# Threat Modeling Report

Created on 11/9/2016 7:58:38 PM

Threat Model Name:

Owner:

Reviewer:

Contributors:

Description:

Assumptions:

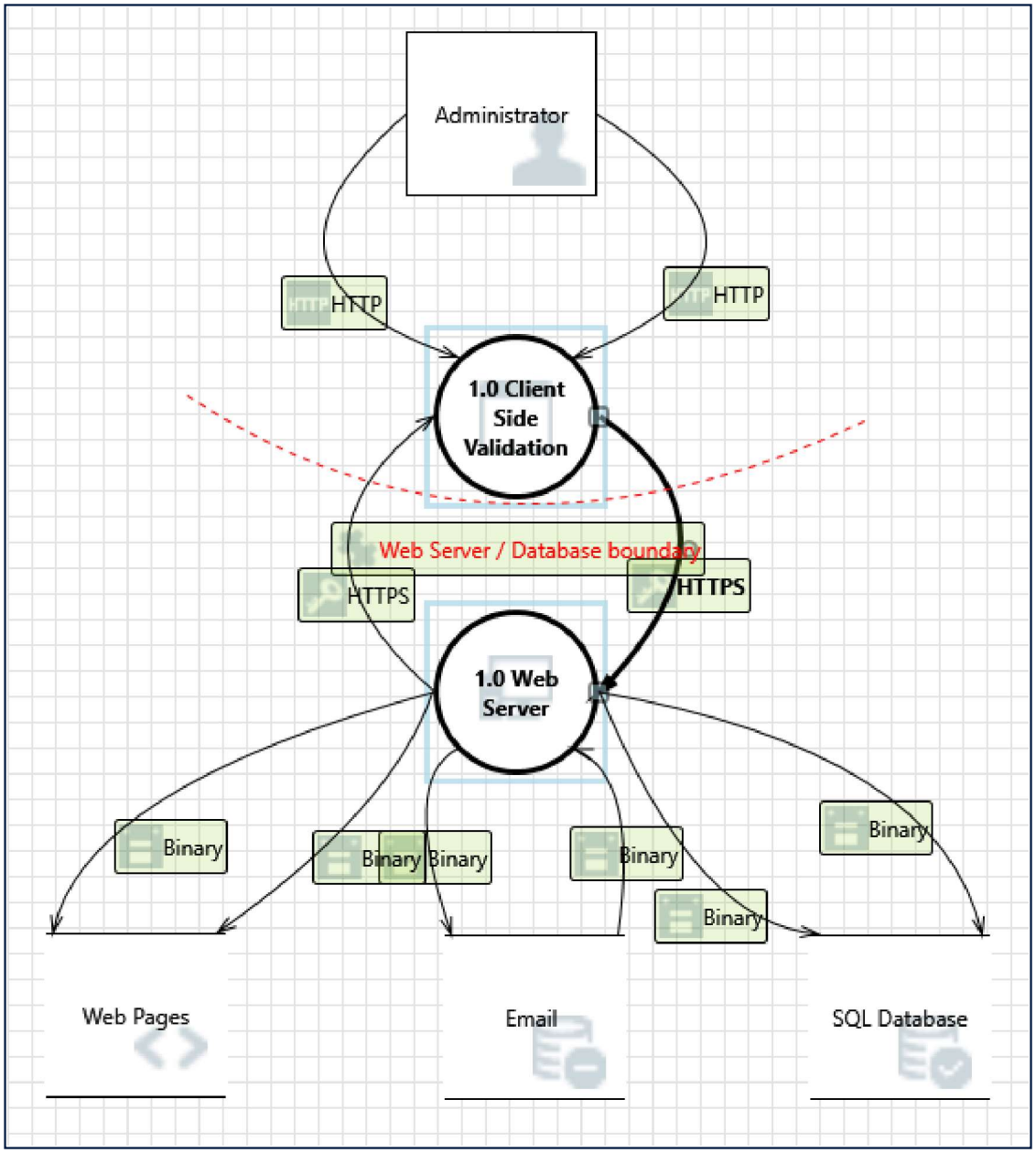
External Dependencies:

## Threat Model Summary:

| | |
|---|---|
| Not Started | 0 |
| Not Applicable | 1 |
| Needs Investigation | 20 |
| Mitigation Implemented | 14 |
| Total | 35 |
| Total Migrated | 0 |

---

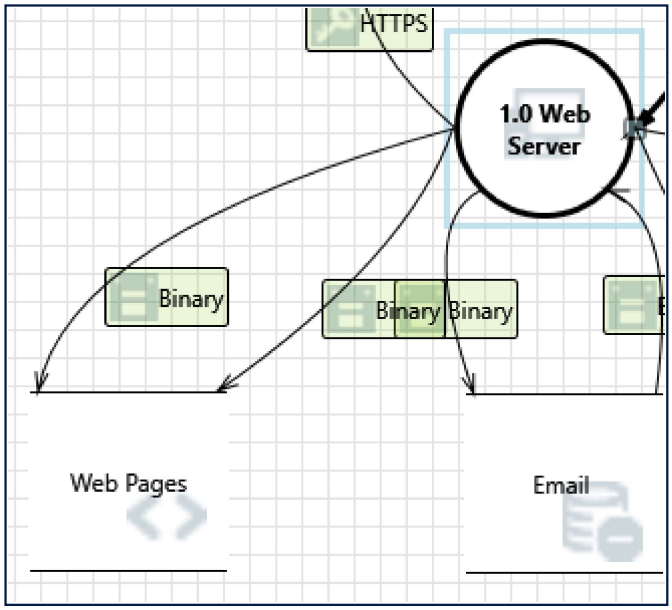
# Diagram: Diagram 1




## Diagram 1 Diagram Summary:

| | |
|---|---|
| Not Started | 0 |
| Not Applicable | 1 |
| Needs Investigation | 20 |
| Mitigation Implemented | 14 |
| Total | 35 |
| Total Migrated | 0 |

## Interaction: Binary




1. Spoofing of Destination Data Store Web Pages [State: Mitigation Implemented] [Priority: High]

**Category:** Spoofing

**Description:** Web Pages may be spoofed by an attacker and this may lead to data being written to the attacker's target instead of Web Pages. Consider using a standard authentication mechanism to identify the destination data store.

**Justification:** Wagtail includes Django's XSS-Projection to prevent remote code execution.

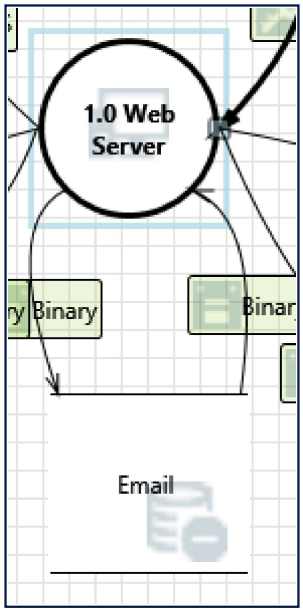
2. Potential Excessive Resource Consumption for 1.0 Web Server or Web Pages [State: Needs Investigation] [Priority: High]

**Category:** Denial Of Service

**Description:** Does 1.0 Web Server or Web Pages take explicit steps to control resource consumption? Resource consumption attacks can be hard to deal with, and there are times that it makes sense to let the OS do the job. Be careful that your resource requests don't deadlock, and that they do timeout.

**Justification:** <no mitigation provided>

## Interaction: Binary




### 3. Spoofing of Source Data Store Email [State: Needs Investigation] [Priority: High]

**Category:** Spoofing

**Description:** Email may be spoofed by an attacker and this may lead to incorrect data delivered to 1.0 Web Server. Consider using a standard authentication mechanism to identify the source data store.

**Justification:** <no mitigation provided>

### 4. Cross Site Scripting [State: Mitigation Implemented] [Priority: High]

**Category:** Tampering

**Description:** The web server '1.0 Web Server' could be a subject to a cross-site scripting attack because it does not sanitize untrusted input.

**Justification:** Wagtail includes Django's XSS-Projection and also performs input sanitization.

### 5. Persistent Cross Site Scripting [State: Mitigation Implemented] [Priority: High]

**Category:** Tampering

**Description:** The web server '1.0 Web Server' could be a subject to a persistent cross-site scripting attack because it does not sanitize data store 'Email ' inputs and output.

**Justification:** Wagtail includes Django's XSS-Projection and also performs input sanitization.

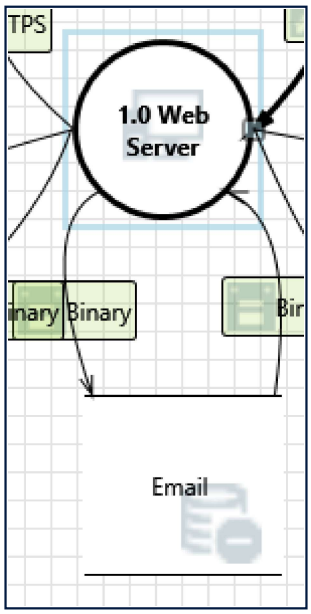
### 6. Weak Access Control for a Resource [State: Needs Investigation] [Priority: High]

**Category:** Information Disclosure

**Description:** Improper data protection of Email can allow an attacker to read information not intended for disclosure. Review authorization settings.

**Justification:** <no mitigation provided>

## Interaction: Binary




### 7. Spoofing of Destination Data Store Email [State: Needs Investigation] [Priority: High]

**Category:** Spoofing

**Description:** Email may be spoofed by an attacker and this may lead to data being written to the attacker's target instead of Email . Consider using a standard authentication mechanism to identify the destination data store.

**Justification:** <no mitigation provided>

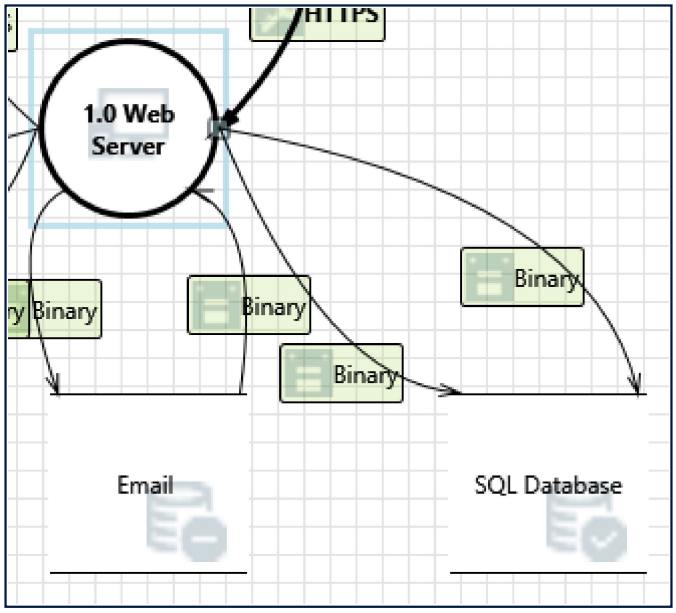
### 8. Potential Excessive Resource Consumption for 1.0 Web Server or Email [State: Needs Investigation] [Priority: High]

**Category:** Denial Of Service

**Description:** Does 1.0 Web Server or Email take explicit steps to control resource consumption? Resource consumption attacks can be hard to deal with, and there are times that it makes sense to let the OS do the job. Be careful that your resource requests don't deadlock, and that they do timeout.

**Justification:** <no mitigation provided>

## Interaction: Binary




### 9. Spoofing of Destination Data Store SQL Database [State: Mitigation Implemented] [Priority: High]

**Category:** Spoofing

**Description:** SQL Database may be spoofed by an attacker and this may lead to data being written to the attacker's target instead of SQL Database. Consider using a standard authentication mechanism to identify the destination data store.

**Justification:** Wagtail includes Django's CsrfViewMiddleware to prevent spoofing as the SQL database.

### 10. Potential SQL Injection Vulnerability for SQL Database [State: Mitigation Implemented] [Priority: High]

**Category:** Tampering

**Description:** SQL injection is an attack in which malicious code is inserted into strings that are later passed to an instance of SQL Server for parsing and execution. Any procedure that constructs SQL statements should be reviewed for injection vulnerabilities because SQL Server will execute all syntactically valid queries that it receives. Even parameterized data can be manipulated by a skilled and determined attacker.

**Justification:** Wagtail does not use unsafe SQL queries.

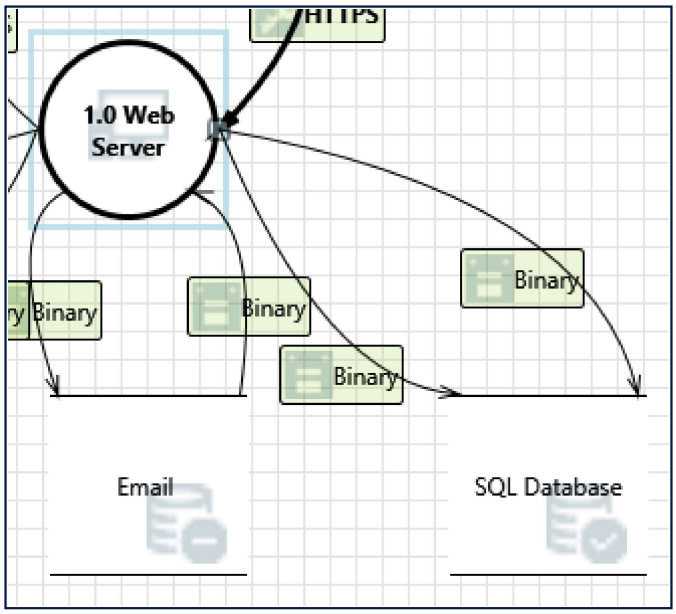
### 11. Potential Excessive Resource Consumption for 1.0 Web Server or SQL Database [State: Needs Investigation] [Priority: High]

**Category:** Denial Of Service

**Description:** Does 1.0 Web Server or SQL Database take explicit steps to control resource consumption? Resource consumption attacks can be hard to deal with, and there are times that it makes sense to let the OS do the job. Be careful that your resource requests don't deadlock, and that they do timeout.

**Justification:** <no mitigation provided>

## Interaction: Binary




### 12. Spoofing of Destination Data Store SQL Database [State: Mitigation Implemented] [Priority: High]

**Category:** Spoofing

**Description:** SQL Database may be spoofed by an attacker and this may lead to data being written to the attacker's target instead of SQL Database. Consider using a standard authentication mechanism to identify the destination data store.

**Justification:** Wagtail includes Django's CsrfViewMiddleware to prevent spoofing as the SQL database.

### 13. Potential SQL Injection Vulnerability for SQL Database [State: Mitigation Implemented] [Priority: High]

**Category:** Tampering

**Description:** SQL injection is an attack in which malicious code is inserted into strings that are later passed to an instance of SQL Server for parsing and execution. Any procedure that constructs SQL statements should be reviewed for injection vulnerabilities because SQL Server will execute all syntactically valid queries that it receives. Even parameterized data can be manipulated by a skilled and determined attacker.

**Justification:** Wagtail does not use unsafe SQL queries.

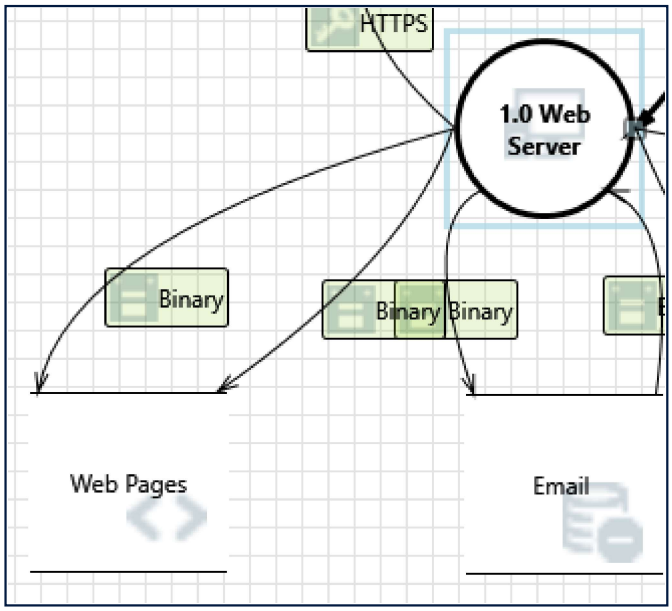
### 14. Potential Excessive Resource Consumption for 1.0 Web Server or SQL Database [State: Needs Investigation] [Priority: High]

**Category:** Denial Of Service

**Description:** Does 1.0 Web Server or SQL Database take explicit steps to control resource consumption? Resource consumption attacks can be hard to deal with, and there are times that it makes sense to let the OS do the job. Be careful that your resource requests don't deadlock, and that they do timeout.

**Justification:** <no mitigation provided>

## Interaction: Binary




### 15. Spoofing of Destination Data Store Web Pages [State: Mitigation Implemented] [Priority: High]

**Category:** Spoofing

**Description:** Web Pages may be spoofed by an attacker and this may lead to data being written to the attacker's target instead of Web Pages. Consider using a standard authentication mechanism to identify the destination data store.

**Justification:** Wagtail includes Django's XSS-Projection to prevent remote code execution.

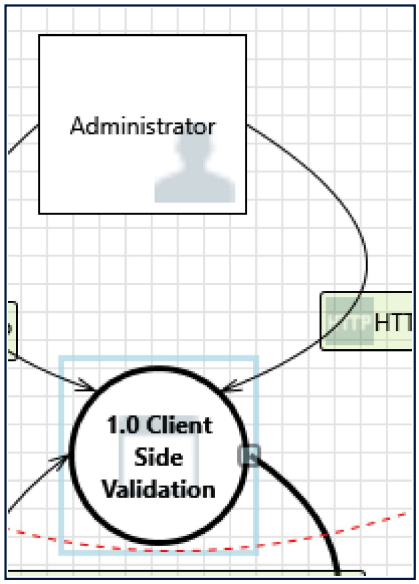
### 16. Potential Excessive Resource Consumption for 1.0 Web Server or Web Pages [State: Needs Investigation] [Priority: High]

**Category:** Denial Of Service

**Description:** Does 1.0 Web Server or Web Pages take explicit steps to control resource consumption? Resource consumption attacks can be hard to deal with, and there are times that it makes sense to let the OS do the job. Be careful that your resource requests don't deadlock, and that they do timeout.

**Justification:** <no mitigation provided>

## Interaction: HTTP




17. Spoofing the Administrator External Entity [State: Mitigation Implemented] [Priority: High]

**Category:** Spoofing

**Description:** Administrator may be spoofed by an attacker and this may lead to unauthorized access to 1.0 Client Side Validation. Consider using a standard authentication mechanism to identify the external entity.

**Justification:** Wagtail includes Django's CsrfViewMiddleware to prevent spoofing as an administrator.

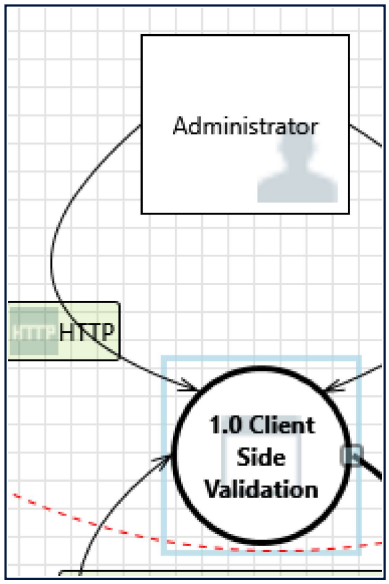
18. Elevation Using Impersonation [State: Mitigation Implemented] [Priority: High]

**Category:** Elevation Of Privilege

**Description:** 1.0 Client Side Validation may be able to impersonate the context of Administrator in order to gain additional privilege.

**Justification:** Wagtail includes Django's CsrfViewMiddleware to prevent spoofing as an administrator.

## Interaction: HTTP




19. Spoofing the Administrator External Entity [State: Needs Investigation] [Priority: High]

**Category:** Spoofing

**Description:** Administrator may be spoofed by an attacker and this may lead to unauthorized access to 1.0 Client Side Validation. Consider using a standard authentication mechanism to identify the external entity.

**Justification:** Wagtail includes Django's CsrfViewMiddleware to prevent spoofing as an administrator.

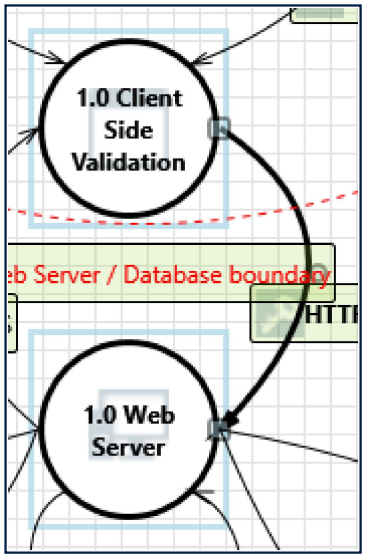
20. Elevation Using Impersonation [State: Needs Investigation] [Priority: High]

**Category:** Elevation Of Privilege

**Description:** 1.0 Client Side Validation may be able to impersonate the context of Administrator in order to gain additional privilege.

**Justification:** Wagtail includes Django's CsrfViewMiddleware to prevent spoofing as an administrator.

## Interaction: HTTPS




21. Cross Site Scripting [State: Mitigation Implemented] [Priority: High]

**Category:** Tampering

**Description:** The web server '1.0 Web Server' could be a subject to a cross-site scripting attack because it does not sanitize untrusted input.

**Justification:** Wagtail includes Django's XSS-Projection and also performs input sanitization.

22. Elevation Using Impersonation [State: Not Applicable] [Priority: High]

**Category:** Elevation Of Privilege

**Description:** 1.0 Web Server may be able to impersonate the context of 1.0 Client Side Validation in order to gain additional privilege.

**Justification:** The web server already has more privileges than the client side.

23. Spoofing the 1.0 Client Side Validation Process [State: Needs Investigation] [Priority: High]

**Category:** Spoofing

**Description:** 1.0 Client Side Validation may be spoofed by an attacker and this may lead to unauthorized access to 1.0 Web Server. Consider using a standard authentication mechanism to identify the source process.

**Justification:** Wagtail includes Django's CsrfViewMiddleware to prevent spoofing as an administrator.

24. Potential Data Repudiation by 1.0 Web Server [State: Needs Investigation] [Priority: High]

**Category:** Repudiation

**Description:** 1.0 Web Server claims that it did not receive data from a source outside the trust boundary. Consider using logging or auditing to record the source, time, and summary of the received data.

**Justification:** <no mitigation provided>

### 25. Potential Process Crash or Stop for 1.0 Web Server [State: Needs Investigation] [Priority: High]

**Category:** Denial Of Service
**Description:** 1.0 Web Server crashes, halts, stops or runs slowly; in all cases violating an availability metric.
**Justification:** <no mitigation provided>

### 26. Data Flow HTTPS Is Potentially Interrupted [State: Needs Investigation] [Priority: High]

**Category:** Denial Of Service
**Description:** An external agent interrupts data flowing across a trust boundary in either direction.
**Justification:** <no mitigation provided>

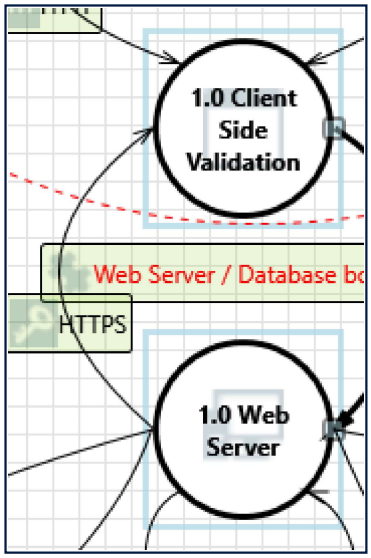
### 27. 1.0 Web Server May be Subject to Elevation of Privilege Using Remote Code Execution [State: Mitigation Implemented] [Priority: High]

**Category:** Elevation Of Privilege
**Description:** 1.0 Client Side Validation may be able to remotely execute code for 1.0 Web Server.
**Justification:** Wagtail includes Django's XSS-Projection to prevent remote code execution.

### 28. Elevation by Changing the Execution Flow in 1.0 Web Server [State: Mitigation Implemented] [Priority: High]

**Category:** Elevation Of Privilege
**Description:** An attacker may pass data into 1.0 Web Server in order to change the flow of program execution within 1.0 Web Server to the attacker's choosing.
**Justification:** Wagtail does not use unsafe SQL queries.

## Interaction: HTTPS




29. Elevation Using Impersonation [State: Needs Investigation] [Priority: High]

**Category:** Elevation Of Privilege

**Description:** 1.0 Client Side Validation may be able to impersonate the context of 1.0 Web Server in order to gain additional privilege.

**Justification:** <no mitigation provided>

30. Spoofing the 1.0 Web Server Process [State: Mitigation Implemented] [Priority: High]

**Category:** Spoofing

**Description:** 1.0 Web Server may be spoofed by an attacker and this may lead to unauthorized access to 1.0 Client Side Validation. Consider using a standard authentication mechanism to identify the source process.

**Justification:** Wagtail includes Django's CsrfViewMiddleware to prevent spoofing as an administrator.

31. Potential Data Repudiation by 1.0 Client Side Validation [State: Needs Investigation] [Priority: High]

**Category:** Repudiation

**Description:** 1.0 Client Side Validation claims that it did not receive data from a source outside the trust boundary. Consider using logging or auditing to record the source, time, and summary of the received data.

**Justification:** <no mitigation provided>

32. Potential Process Crash or Stop for 1.0 Client Side Validation [State: Needs Investigation] [Priority: High]

**Category:** Denial Of Service

**Description:** 1.0 Client Side Validation crashes, halts, stops or runs slowly; in all cases violating an availability metric.

Justification: <no mitigation provided>

### 33. Data Flow HTTPS Is Potentially Interrupted [State: Needs Investigation] [Priority: High]

Category: Denial Of Service
Description: An external agent interrupts data flowing across a trust boundary in either direction.
Justification: <no mitigation provided>

### 34. 1.0 Client Side Validation May be Subject to Elevation of Privilege Using Remote Code Execution [State: Needs Investigation] [Priority: High]

Category: Elevation Of Privilege
Description: 1.0 Web Server may be able to remotely execute code for 1.0 Client Side Validation.
Justification: Wagtail includes Django's XSS-Projection to prevent remote code execution.

### 35. Elevation by Changing the Execution Flow in 1.0 Client Side Validation [State: Needs Investigation] [Priority: High]

Category: Elevation Of Privilege
Description: An attacker may pass data into 1.0 Client Side Validation in order to change the flow of program execution within 1.0 Client Side Validation to the attacker's choosing.
Justification: <no mitigation provided>