25 Most Dangerous Software Weaknesses

Pete Freitag, Foundeo Inc.

foundeo inc.

About Me

Pete Freitag

foundeo inc.

- 20+ Years ColdFusion Experience
- Company: Foundeo Inc.
  - Products: FuseGuard, HackMyCF, Fixinator
  - Consulting: Code Reviews, Server Review, CFML Security Training
- blog: petefreitag.com
- twitter/github: @pfreitag
- email: pete@foundeo.com

Where did the “25 Most Dangerous Software Weaknesses” list come from?

The CWE Top 25 List

Photo by Claus Jensen on Unsplash

CWE - Common Weakness Enumeration

From the people who brought you the CVE: Mitre

CWE Top 25

Scoring Metrics

Rank	CWE	NVD Count	Avg CVSS	Overall Score
1	CWE-787	4123	7.93	64.20
2	CWE-79	4740	5.73	45.97
3	CWE-89	1263	8.66	22.11
4	CWE-20	1520	7.19	20.63
5	CWE-125	1489	6.54	17.67
6	CWE-78	999	8.67	17.53
7	CWE-416	1021	7.79	15.50
8	CWE-22	1010	7.32	14.08
9	CWE-352	847	7.20	11.53
10	CWE-434	551	8.61	9.56
11	CWE-476	611	6.49	7.15
12	CWE-502	378	8.73	6.68
13	CWE-190	452	7.52	6.53
14	CWE-287	412	7.88	6.35
15	CWE-798	333	8.48	5.66
16	CWE-862	468	6.53	5.53
17	CWE-77	325	8.36	5.42
18	CWE-306	328	8.00	5.15
19	CWE-119	323	7.73	4.85
20	CWE-276	368	7.04	4.84
21	CWE-918	317	7.16	4.27
22	CWE-362	301	6.56	3.57
23	CWE-400	277	6.93	3.56
24	CWE-611	232	7.58	3.38
25	CWE-94	192	8.60	3.32

Source: https://cwe.mitre.org/top25/archive/2022/2022_cwe_top25.html

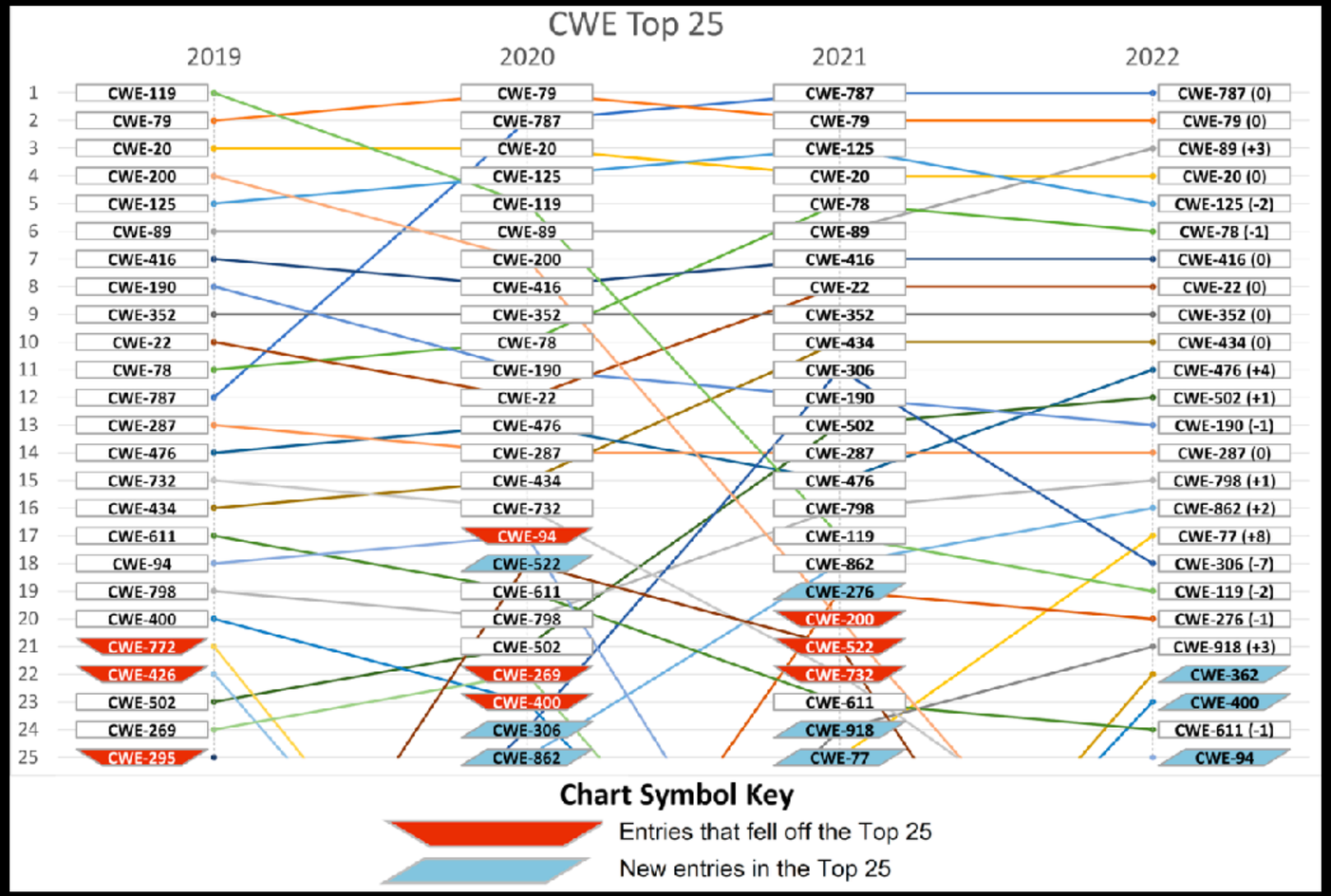
The thing about TOP N Lists

There are more than 25 CWE's…

Rank	CWE	Name	NVD Count	Avg CVSS	Overall Score	KEV Count (CVEs)	Rank Change vs. 2021
26	CWE-295	Improper Certificate Validation	242	6.95	3.12	2	0
27	CWE-427	Uncontrolled Search Path Element	211	7.66	3.12	0	+7 ▲
28	CWE-863	Incorrect Authorization	250	6.76	3.10	0	+10 ▲
29	CWE-269	Improper Privilege Management	207	7.67	3.06	3	0
30	CWE-732	Incorrect Permission Assignment for Critical Resource	212	7.31	2.93	1	-8 ▼
31	CWE-843	Access of Resource Using Incompatible Type ('Type Confusion')	173	8.34	2.87	10	+5 ▲
32	CWE-668	Exposure of Resource to Wrong Sphere	230	6.48	2.68	0	+21 ▲
33	CWE-200	Exposure of Sensitive Information to an Unauthorized Actor	241	5.99	2.49	2	-13 ▼
34	CWE-1321	Improperly Controlled Modification of Object Prototype Attributes ('Prototype Pollution')	140	8.77	2.48	0	new
35	CWE-601	URL Redirection to Untrusted Site ('Open Redirect')	230	6.04	2.41	0	+2 ▲
36	CWE-401	Missing Release of Memory after Effective Lifetime	195	6.71	2.39	0	-4 ▼
37	CWE-59	Improper Link Resolution Before File Access ('Link Following')	183	7.00	2.38	4	-6 ▼
38	CWE-522	Insufficiently Protected Credentials	180	6.80	2.25	0	-17 ▼
39	CWE-319	Cleartext Transmission of Sensitive Information	174	6.74	2.15	0	-4 ▼
40	CWE-312	Cleartext Storage of Sensitive Information	182	6.25	2.01	0	+1 ▲

Source: https://cwe.mitre.org/top25/archive/2022/2022_cwe_top25.html

Trends of the CWE Top 25



Source: https://cwe.mitre.org/top25/archive/2022/2022_cwe_top25.html

#25 - Improper Control of Generation of Code

CWE-94 Code Injection

- A few different ways this can happen in CFML, most common:
  - Evaluate
  - IIF
  - cfinclude

#25 Fixing RCE

Replace IIF with Ternary Operator

```
<cfset greet = iif( len(name), de("Hi #name#"), de("Hi") )>
```

```
<cfset greet = ( len(name) ) ? "Hi #name#" : "Hi">
```

#25 Fixing RCE

Fixing Evaluate

```
evaluate(“url.#name#”)
```

```
url[name]
```

Rid your code of evaluate() - terrible for both performance and security

#25 RCE

Good News / Bad News

- Good News
  - Easy to find
  - Easy to fix
- Bad News
  - Very Dangerous
  - Might have a lot if your code was written early 2000’s

#24 - Improper Restriction of XML External Entity Reference

CWE-611 XXE

- What is an XML Entity? Similar to HTML Entities…
  - <
  - "
- BUT Xml Lets you define your own entities

#24 - XML Entities

XML Lets you define your own custom entities

```
<?xml version="1.0" ?>
<!DOCTYPE d [
  <!ENTITY xxe SYSTEM "http://httpbin.org/uuid">]>
<tag>&xxe;</tag>
```

Example: https://trycf.com/gist/64b7ec4e8d6774a51c1e99552030df7f/acf2021?theme=monokai

#24 - XML Entities
Mitigating

- No Global API in Adobe ColdFusion To Disable XML Entities: Feature Request #CF-4201057
  - You can pass: xmlParse(xml, false, {allowExternalEntities=false}) in ACF (thanks Brian!)
- Avoid Parsing XML From Untrusted Sources
  - HTTP Requests, APIs, Feeds
- Check XML For Custom DocTypes, ELEMENT Definitions
- Use a safer API:
    - SafeXmlParse: https://github.com/foundeo/safexmlparse
    - Convert to JSON: https://gamesover2600.tumblr.com/post/180776378949/convert-xml-to-json-in-1-line-of-code-using

#23 Uncontrolled Resource Consumption
CWE-400

- Denial of Service
  - Use LIMIT / TOP on Queries
  - Add a maximum iteration when looping over an untrusted input

#22 Concurrent Execution using Shared Resource with Improper Synchronization

CWE-362 - Race Condition

- Ensure Proper Use of cflock
- Use var / local scope within parallel ArrayEach

#21 - Server-Side Request Forgery (SSRF)

CWE-918

- SSRF Happens When your server makes a HTTP request to an arbitrary URL
- Can allow attacker to hit other http services behind the firewall (dbs, caches)
- Cloud Metadata APIs can leak access keys or other sensitive info:
  - Tip: on AWS Disable IMDSv1 and use IMDSv2 instead

#21 - SSRF

Some Functions / Tags That Can Request a URL

- cfhttp
- PDF: cfdocument / cfhtmltopdf (within HTML: img, iframe, etc)
- Images: isImageFile
- XML: XmlParse, XmlSearch, XmlValidate
- Additional List: https://hoyahaxa.blogspot.com/2021/04/ssrf-in-coldfusioncfml-tags-and.html

#20 - Incorrect Default Permissions
CWE-276

- Use OS File System Permissions
  - Can your CFML app write anywhere it wants? Does it need to?
- Use the Lockdown Guide

#19 Improper Restriction of Operations within the Bounds of a Memory Buffer

CWE-119

“Java is said to be **memory-safe** because its runtime error detection checks array bounds and pointer dereferences.”

Source: https://en.wikipedia.org/wiki/Memory_safety

Photo by Tyler Nix on Unsplash

#18 - Missing Authentication for Critical Function

CWE-306

- Authentication = User is who they attest to be
- Authorization = User has permission to perform action
- Without Authentication you cannot have Authorization
- Recent Example: Optus Data Breach
  - Allegedly there was an API exposed to the internet without any authentication.
- Check your app to make sure it requires authentication where it should. For bonus points automate this check in unit / integration tests.

#17 Improper Neutralization of Special Elements used in a Command

CWE-77 Command Injection

- Take care when using cfexecute, or other APIs that may wrap a native command

```
<cfexecute name="c:\bin\tool.exe" arguments="-n #url.n#">
```

#16 - Missing Authorization

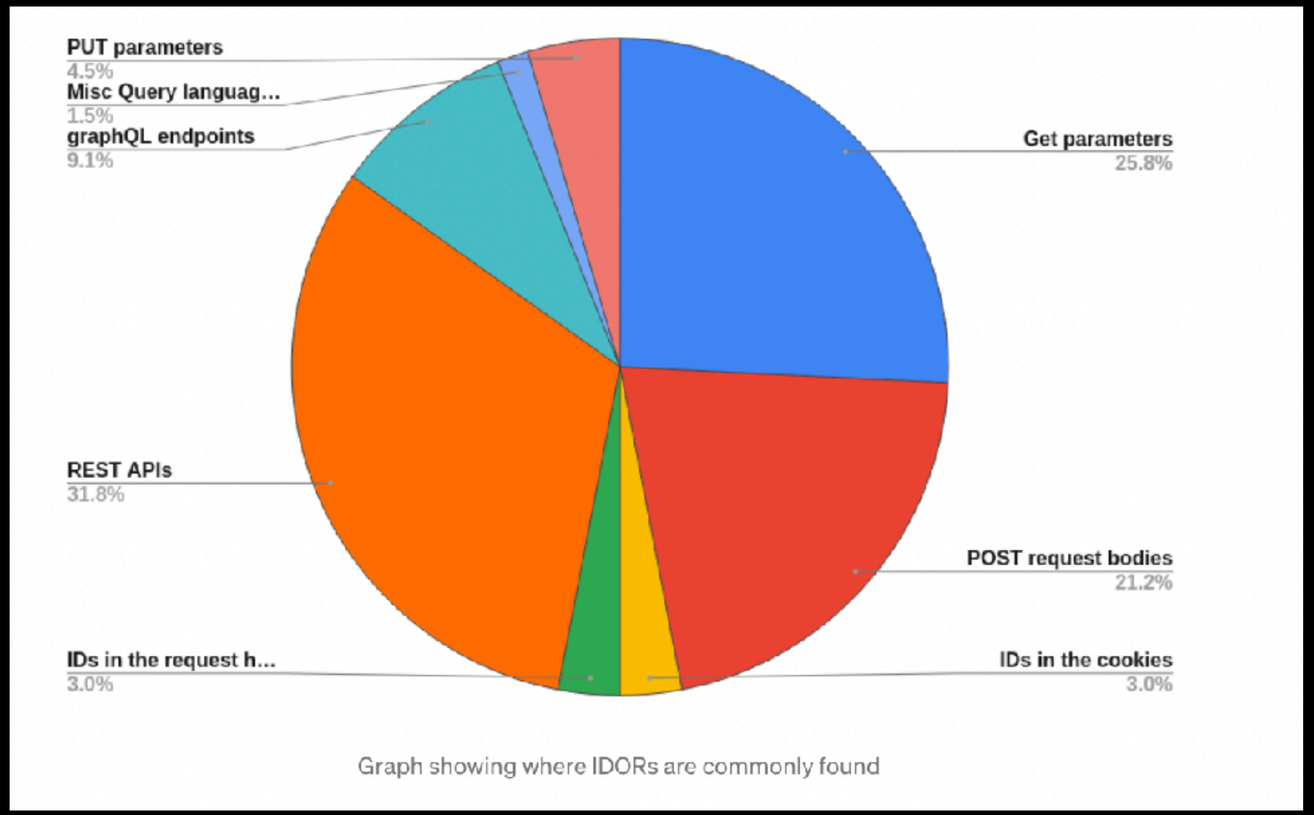
CWE-862

- Does your code check that the user is allowed to perform the requested function?
- IDOR: Insecure Direct Object Reference: document.cfm?id=123
- Sounds simple but these types of issues fall under the radar, because “it works”
  - Need to test that it “doesn’t work” for X role
  - No easy way to do this, but you can write tests

#16 - Missing Authorization

Where are IDORs Commonly Found?

1. REST APIs 31.8%
2. GET parameters 25.8%
3. POST request bodies 21.2%
4. graphQL endpoints 9.1%
5. PUT parameters 4.5%
6. IDs in the request header 3.0%
7. IDs in the cookies 3.0%
8. Misc Query langauges 1.5%



Graph showing where IDORs are commonly found

Source: https://medium.com/@nynan/what-i-learnt-from-reading-220-idor-bug-reports-6efbea44db7

#15 - Use of Hard-coded Credentials
CWE-798

- API Keys / Passwords in Code
- Embedded / Hard Coded Certificates

#15 - Hard Coded Credentials

Avoiding Hard Coded Credentials

- Environment Variables
- Secure Key Store Services:
    - Self Hosted: Hashicorp Vault
    - AWS: EC2 Parameter Store, KMS, Secrets Manager
    - Azure: Key Vault
    - GCP: Key Vault, Secret Manager

#14 - Improper Authentication
CWE-287

- So many ways Authentication can go wrong…
  - Weak Passwords, Credential Stuffing, Weak Session Cookie Config
- Use SSO
  - Most orgs now have the ability a SSO provider, either through ActiveDirectory, Google Apps, Okta, etc.
  - You can use SAML to integrate with the identity provider in your ColdFusion Apps. SAML Features added to CF2021

#13 - Integer Overflow or Wraparound
CWE-190

- Max value of a 32 bit unsigned integer is 4,294,967,295
- What happens when you add 1?
  - In MySQL < 5.5.5 it silently wraps around to 0

#12 - Deserialization of Untrusted Data

CWE-502

- Java Deserialization Vulnerabilities
  - Has the ability to cause remote code execution if malicious content is added to the serialized class.
- Avoid Untrusted Input to ColdFusion’s Deserialize Function
- Good reason to make sure you have blocked the flash remoting endpoints
- JSON Deserialization - Consider Validating JSON with a JSON schema first

#11 - NULL Pointer Dereference

CWE-476

#10 - Unrestricted Upload of File with Dangerous Type

CWE-434

- Regularly review all your file upload code, and make sure that it:
  - Always checks the file extensions of uploaded files against a list of allowed extensions.
  - Does not upload directly under the web root (at least before validation)
  - Don’t rely on mime type checks alone, they don’t work!

#9 - Cross-Site Request Forgery (CSRF)

CWE-352

- Causing a request to be made by an **authenticated** and **authorized** user's browser to perform an unwanted action.

#9 - CSRF

Best Example

- Netflix in 2006 - Remember when they rented DVDs?
  - To Add a Movie to your Queue:
    - Request to: **http://www.netflix.com/AddToQueue**
    - Pass a movie id: **movieid=70011204**

#9 CSRF

Netflix Example

Step 1: Create a Web Page With The Following img tag:

```
<img src="http://www.netflix.com/AddToQueue?movieid=70011204">
```

Step 2: Get People to Visit the Page

Step 3: Millions of people have added *Sponge Bob Square Pants the Movie* to their Queue

#9 CSRF
Fixing CSRF

- SameSite Cookies
- Check HTTP Method (eg: require POST)
- CAPTCHAs - Helpful but causes usability issues
- Use a CSRF Token
  - CFML Functions: CSRFGenerateToken() and CSRFVerifyToken()

#8 - Improper Limitation of a Pathname to a Restricted Directory

CWE-22 Path Traversal

- Path Traversal can happen whenever you construct a file path with unsafe variables.
- Example:

```
<cfinclude template="html/#url.name#">
```

#7 - Use After Free

CWE-416

#6 - Improper Neutralization of Special Elements used in an OS Command

CWE-78 - OS Command Injection

- Similar / Child of to #17, CWE-77
  - Command vs OS Command
  - Same Protections Apply

#5 - Out-of-bounds Read

CWE-125

#4 - Improper Input Validation
CWE-20

- This is a catch all CWE
    - Almost all vulnerabilities are caused by failing to validate an input!
    - TLDR: Add validation, improve security

#3 - Improper Neutralization of Special Elements used in an SQL Command

CWE-89 SQL Injection

- Classic Example:

```
<cfquery>
  SELECT story
  FROM news
  WHERE id = #url.id#
</cfquery>
```

#3 SQL Injection
Fixing SQL Injection

- Use cfqueryparam

```
<cfquery>
  SELECT story
  FROM news
  WHERE id = <cfqueryparam value="#url.id#">
</cfquery>
```

#3 SQL Injection

With queryExecute

```
queryExecute(“SELECT story
  FROM news
  WHERE id = #url.id#”);
```

```
queryExecute(“SELECT story
  FROM news
  WHERE id = :id”, {id=url.id} );
```

#3 - SQL Injection

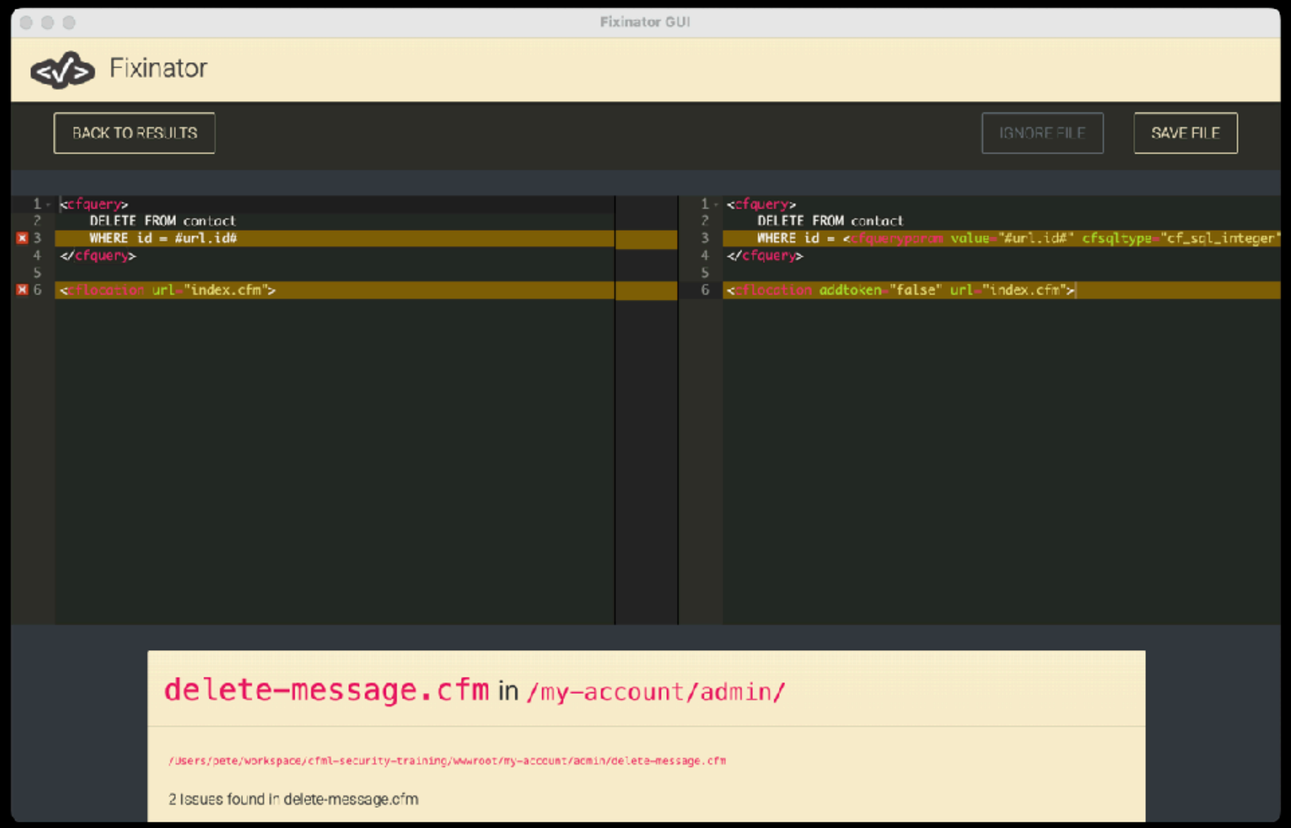
When Parameters Don’t Work

- Places that parameters *may (depending on DB)* not work:
    - ORDER BY clause
    - SELECT TOP n
    - LIMIT / OFFSET
- Validate!
- Use SELECT TOP #int(url.n)#
- Use cfqueryparam whenever you can

#3 - SQL Injection

Auto Fixing SQL Injection

- Fixinator can scan your code and autofix certain vulnerabilities



#2 - Improper Neutralization of Input During Web Page Generation

CWE-79 Cross-site Scripting / XSS

```
<cfoutput>Hello #url.name#</cfquery>
```

#2 - Fixing XSS

Encoder Methods

```
<cfoutput>Hello #encodeForHTML(url.name)#</cfquery>
```

```
<cfoutput encodefor="html">Hello #url.name#</cfquery>
```

#2 Fixing XSS

Picking the correct encoder

Context	Method
HTML	`encodeForHTML(variable)`
HTML Attribute	`encodeForHTMLAttribute(variable)`
JavaScript	`encodeForJavaScript(variable)`
CSS	`encodeForCSS(variable)`
URL	`encodeForURL(variable)`

#1 - Out-of-bounds Write

CWE-787

Learn More

- ColdFusion Security Class - Online in December:
  - https://foundeo.com/consulting/coldfusion/security-training/
- Ask Me
- Resources: OWASP, CWE Site

foundeo inc.

Thank You!

pete@foundeo.com