black hat BRIEFINGS

DECEMBER 10-11, 2025

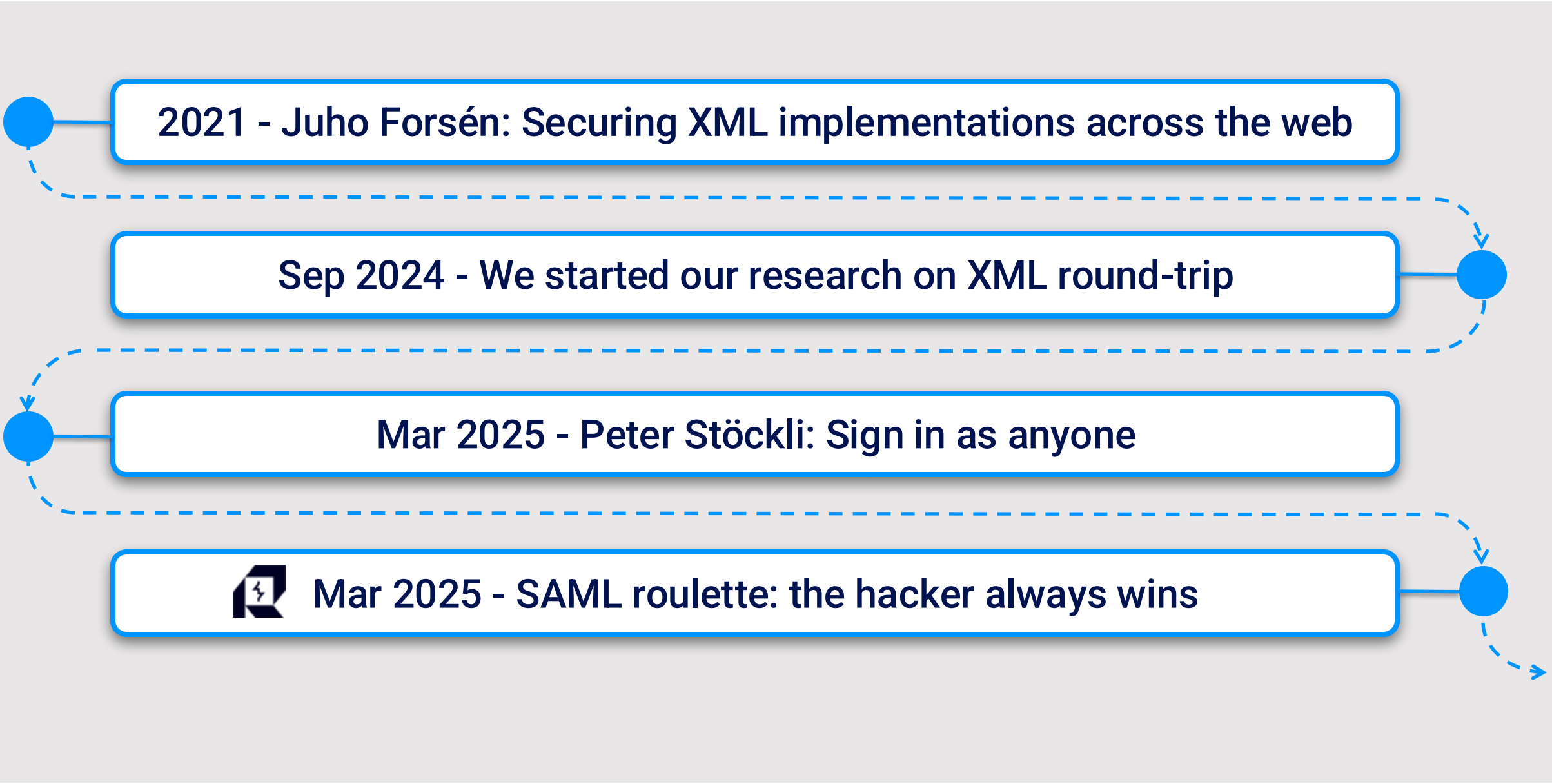
EXCEL LONDON / UNITED KINGDOM

PortSwigger Research


# Novel Bypasses for SAML Authentication

**Zak Fedotkin**

# SAML Still Matters

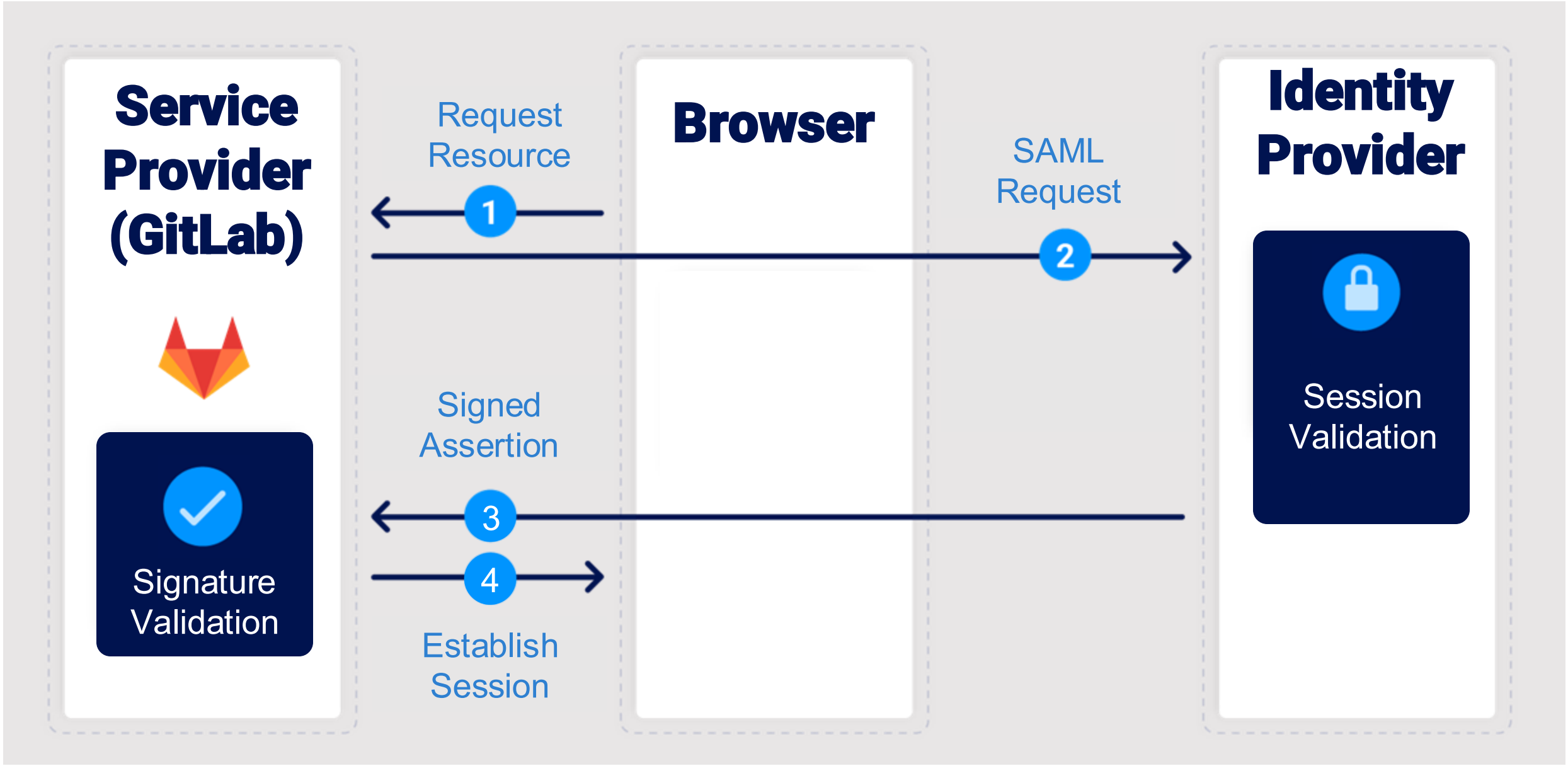


# Outline

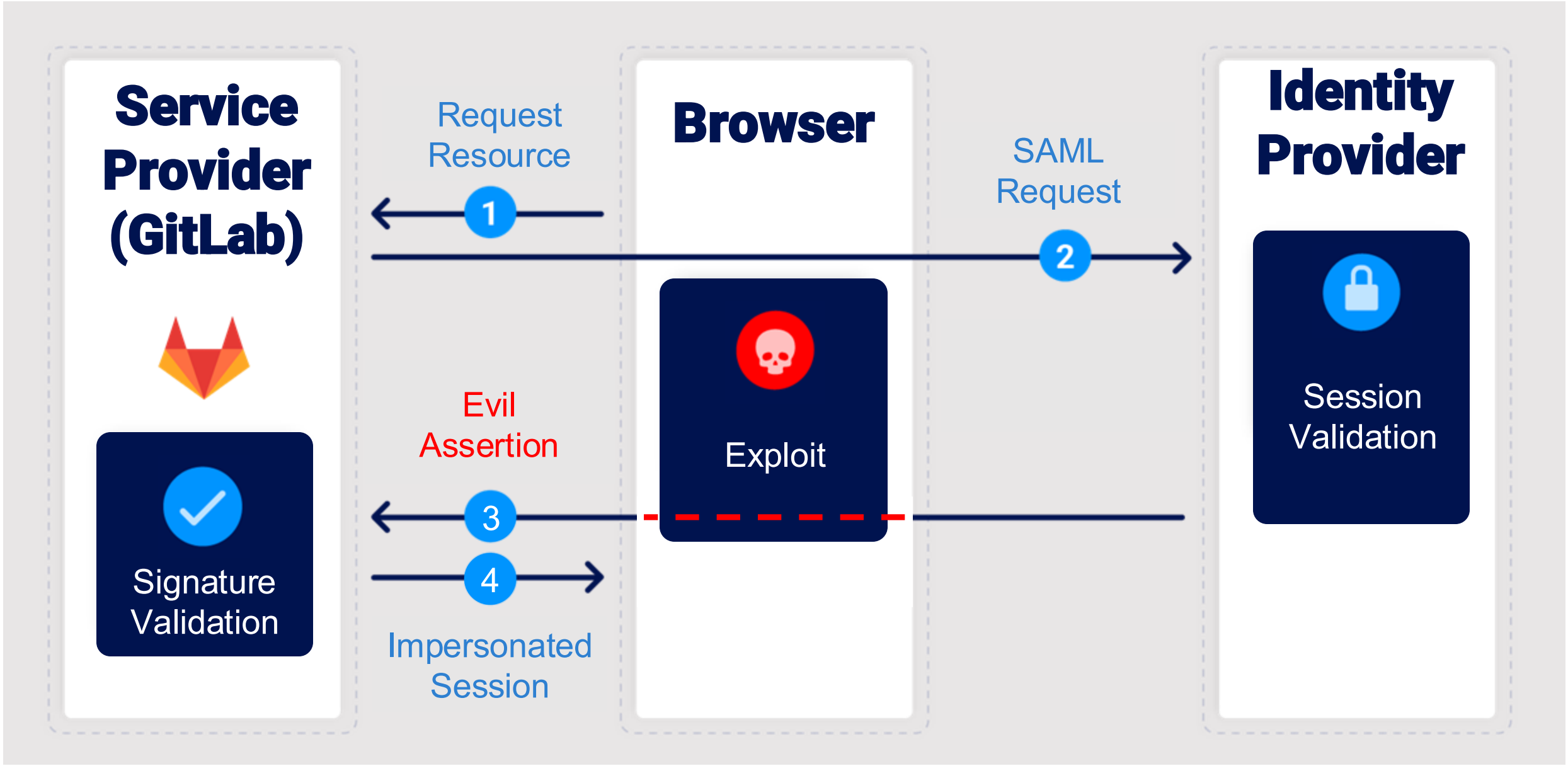
- Beyond privilege escalation: complete authentication bypass
- Bypassing defences & patches
  - New classes of privilege escalation attack
  - Chaining vulnerabilities for full compromise
- Demo & case study
- Defence / Takeaways / Questions

# Complete Authentication Bypass

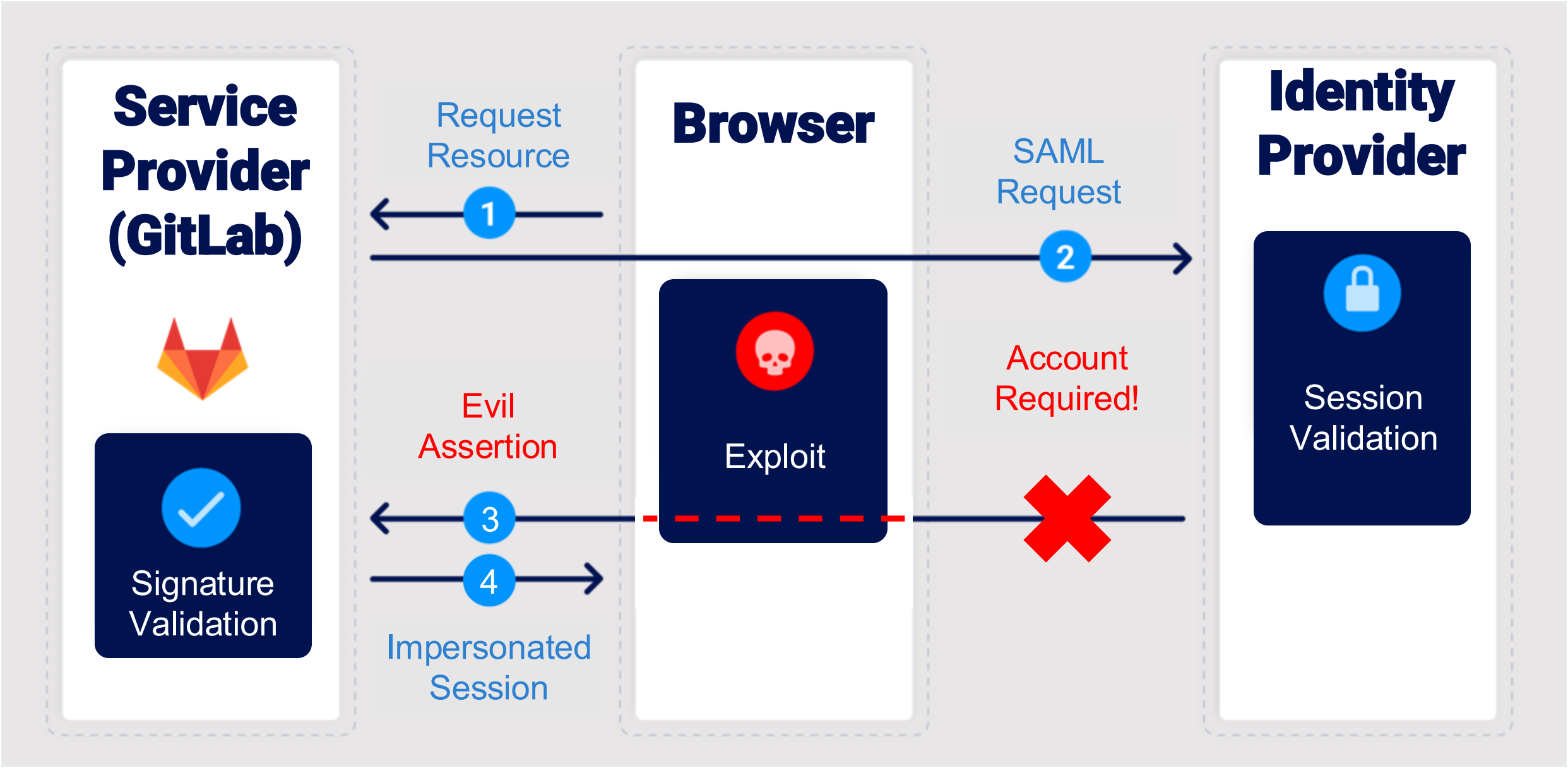
# How SAML authentication is meant to work



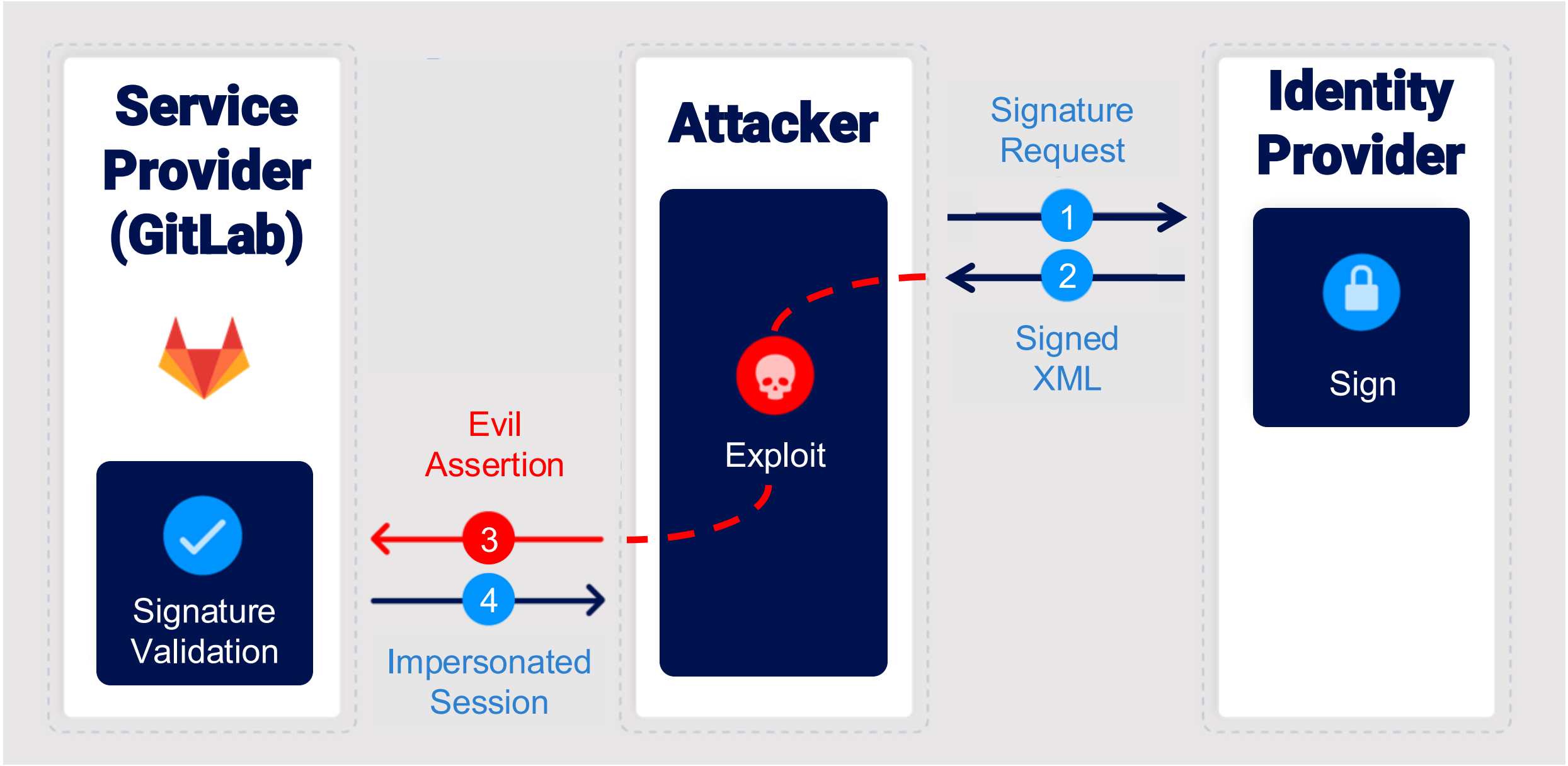
# The classic approach to Signature Wrapping



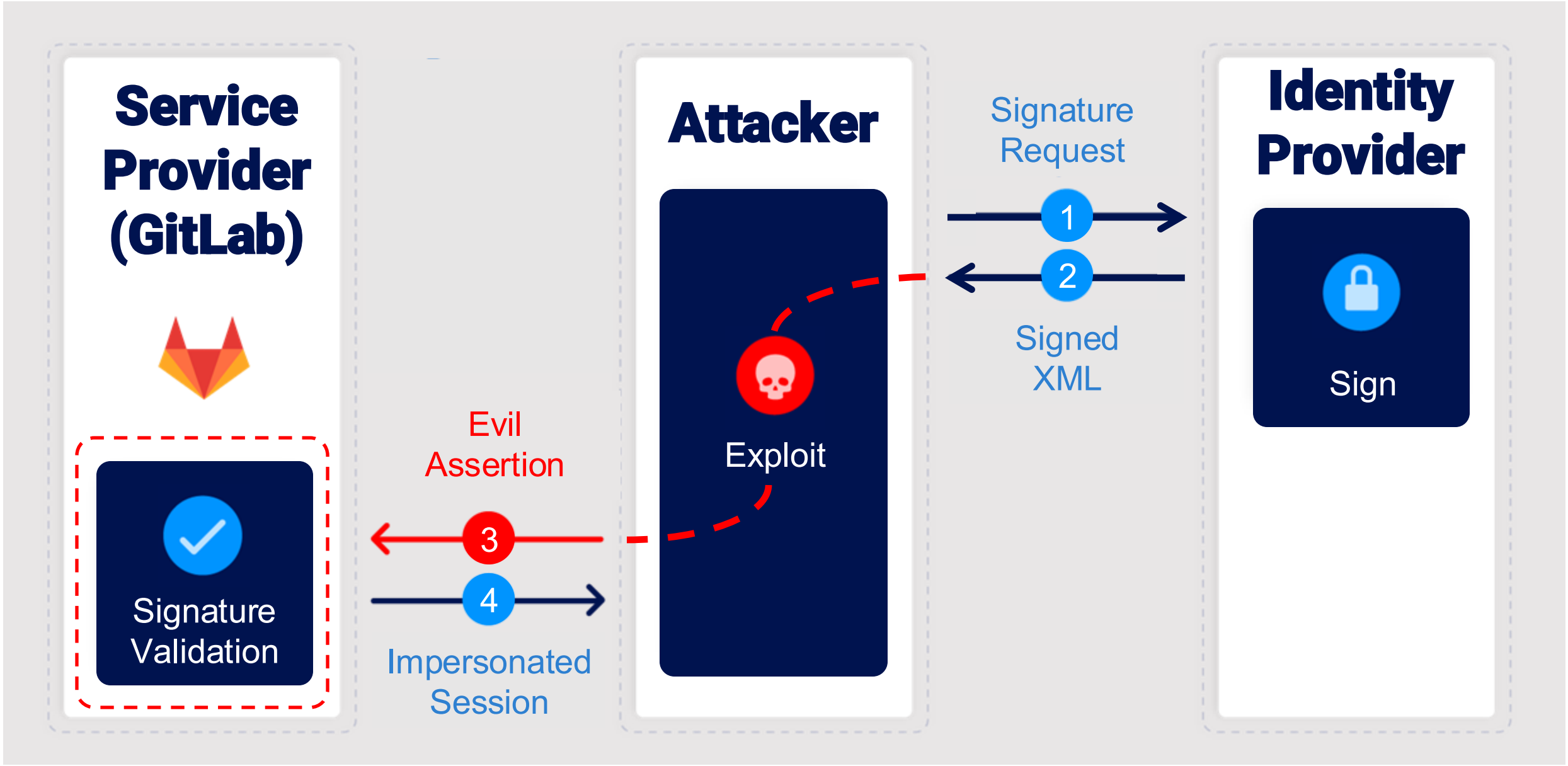
# The classic approach rarely works!



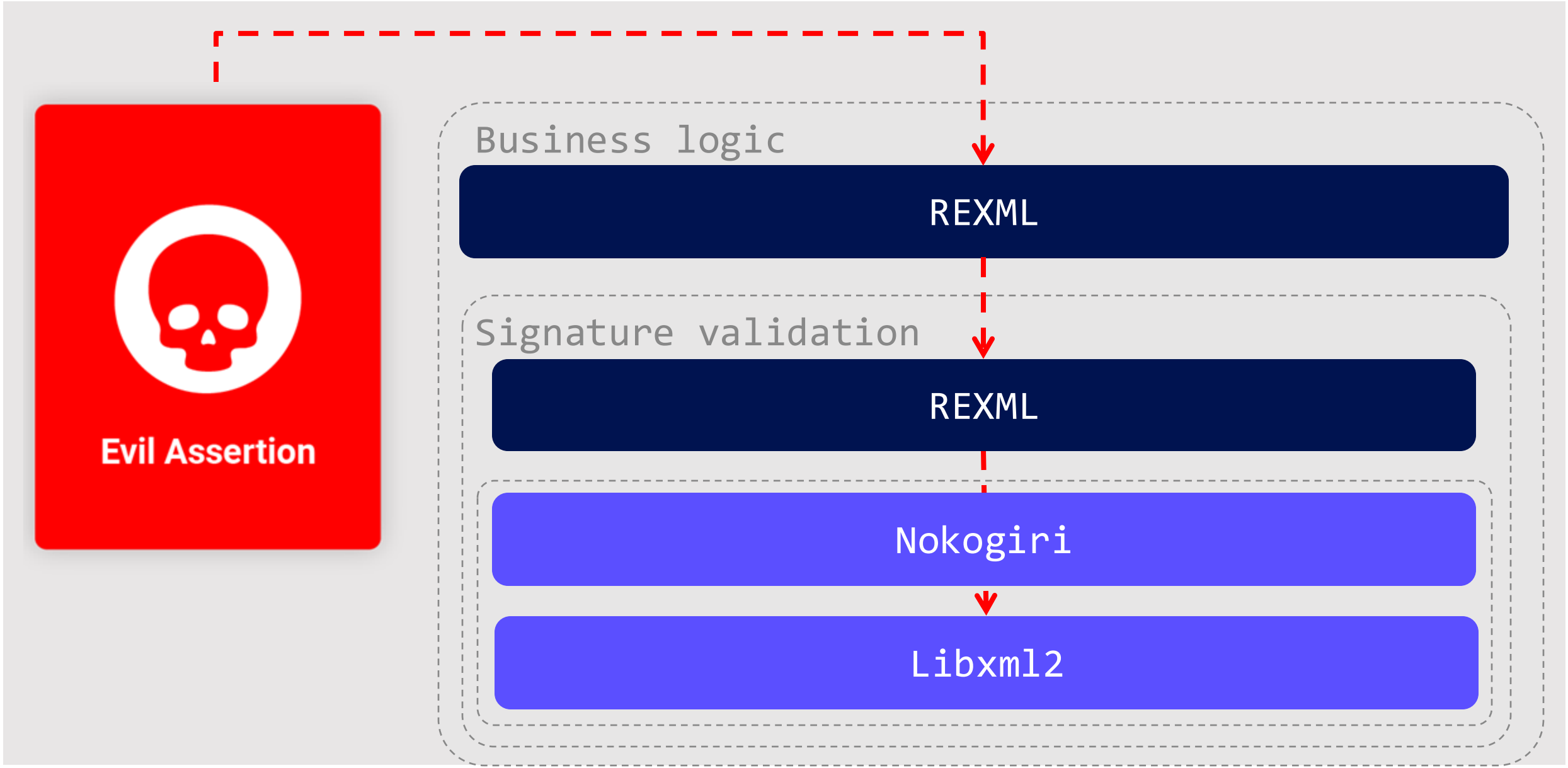
# A better approach to Signature Wrapping



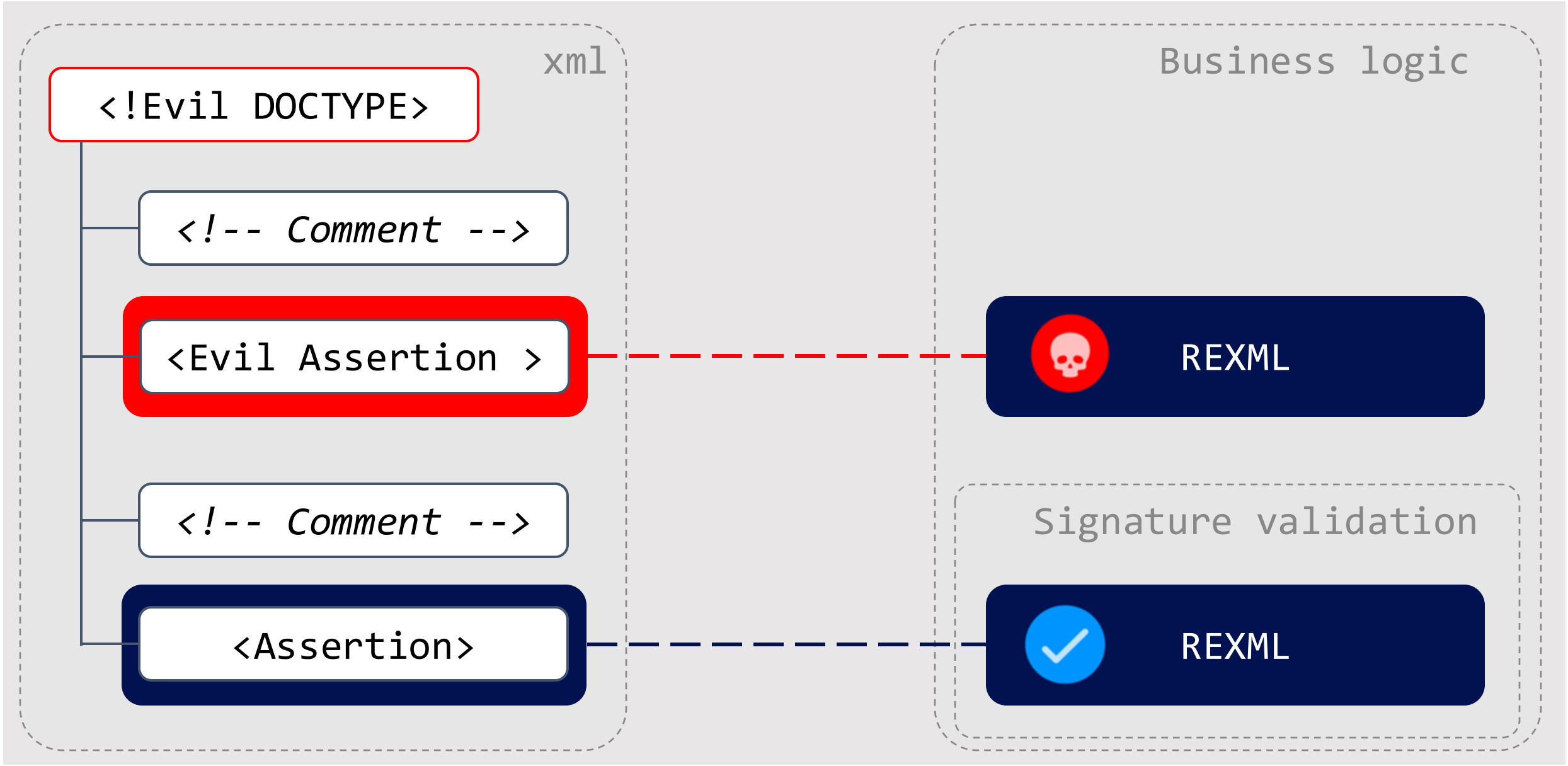
# A better approach to Signature Wrapping



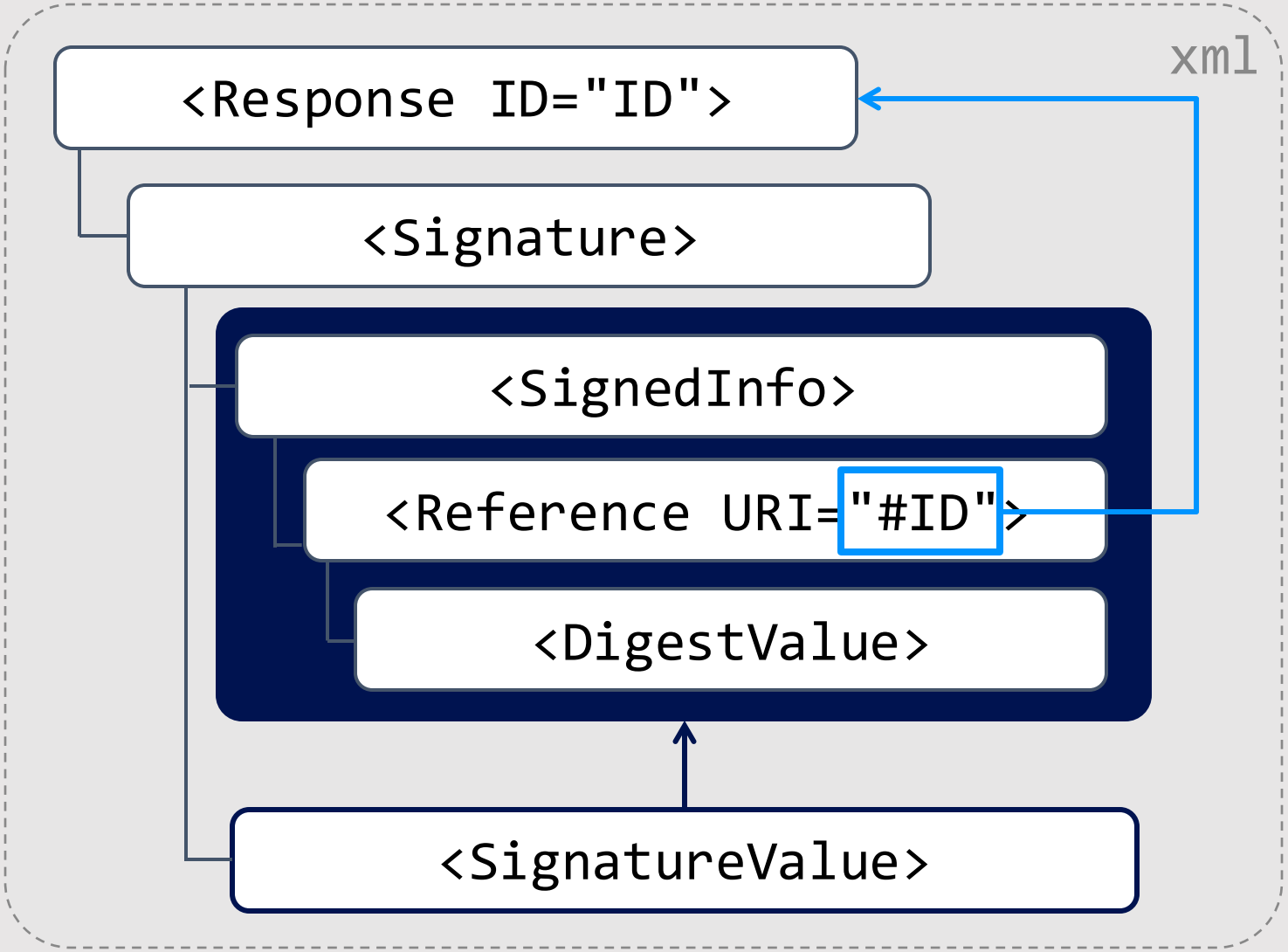
# GitLab Signature validation



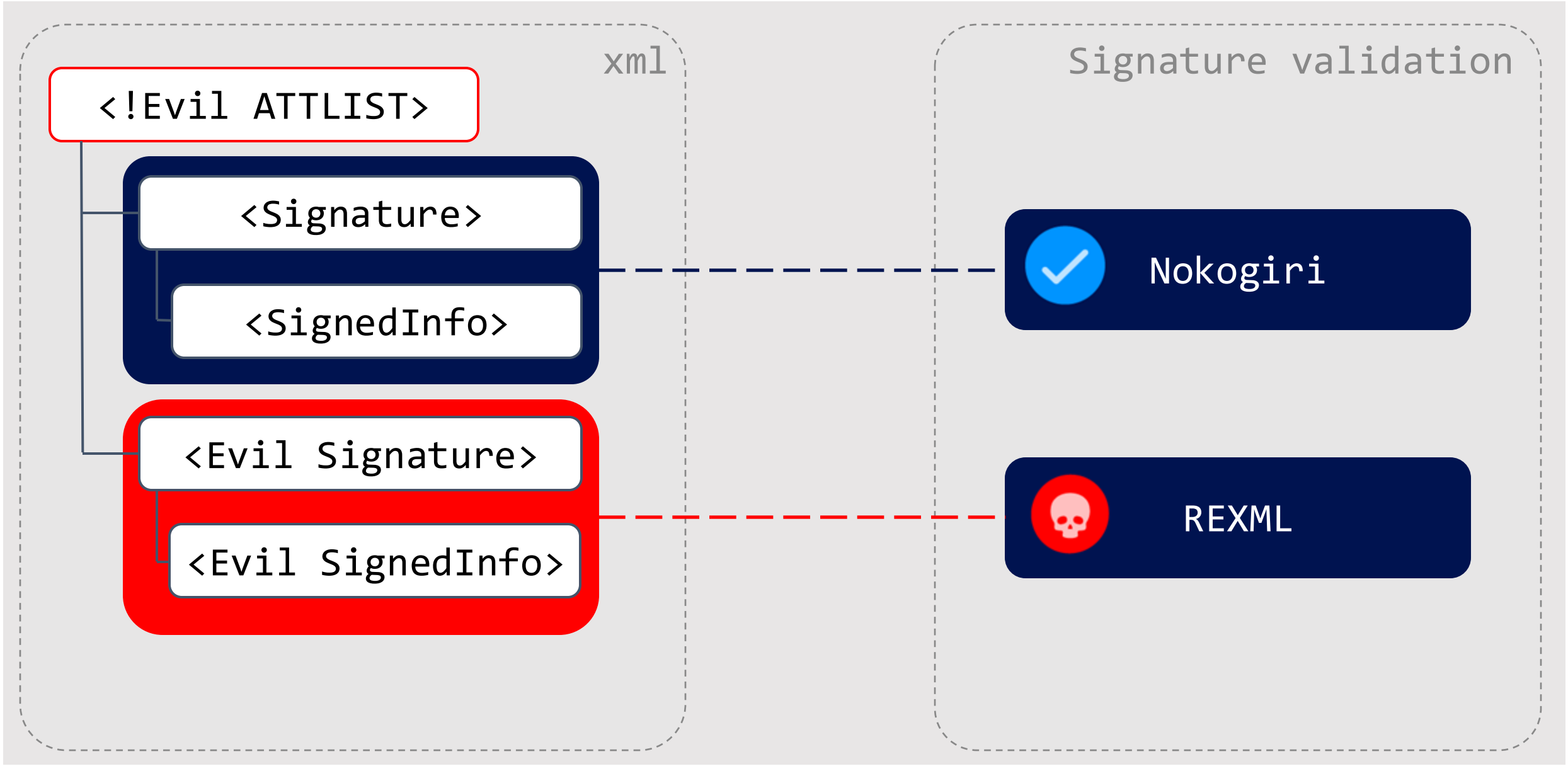
# Part 1 - Round-Trip Vulnerability



# XML Signature



# Part 2 - Namespace confusion



# Two branches, two patches:

**1.18.0**

- Changes the REXML and Nokogiri communication

**1.12.4**

- Does not change the core logic
- **Did the patch work?**

# Bypassing Patches

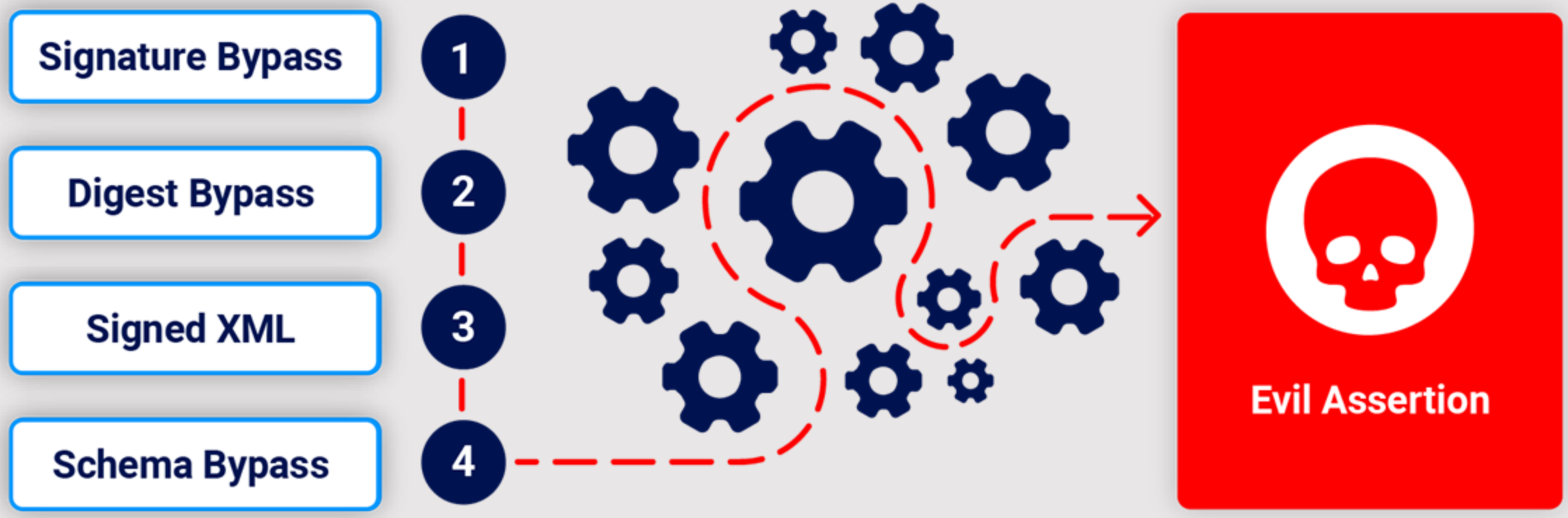
# 1.12.4 patch

1. Disables DOCTYPE declaration
2. Checks XML parsing errors

Flaws:

1. Still uses XPath query: `«//ds:Signature»`
2. Still uses two different parsers REXML and Nokogiri

# Crafting the Exploit



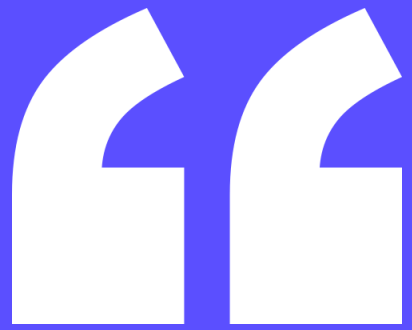
# Signature validation bypass with namespaces

```xml
<Response xmlns="urn:oasis:names:tc:SAML:2.0:protocol">
```

```xml
<saml:Response xmlns:saml="urn:oasis:names:tc:SAML:2.0:protocol">
```

XML namespaces provide a mechanism for qualifying element and attribute names by associating them with URI

# Libxml2 quirks are everywhere



This function looks in DTD attribute declarations for #FIXED or default declaration values. NOTE: This function is **ignores namespaces**. Use xmlGetNsProp or xmlGetNoNsProp for namespace aware processing.

*xmlGetProp()*

# Attribute pollution at Nokogiri (Libxml2)

xml

```
<saml:Response ID="1" saml:ID="2">
```

xml

```
<saml:Response saml:ID="2" ID="1">
```

**Nokogiri:** `element.attribute('ID')`

# Attribute pollution at PHP DOM (Libxml2)

xml

```
<saml:Response ID="1" saml:ID="2">
```

xml

```
<saml:Response saml:ID="2" ID="1">
```

PHP DOM: `element->attributes->getNamedItem('ID')->nodeValue`

# Attribute pollution at REXML is different

xml

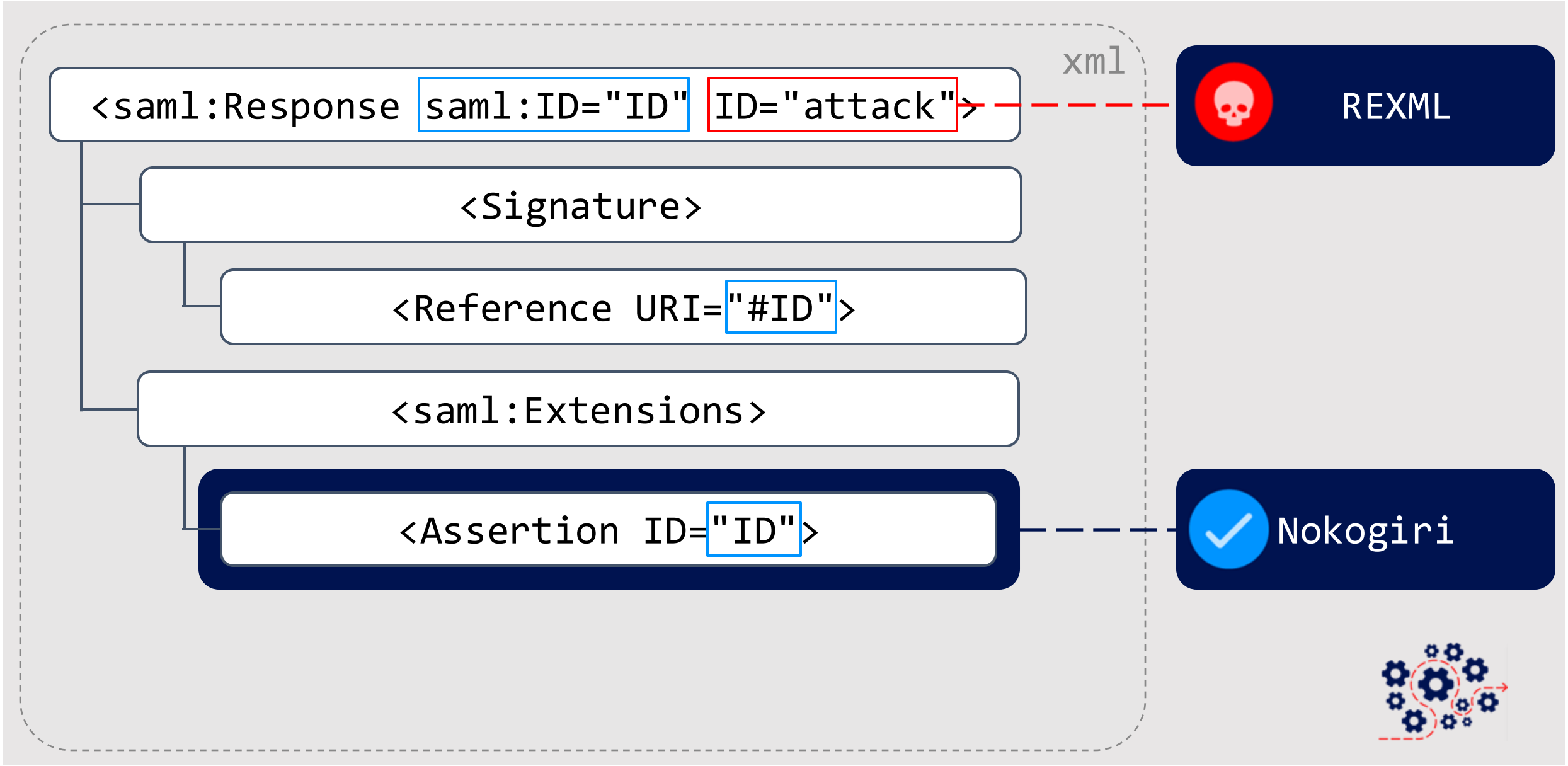
```
<Response ID="1" saml:ID="2">
```

xml

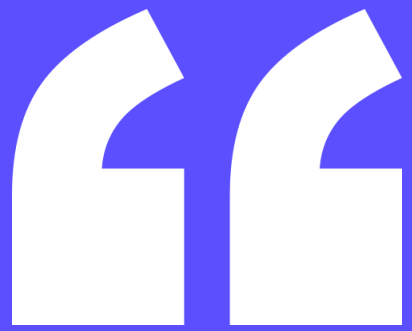
```
<saml:Response ID="1" saml:ID="2">
```

REXML: `attributes['ID']`

# Signature Wrapping with attribute pollution



# Standards vs reality

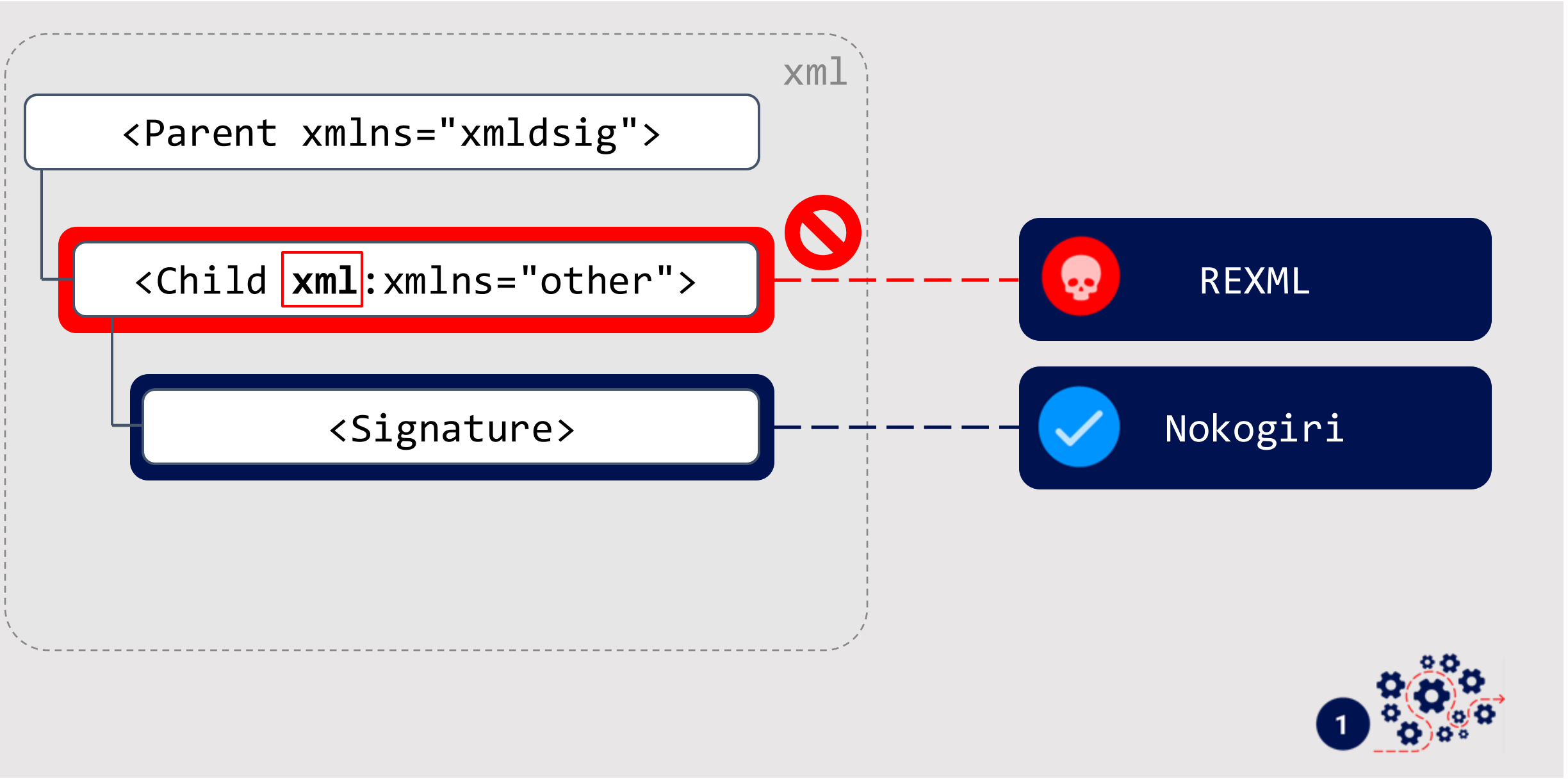


> The prefix **xml** MUST NOT be bound to any other namespace
>
> The prefix **xmlns** MUST NOT be declared

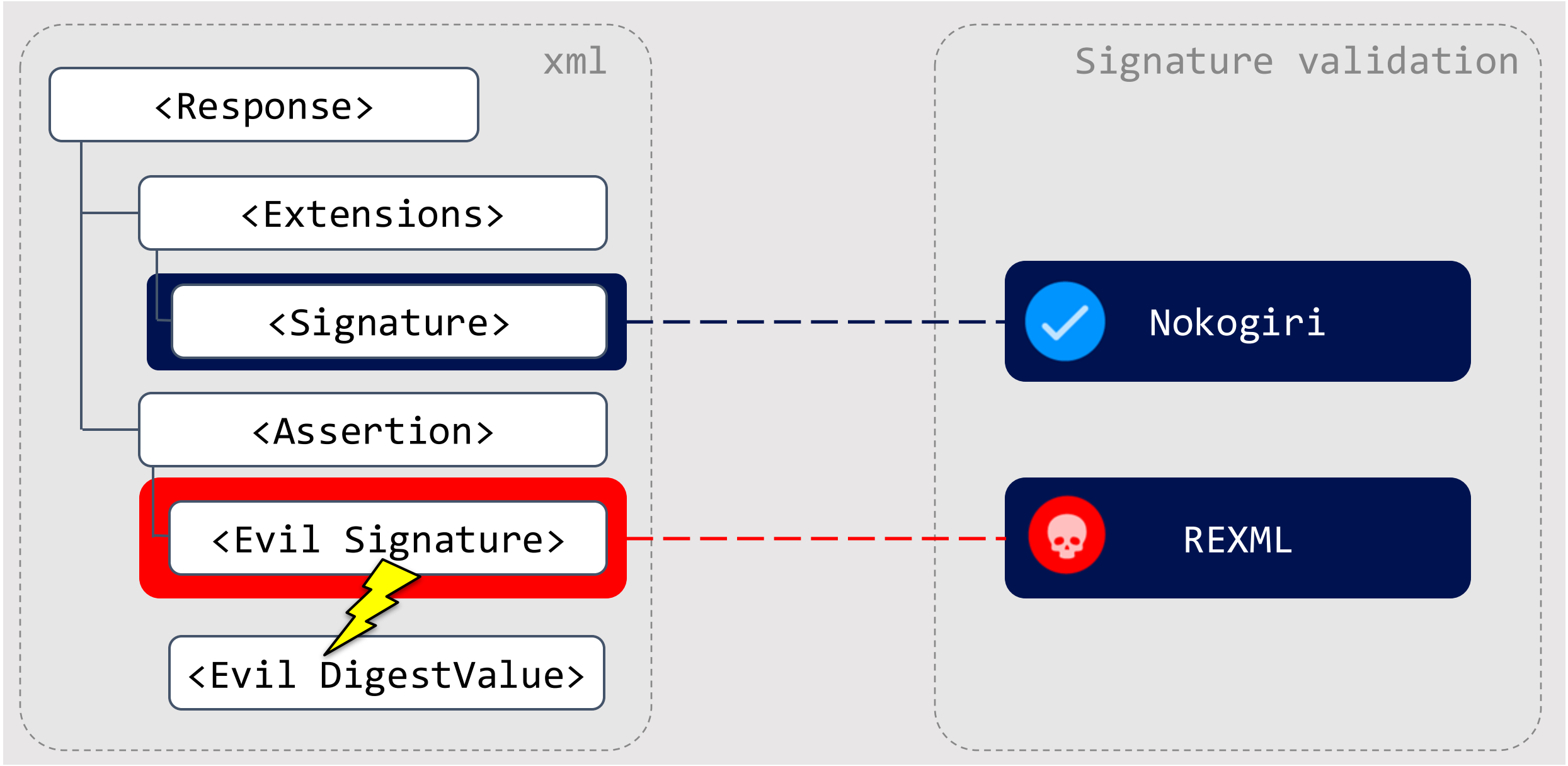
*https://www.w3.org/TR/REC-xml-names*

**REXML didn't follow it strictly!**

# REXML Namespace confusion without DTDs

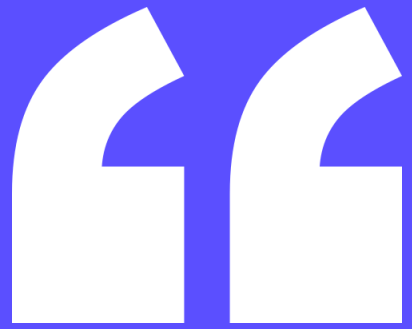


# Impossible Hash Collision?



# The Void

# Exploiting relative URI canonicalization



> The relative URIs will not be operational in the canonical form. The processing SHOULD create a new document in which relative URIs have been converted to absolute URIs, thereby mitigating any security risk for the new document.

*https://www.w3.org/TR/2001/REC-xml-c14 n-20010315#Limitations*

# Exploiting relative URI canonicalization

Libmxl2's xmlC14NExecute() returns a negative value on failure

xml_document.c

```
xmlC14NExecute(c_doc, … ,c_obuf);
…
return;
```

**But Nokogiri ignores it!**

# Exploiting relative URI canonicalization

## PHP returns false

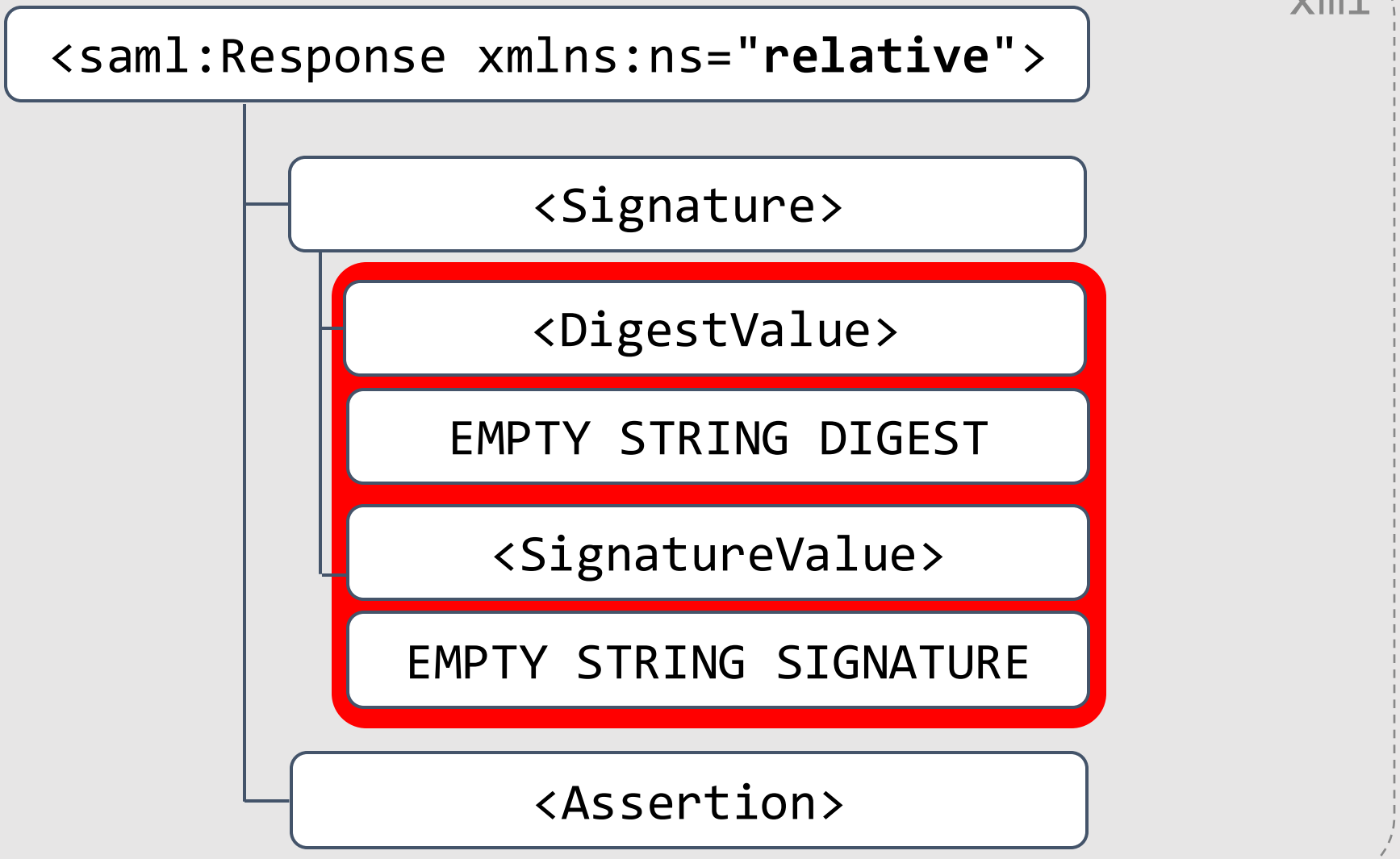
node.c

```
ret = xmlC14NExecute(docp, … , buf);
…
if (buf == NULL || ret < 0) {
            RETVAL_FALSE;
}
```

**But all PHP signature validation libraries ignore it!**

# Privileges escalation with signed empty string

xml

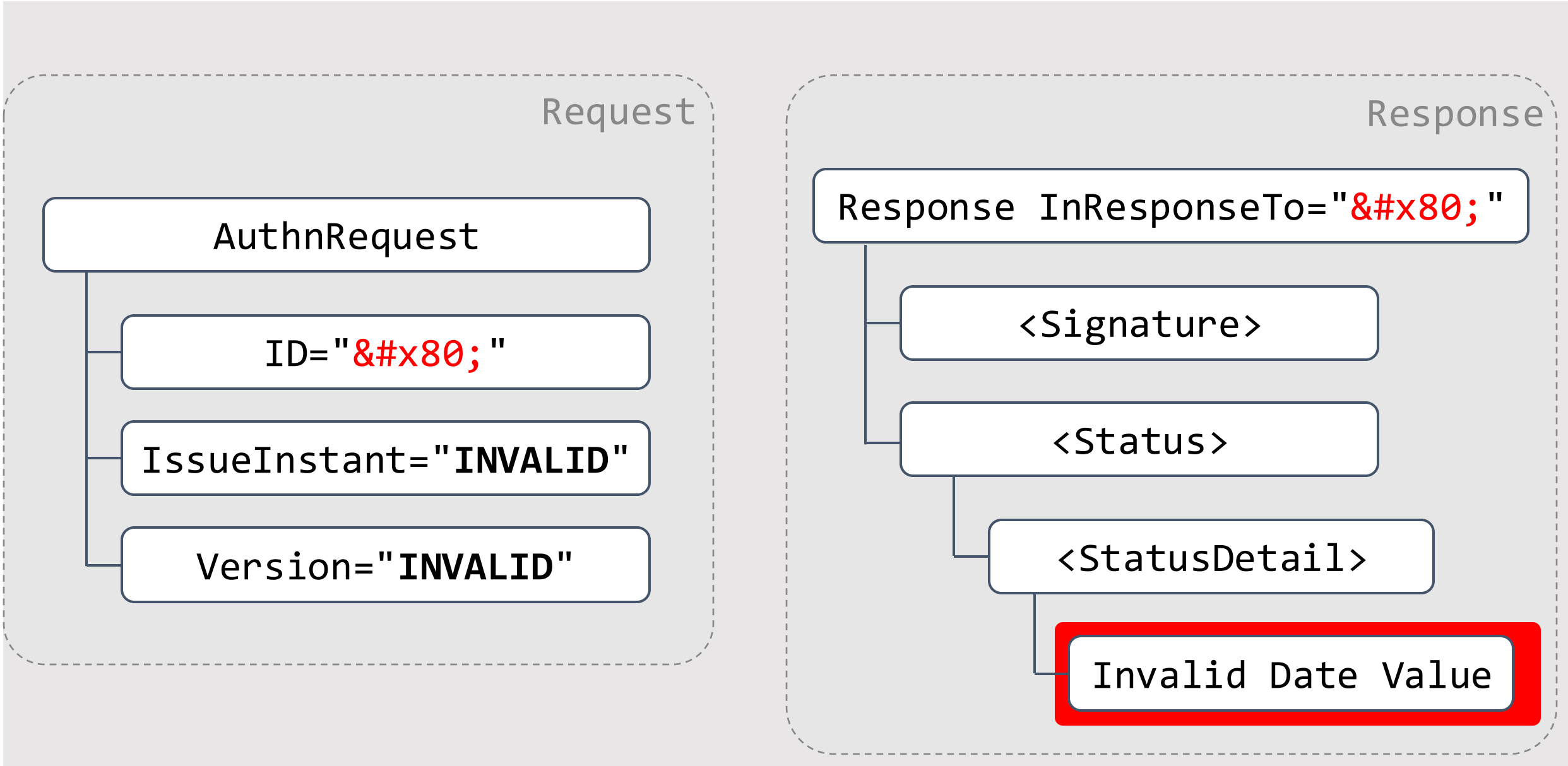


# Vulnerable libraries

| Library | Ruby xml_security | PHP xmlseclibs | C XMLSec | Java Shibboleth |
|---|---|---|---|---|
| Result |  |  | | |

# Final Exploit

# Getting a Valid Signature for the Attack



# Getting a Web Services Federation metadata XML

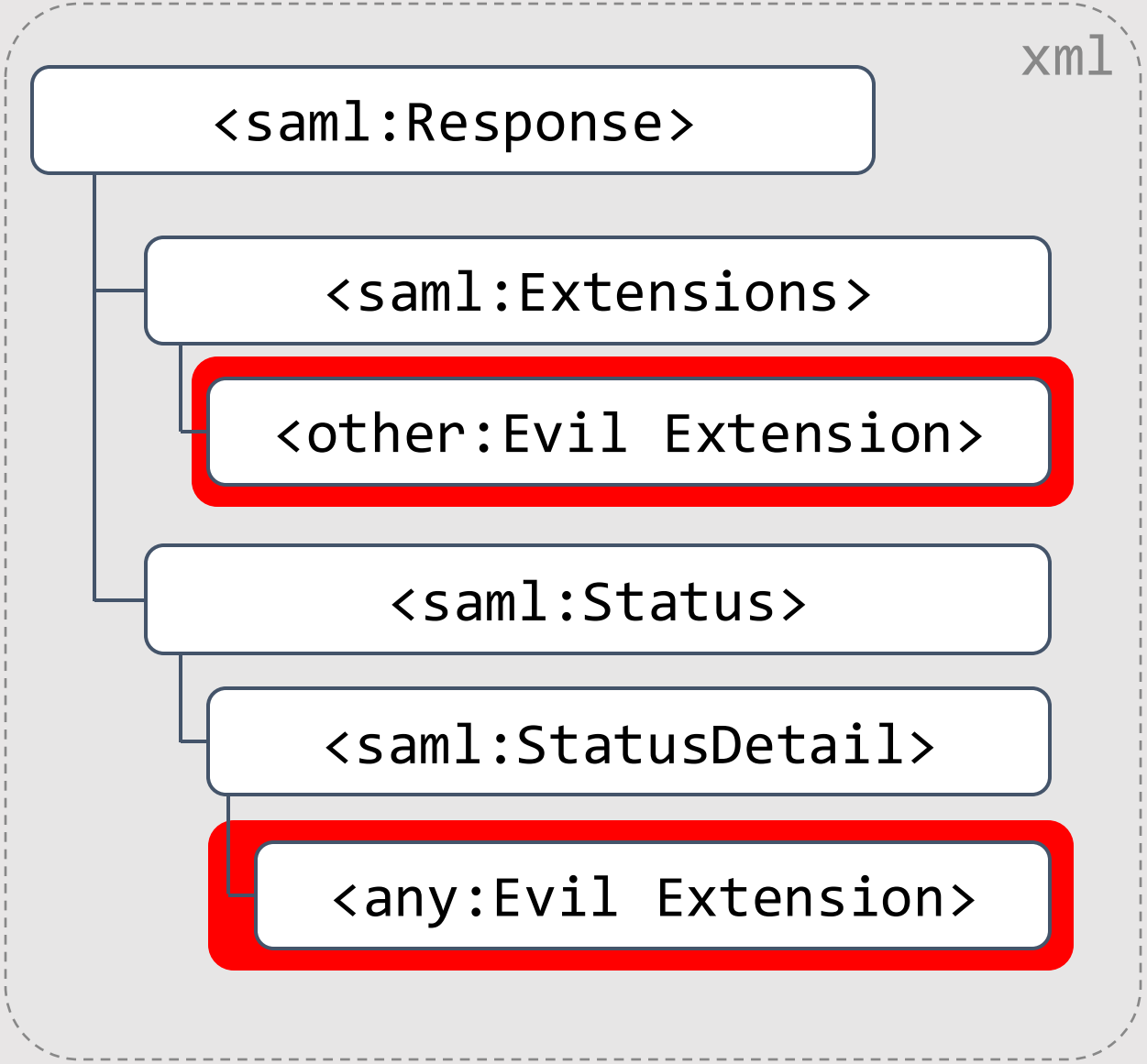
Request

```
GET /<appid>/FederationMetadata.xml
Host: idp.example.com
```

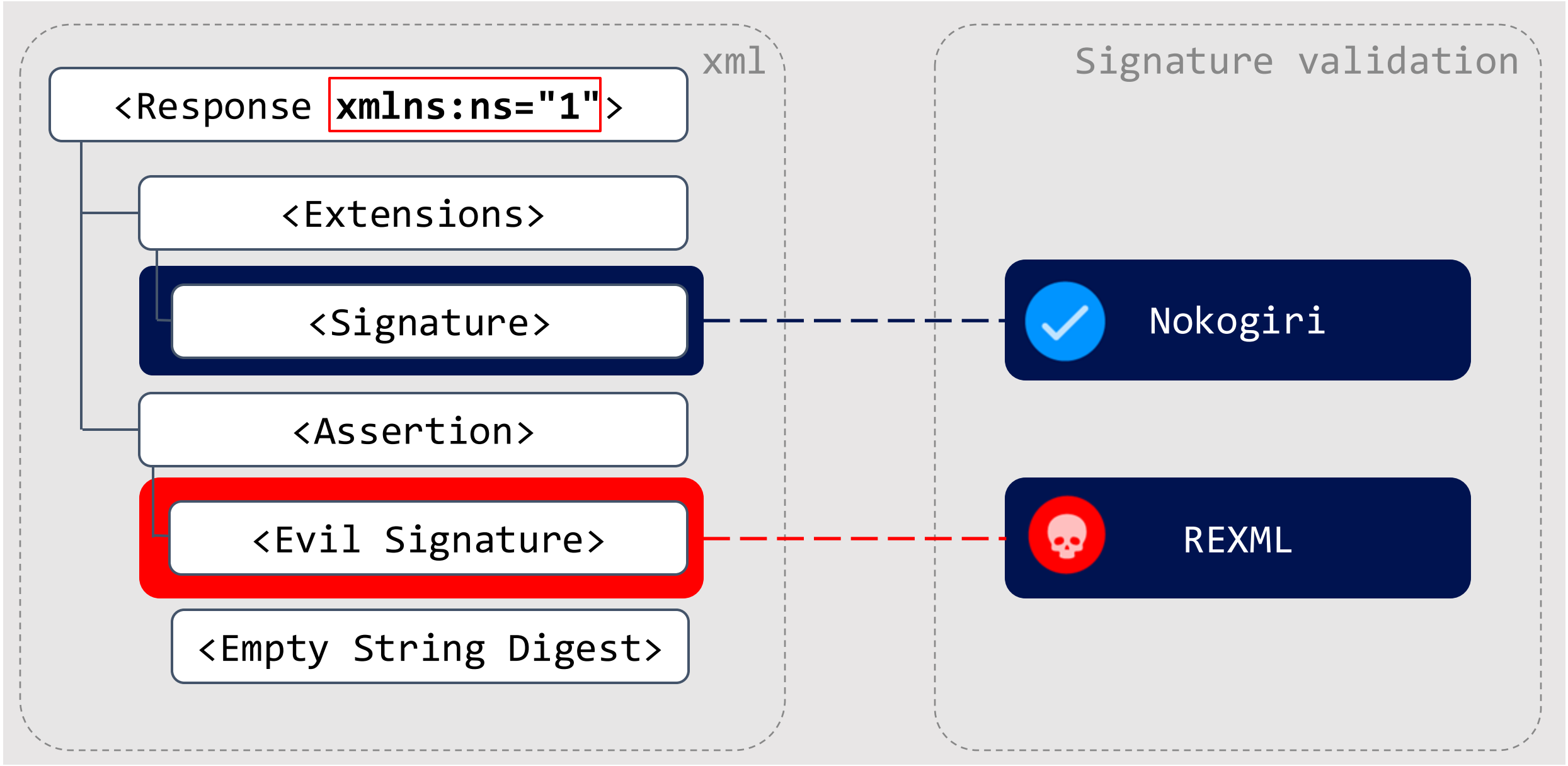
Response

```
<EntityDescriptor>
  <Signature/>
</EntityDescriptor>
```

# Schema bypass



# Complete authentication bypass!



# Demo

# Case study

# Defence

- Keep all SAML and XML security libraries up to date:
  - Ruby-SAML: update to 1.18.1
  - PHP: Rob Richards xmlseclibs update to 3.1.4
- Maybe it’s time to let SAML 2 go

# Takeaways

- SAML’s complexity is also its weakness
- Ensuring its security requires a coordinated effort from the entire open source community
- Comprehensive fixes require significant restructuring of SAML libraries

PortSwigger Research

# Q&A

Gareth Heyes
@garethheyes

Zak Fedotkin
@zakfedotkin