Threat Modeling Report

Created on 11/9/2016 10:42:04 PM

Threat Model Name: Wagtail Threat Model

Owner: Team Node

Reviewer: David Nelson

Contributors: Tyler Chrisman, Sean Hoefler, Juan Membreno, David Nelson

Description:

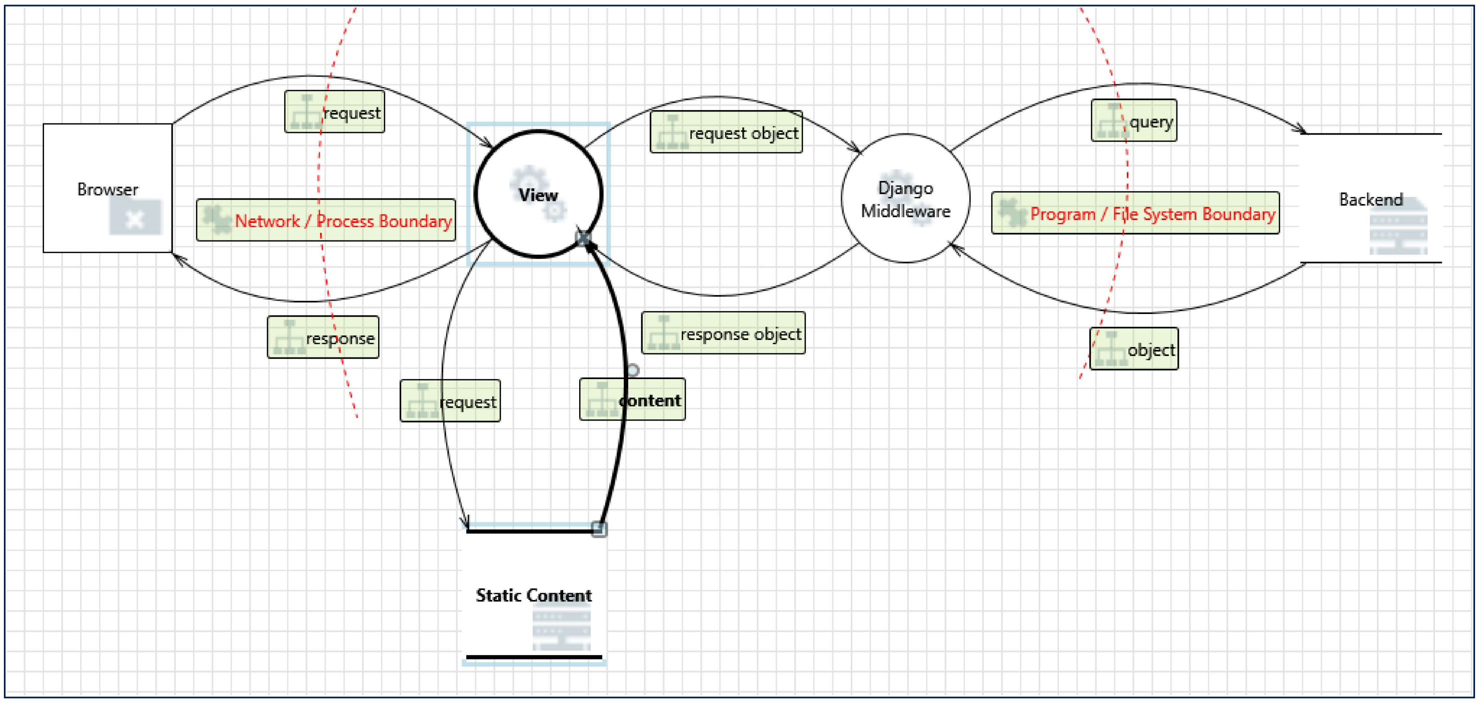
Assumptions:

External Dependencies:

Threat Model Summary:

Not Started	0
Not Applicable	5
Needs Investigation	25
Mitigation Implemented	7
Total	37
Total Migrated	0

Diagram: Wagtail Threat Model

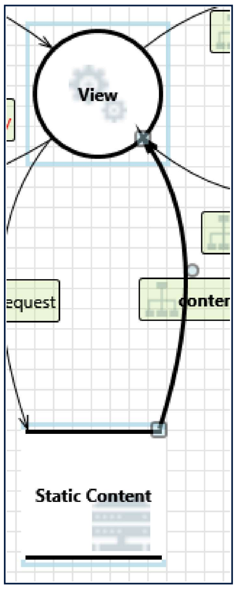


Wagtail Threat Model Diagram Summary:

Not Started	0
Not Applicable	5
Needs Investigation	25

Mitigation Implemented	7
Total	37
Total Migrated	0

Interaction: content



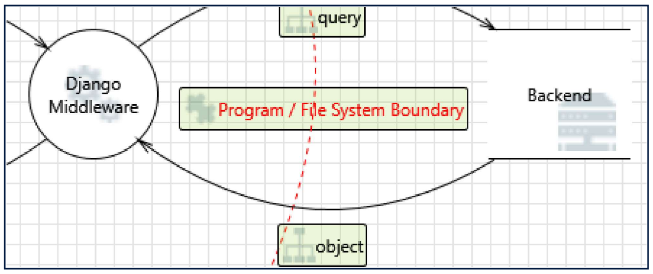
1. Spoofing of Source Data Store Static Content [State: Needs Investigation] [Priority: High]

Category: Spoofing
Description: Static Content may be spoofed by an attacker and this may lead to incorrect data delivered to View. Consider using a standard authentication mechanism to identify the source data store.
Justification: <no mitigation provided>

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

Category: Information Disclosure
Description: Improper data protection of Static Content can allow an attacker to read information not intended for disclosure. Review authorization settings.
Justification: <no mitigation provided>

Interaction: object



3. Elevation by Changing the Execution Flow in Django Middleware [State: Needs Investigation] [Priority: High]

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

4. Django Middleware May be Subject to Elevation of Privilege Using Remote Code Execution [State: Not Applicable] [Priority: High]

Category: Elevation Of Privilege
Description: Backend may be able to remotely execute code for Django Middleware.
Justification: The backend is a data store.

5. Data Store Inaccessible [State: Needs Investigation] [Priority: High]

Category: Denial Of Service
Description: An external agent prevents access to a data store on the other side of the trust boundary.
Justification: <no mitigation provided>

6. Data Flow Model 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>

7. Potential Process Crash or Stop for Django Middleware [State: Not Applicable] [Priority: High]

Category: Denial Of Service
Description: Django Middleware crashes, halts, stops or runs slowly; in all cases violating an availability metric.
Justification: This is not a problem caused or mitigated by Wagtail.

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

Category: Information Disclosure
Description: Improper data protection of Backend can allow an attacker to read information not intended for disclosure. Review authorization settings.
Justification: <no mitigation provided>

9. Potential Data Repudiation by Django Middleware [State: Needs Investigation] [Priority: High]

Category: Repudiation
Description: Django Middleware 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>

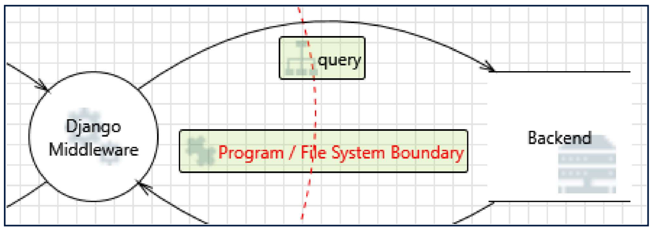
10. Spoofing of Source Data Store database [State: Needs Investigation] [Priority: High]

Category: Spoofing
Description: Backend may be spoofed by an attacker and this may lead to incorrect data delivered to Django Middleware. Consider using a standard authentication mechanism to identify the source data store.
Justification: <no mitigation provided>

11. Spoofing the Django Middleware Process [State: Not Applicable] [Priority: High]

Category: Spoofing
Description: Django Middleware may be spoofed by an attacker and this may lead to information disclosure by Backend. Consider using a standard authentication mechanism to identify the destination process.
Justification: Django Middleware is not at risk of being spoofed.

Interaction: query



12. Data Store Inaccessible [State: Needs Investigation] [Priority: High]

Category: Denial Of Service
Description: An external agent prevents access to a data store on the other side of the trust boundary.
Justification: <no mitigation provided>

13. Data Flow Database Query 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>

14. Potential Excessive Resource Consumption for Django Middleware or database [State: Needs Investigation] [Priority: High]

Category: Denial Of Service
Description: Does Django Middleware or Backend 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>

15. Data Flow Sniffing [State: Mitigation Implemented] [Priority: High]

Category: Information Disclosure
Description: Data flowing across query may be sniffed by an attacker. Depending on what type of data an attacker can read, it may be used to attack other parts of the system or simply be a disclosure of information leading to compliance violations. Consider encrypting the data flow.
Justification: Wagtail implements session token authentication and SSL.

16. Data Store Denies database Potentially Writing Data [State: Needs Investigation] [Priority: High]

Category: Repudiation
Description: Backend claims that it did not write data received from an entity on the other side of the trust boundary. Consider using logging or auditing to record the source, time, and summary of the received data.
Justification: <no mitigation provided>

17. The database Data Store Could Be Corrupted [State: Needs Investigation] [Priority: High]

Category: Tampering
Description: Data flowing across query may be tampered with by an attacker. This may lead to corruption of Backend. Ensure the integrity of the data flow to the data store.
Justification: <no mitigation provided>

18. Spoofing of Destination Data Store database [State: Needs Investigation] [Priority: High]

Category: Spoofing
Description: Backend may be spoofed by an attacker and this may lead to data being written to the attacker's target instead of Backend. Consider using a standard authentication mechanism to identify the destination data store.
Justification: <no mitigation provided>

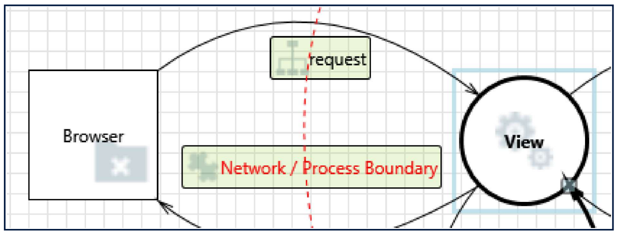
19. Spoofing the Django Middleware Process [State: Needs Investigation] [Priority: High]

Category: Spoofing

Description: Django Middleware may be spoofed by an attacker and this may lead to unauthorized access to Backend. Consider using a standard authentication mechanism to identify the source process.

Justification: <no mitigation provided>

Interaction: request



20. Spoofing the Browser External Entity [State: Mitigation Implemented] [Priority: High]

Category: Spoofing

Description: Browser may be spoofed by an attacker and this may lead to unauthorized access to View. Consider using a standard authentication mechanism to identify the external entity.

Justification: Wagtail requires the use of Django CsrfViewMiddleware.

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

Category: Elevation Of Privilege

Description: View may be able to impersonate the context of Browser in order to gain additional privilege.

Justification: The view does not have access to the browser.

22. Spoofing the Django Middleware Process [State: Needs Investigation] [Priority: High]

Category: Spoofing

Description: View may be spoofed by an attacker and this may lead to information disclosure by Browser. Consider using a standard authentication mechanism to identify the destination process.

Justification: <no mitigation provided>

23. Potential Lack of Input Validation for Django Middleware [State: Needs Investigation] [Priority: High]

Category: Tampering

Description: Data flowing across request may be tampered with by an attacker. This may lead to a denial of service attack against View or an elevation of privilege attack against View or an information disclosure by View. Failure to verify that input is as expected is a root cause of a very large number of exploitable issues. Consider all paths and the way they handle data. Verify that all input is verified for correctness using an approved list input validation approach.

Justification: <no mitigation provided>

24. Potential Data Repudiation by Django Middleware [State: Needs Investigation] [Priority: High]

Category: Repudiation

Description: View 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. Data Flow Sniffing [State: Mitigation Implemented] [Priority: High]

Category: Information Disclosure

Description: Data flowing across request may be sniffed by an attacker. Depending on what type of data an attacker can read, it may be used to attack other parts of the system or simply be a disclosure of information leading to compliance violations. Consider encrypting the data flow.

Justification: Wagtail implements session token authentication and SSL.

26. Potential Process Crash or Stop for Django Middleware [State: Needs Investigation] [Priority: High]

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

27. Data Flow User Credentials 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>

28. Django Middleware May be Subject to Elevation of Privilege Using Remote Code Execution [State: Needs Investigation] [Priority: High]

Category: Elevation Of Privilege
Description: Browser may be able to remotely execute code for View.
Justification: <no mitigation provided>

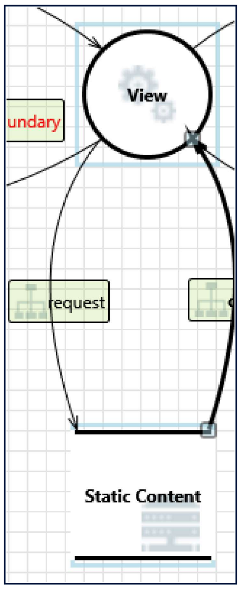
29. Elevation by Changing the Execution Flow in Django Middleware [State: Mitigation Implemented] [Priority: High]

Category: Elevation Of Privilege
Description: An attacker may pass data into View in order to change the flow of program execution within View to the attacker's choosing.
Justification: Wagtail requires the use of Django CsrfViewMiddleware.

30. Cross Site Request Forgery [State: Mitigation Implemented] [Priority: High]

Category: Elevation Of Privilege
Description: Cross-site request forgery (CSRF or XSRF) is a type of attack in which an attacker forces a user's browser to make a forged request to a vulnerable site by exploiting an existing trust relationship between the browser and the vulnerable web site. In a simple scenario, a user is logged in to web site A using a cookie as a credential. The other browses to web site B. Web site B returns a page with a hidden form that posts to web site A. Since the browser will carry the user's cookie to web site A, web site B now can take any action on web site A, for example, adding an admin to an account. The attack can be used to exploit any requests that the browser automatically authenticates, e.g. by session cookie, integrated authentication, IP whitelisting, … The attack can be carried out in many ways such as by luring the victim to a site under control of the attacker, getting the user to click a link in a phishing email, or hacking a reputable web site that the victim will visit. The issue can only be resolved on the server side by requiring that all authenticated state-changing requests include an additional piece of secret payload (canary or CSRF token) which is known only to the legitimate web site and the browser and which is protected in transit through SSL/TLS. See the Forgery Protection property on the flow stencil for a list of mitigations.
Justification: Wagtail requires the use of Django CsrfViewMiddleware.

Interaction: request



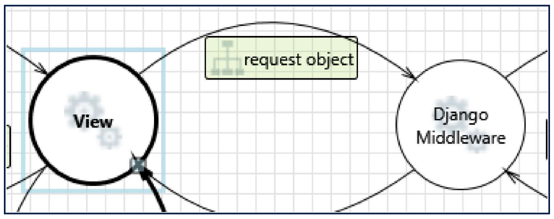
31. Spoofing of Destination Data Store Static Content [State: Mitigation Implemented] [Priority: High]

Category: Spoofing
Description: Static Content may be spoofed by an attacker and this may lead to data being written to the attacker's target instead of Static Content. Consider using a standard authentication mechanism to identify the destination data store.
Justification: Wagtail implements Django XSS protection.

32. Potential Excessive Resource Consumption for View or Static Content [State: Needs Investigation] [Priority: High]

Category: Denial Of Service
Description: Does View or Static Content 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>

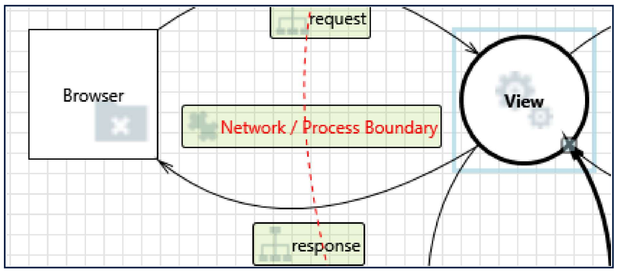
Interaction: request object



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

Category: Elevation Of Privilege
Description: Django Middleware may be able to impersonate the context of View in order to gain additional privilege.
Justification: Django Middleware cannot impersonate a view.

Interaction: response



34. Spoofing of the Browser External Destination Entity [State: Mitigation Implemented] [Priority: High]

Category: Spoofing
Description: Browser may be spoofed by an attacker and this may lead to data being sent to the attacker's target instead of Browser. Consider using a standard authentication mechanism to identify the external entity.
Justification: Wagtail includes Django SessionMiddleware which protects against session hijacking.

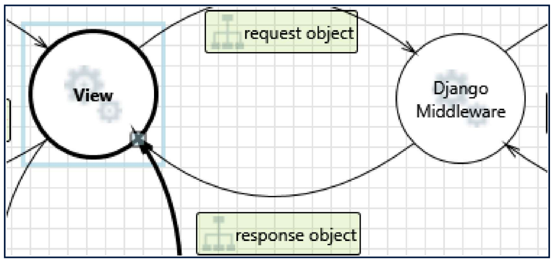
35. External Entity Browser Potentially Denies Receiving Data [State: Needs Investigation] [Priority: High]

Category: Repudiation
Description: Browser claims that it did not receive data from a process on the other side of the trust boundary. Consider using logging or auditing to record the source, time, and summary of the received data.
Justification: <no mitigation provided>

36. Data Flow Response Data 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>

Interaction: response object



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

Category: Elevation Of Privilege
Description: View may be able to impersonate the context of Django Middleware in order to gain additional privilege.
Justification: <no mitigation provided>