the 1 configured stream rule(s). [Show stream rules](#)

[Edit rules](#)[Manage outputs](#)[Manage alerts](#)[Pause stream](#)[More actions ▾](#)

John workstation realtime debugging (ignore) stopped

Just me debugging locally

0 messages/second, No configured rules. [Show stream rules](#)

[Edit rules](#)[Manage outputs](#)[Manage alerts](#)[Start stream](#)[More actions ▾](#)

Exceptions on all platforms

All exceptions anywhere in the stack

0 messages/second, Must match all of the 1 configured stream rule(s). [Show stream rules](#)

[Edit rules](#)[Manage outputs](#)[Manage alerts](#)[Pause stream](#)[More actions ▾](#)

Firewall messages

All logs from all firewalls

8 messages/second, Must match all of the 4 configured stream rule(s). [Show stream rules](#)

[Edit rules](#)[Manage outputs](#)[Manage alerts](#)[Pause stream](#)[More actions ▾](#)

HTTP 500's

All HTTP requests that we answered in the 500 range

0 messages/second, Must match all of the 1 configured stream rule(s). [Show stream rules](#)

[Edit rules](#)[Manage outputs](#)[Manage alerts](#)[Pause stream](#)[More actions ▾](#)

GRAYLOG - DOCKER COMPOSE

```
version: '2'      Jim Burger, a day ago · /logging-fixed:
services:
  some-mongo:
    image: "mongo:3"
  some-elasticsearch:
    image: "elasticsearch:2"
    command: "elasticsearch -Des.cluster.name='graylog'"
  graylog:
    image: graylog2/server:2.1.1-1
    environment:
      GRAYLOG_PASSWORD_SECRET: [REDACTED]
      GRAYLOG_ROOT_PASSWORD_SHA2: [REDACTED]
      GRAYLOG_WEB_ENDPOINT_URI: http://127.0.0.1:9000/api
    links:
      - some-mongo:mongo
      - some-elasticsearch:elasticsearch
    ports:
      - "9000:9000"
      - "12201:12201/udp"
```


SERilog - ASPNET CORE DEPENDENCIES


```
<ItemGroup>
  <PackageReference Include="Microsoft.AspNetCore.All" Version="2.0.5" />
  <PackageReference Include="newtonsoft.json" Version="11.0.2" />
  <PackageReference Include="serilog" Version="2.6.0" />
  <PackageReference Include="serilog.extensions.logging" Version="2.0.2" />
  <PackageReference Include="serilog.sinks.graylog" Version="1.6.0" />
</ItemGroup>
```

Jim Burger, a day ago ·

SERilog - ASPNET CORE SETUP

```
public Startup(IConfiguration configuration, IHostingEnvironment env)
{
    Log.Logger =
        new LoggerConfiguration()
            2 references
            .Enrich.FromLogContext()
            .WriteTo.Graylog(new GraylogSinkOptions
            {
                0 references
                HostnameOrAddress = configuration.GetValue<string>("application:grayloghost"),
                Port = configuration.GetValue<int>("application:graylogport")
            })
            .CreateLogger();
    Configuration = configuration;
}

public void ConfigureServices(IServiceCollection services)
{
    ConfigureDependencies(services);
    services.Configure<ApplicationSettings>(Configuration.GetSection("Application"));
    services.AddMvc();
    services.AddSession();
    services.AddAntiforgery();
    services.AddLogging(logBuilder => logBuilder.AddSerilog(dispose: true));
}
```



The diagram consists of two blue arrows. The first arrow originates from the right side of the code block and points to the `new LoggerConfiguration()` line in the `Startup` class. The second arrow originates from the right side of the code block and points to the `services.AddSerilog(dispose: true)` line in the `ConfigureServices` method.

> DEMO TIME

- ▶ **OWASP TOP 10 DATA**

<https://github.com/OWASP/Top10/tree/master/2017/datacall/analysis>

- ▶ **OWASP TOP 10 HTML VERSION**

https://www.owasp.org/index.php/Category:OWASP_Top_Ten_2017_Project

- ▶ **ADVANCED XXE**

<https://www.blackhillsinfosec.com/xml-external-entity-beyond-etcpasswd-fun-profit/>

- ▶ **XXE PREVENTION CHEATSHEET**

[https://www.owasp.org/index.php/XML_External_Entity_\(XXE\)_Prevention_Cheat_Sheet](https://www.owasp.org/index.php/XML_External_Entity_(XXE)_Prevention_Cheat_Sheet)

- ▶ **GRAYLOG**

<https://www.graylog.org/>

- ▶ **SONARQUBE**

<https://www.sonarqube.org/>

A LITTLE MORE ABOUT ME



<https://twitter.com/burgomg>



<https://github.com/jburger>



<https://bit.ly/2EyISJY>



<https://medium.com/defence-in-depth>