# Protecting Against Injection and Inclusion

Josh Cummings
PRINCIPAL SOFTWARE ENGINEER

@jzheaux   blog.jzheaux.io

## Preventing Injection, Part I

**Identify the Use Case**

Does this system need that data?

**Use a Library**

Where possible, don't roll your own security

**Turn Off Features**

The most secure machine is the one that's turned off

# SQL Injection Example

| HTTP Query | SQL Command |
|---|---|
| **GET** /messages?forum=1 | **SELECT** * **FROM** messages **WHERE** forum = 1 |
| **GET** /messages?forum=1%27+UNION+ALL+SELECT+TABLE_NAME+FROM+INFORMATION_SCHEMA.TABLES-- | **SELECT** * **FROM** messages **WHERE** forum = 1 UNION ALL **SELECT** TABLE_NAME **FROM** INFORMATION_SCHEMA.TABLES |

# Injection Prevention Checklist

**Do we need it?**

**Is there a library?**

**Are there any features we should turn off or avoid?**

# Preventing Injection, Part II

**Canonicalize Input**

One representation to rule them all

**Validate Input**

Favor allowlists over blocklists

**Encode Output**

## Encodings Can Bypass a Blocklist

<         &lt;         %3C         ...

## Encodings Can Bypass a Blocklist

啀

## Encodings Can Bypass a Blocklist

## Encodings Can Bypass a Blocklist

**%56%0a**

## Encodings Can Bypass a Blocklist

**%E5%98%8A**

## Preventing Injection, Part II

**Canonicalize Input**

One representation to rule them all

**Validate Input**

Favor allowlists over blocklists

**Encode Output**

Consider where it's being rendered

# Preventing Inclusion

| | |
|---|---|
| ★★★★★ | **Don’t load resources** |
| ★★★★☆ | **Load only local resources and use an allowlist** |
| ★★★☆☆ | **Load remote resources over HTTPS and use an allowlist** |

Ensure third-party components are following the same rules

```
<m:message
   schemaLocation="…"
   xmlns:m="msg.xsd">
 <m:id>5</m:id>
 <m:text>content</m:text>
</m:message>
```

- The schema can be a hint for how to interpret or validate a payload

```
<m:message
   schemaLocation="…"
   xmlns:m="old-msg.xsd">
 <m:id>5</m:id>
 <m:text>content</m:text>
</m:message>
```

- Only use schemas that you trust

```
<m:message schemaLocation=
  “http://example.com/msg https://example.com/msg.xsd”
```

## Load Remote Resources Using HTTPS

Schemas can be loaded remotely

Attackers may use that fact to verify connectivity

If HTTP, attackers may perform a man-in-the-middle to serve a different file

# Prevent Injection and Inclusion

**To prevent injection:**

- Don't output untrusted data
- Canonicalize and validate inputs
- Encode outputs
- Use an appropriately-configured third party

**To prevent inclusion:**

- Don't use untrusted input to load resources
- Load resources locally, using an allowlist
- Load remote resources over HTTPS, using an allowlist
- Use an appropriately-configured third party

```
<?xml version="1.0"?>
<!DOCTYPE request [
 <!ENTITY lol "lol">
 <!ELEMENT lolz (#PCDATA)>
 <!ENTITY lol1 "&lol;&lol;&lol;&lol;&lol;&lol;&lol;&lol;&lol;&lol;">
 <!ENTITY lol2 "&lol1;&lol1;&lol1;&lol1;&lol1;&lol1;&lol1;&lol1;&lol1;&lol1;">
 <!ENTITY lol3 "&lol2;&lol2;&lol2;&lol2;&lol2;&lol2;&lol2;&lol2;&lol2;&lol2;">
 <!ENTITY lol4 "&lol3;&lol3;&lol3;&lol3;&lol3;&lol3;&lol3;&lol3;&lol3;&lol3;">
 <!ENTITY lol5 "&lol4;&lol4;&lol4;&lol4;&lol4;&lol4;&lol4;&lol4;&lol4;&lol4;">
 <!ENTITY lol6 "&lol5;&lol5;&lol5;&lol5;&lol5;&lol5;&lol5;&lol5;&lol5;&lol5;">
 <!ENTITY lol7 "&lol6;&lol6;&lol6;&lol6;&lol6;&lol6;&lol6;&lol6;&lol6;&lol6;">
 <!ENTITY lol8 "&lol7;&lol7;&lol7;&lol7;&lol7;&lol7;&lol7;&lol7;&lol7;&lol7;">
 <!ENTITY lol9 "&lol8;&lol8;&lol8;&lol8;&lol8;&lol8;&lol8;&lol8;&lol8;&lol8;">
 ]>
 <request>&lol9;</request>
```