Wagtail Security

Wagtail release versions are numbered in the form A.B or A.B.C. The ‘C’ is the patch release version number which indicates a bugfix and/or security release. These releases will be 100% backwards-compatible with the previous patch release. Wagtail requires several common Django middleware which supply basic security controls. Examples of Django security concerns that have been addressed in the past include:

* CVE-2016-7401: An interaction between Google Analytics and Django's cookie parsing allows an attacker to set arbitrary cookies leading to a bypass of CSRF protection.
* CVE-2016-9013:When running tests with an Oracle database, Django creates a temporary database user. In older versions, Django hard-codes a password if a password is not manually specified. A randomly generated password is now used for each test run.
* CVE-2016-9014: Older versions of Django do not validate the Host header against settings. This makes them vulnerable to a DNS binding attack.

Other security controls that are covered by Django security middleware include:

* **XSS Protection:** Using Django templates protects you against the majority of XSS attacks. Django templates escape specific characters which enable executing code.
* **Cross-Site Request Forgery (CSRF) Protection:** CSRF protection checks every POST request for a secret token. A malicious user cannot replay a form post to our website because they do not know the secret token.
* **CsrfViewMiddleware** will verify that the HTTP referrer is a URL on the same origin (including subdomain and port).
* **SQL Injection Protection:** Arbitrary SQL code is prevented from being executed by using Django’s QuerySets. The SQL will be properly escaped by the underlying database driver.
* **Clickjacking Protection:** X-Frame-Options middleware prevents a site from being rendered inside a frame (clickjacking).
* **HTTPS Security:** HTTPS prevents malicious network users from sniffing traffic to discover authentication credentials.
* **Session Security:** The django.contrib.sessions framework contains the SessionMiddleware and prevents access to other users’ sessions.
* **User Uploaded Content:** Django allows for handlers to be disabled so that static files cannot be executed as code.