Security engineering overview

The security model of Exponential for engineers - access control through policies, output escaping, safe SQL, CSRF tokens, the secure defaults of 2026, the HTTP security layer, authentication and two-factor sign-in, the...

Source file: `doc/eng-us/help/software-engineers/security-engineering-overview.md` Last changed 2026-11-04
https://doc.exponential.earth/learninminutes/exphelp/viewer/help/software-engineers/security-engineering-overview.md

This page is for engineers who assess, design or review the security of an Exponential system. It names the mechanisms, where they live, what is on by default, and where the project itself says it falls short. The standards status is Security standards; the attacker-oriented operator's guide is Security hardening.

Contents

1. The mechanisms
2. Secure defaults
3. Hardening passes on record
4. The project's own assessment
5. Reporting and fixing vulnerabilities
6. Where to go next

The mechanisms

Threat	Mechanism	Where
Broken access control	roles, policies and limitations checked for every module view; content reads filtered by permission	Modules, views and access control
Injection (SQL)	`escapeString()` and integer casts; persistent objects escape for you	Writing portable SQL
Injection (shell)	arguments through `escapeshellarg()`	Security hardening of 6.0.13
Cross-site scripting	`|wash` in templates; Content Security Policy with nonces	HTTP security
Cross-site request forgery	form tokens on every form (`ezformtoken`)	ezformtoken and CSRF
Weak authentication	`password_hash()`, minimum length 10, optional lockout, two-factor sign-in	Authentication, sessions and two-factor sign-in
Spoofed proxy headers	forwarded headers trusted only from `TrustedProxies[]`	Trusted proxies
Open redirects	allowed redirect hosts	Safe redirects
Malicious uploads	upload checks, SVG sanitizing	SVG images
Configuration drift in production	`ManagedByDeployment`	Configuration managed by the deployment

Threat	Mechanism	Where
Undetected abuse	tamper-evident audit trail, request classifier and `security.log`	Observability

Secure defaults

In September and October 2026 security headers, `HttpOnly` and `SameSite` session cookies, a stricter upload check and new access requirements for some remote services were turned on by default. Security fixes are treated like bug fixes: on by default, documented with how to undo them deliberately (Security defaults, Behaviour changes of 16 to 30 September 2026).

Shipped values worth knowing (`settings/site.ini`): security headers such as `X-Content-Type-Options: nosniff`, `X-Frame-Options: SAMEORIGIN` and `Referrer-Policy: strict-origin-when-cross-origin`; `MinPasswordLength=10`; `HashType=php_default`; `MaxNumberOfFailedLogin=0` (the lockout is off until you set it); the Content Security Policy in `Mode=auto` (report-only on the administration, off on sites).

Hardening passes on record

Pass	Content
6.0.13	four SQL injections, two OS command injections, one reflected cross-site scripting (Security hardening of 6.0.13)
August 2026	Security hardening, August 2026
September 2026	Security defaults
October 2026	the fixes of the Opencontent and Mugo Web forks (pull request #345): a URL alias depth limit, one-line eZLog entries, REST listeners, a longer upload blacklist, a subtree `HardLimit` (Fixes from the Opencontent and Mugo Web forks)

The 6.0.13 page also lists the patterns extension developers must audit in their own code: a kernel update does not fix them in extensions.

The project's own assessment

The security standard rates OWASP Top 10 (2021) as **Partly complies**, with mitigations for all ten categories and named gaps (account lockout off by default, no dependency scanning in CI, public-site CSP off), OWASP ASVS 4.0.3 as **Unverified** (no assessment made), and Content Security Policy Level 3 as **Partly complies**. "Complies" would need an external assessment, which has not been made. Use this list as the starting point of your own threat model.

Reporting and fixing vulnerabilities

- Report privately by e-mail to the address in `SECURITY.md`, not in the public issue tracker or discussions. A first answer is promised within one business day.
- Include the version, the PHP version, the engine, the database, what an attacker can do and the steps to reproduce; leave out passwords, secrets and personal data.
- Fixes are made on `main` and released with the next 6.0.x version. A published tag is never

changed: a fix for a problem in an older tag reaches you as the next release. Only the 6.0 line is supported.

Where to go next

- HTTP security: headers, CSP, request rules and proxies
- Authentication, sessions and two-factor sign-in
- Security review checklist
- Chapter 13 of the installation book: Security hardening

Related pages

- Security for applications
- Security in extensions
- Security and audit
- Free documentation: this page is free to copy, translate and improve under the GNU FDL 1.3.

> **Become an Exponential Partner**
>
> Partners share their information, provide funding, contribute back and develop the products with us, and that is how Exponential grows. A partner website and dashboard are planned for mid-2027: please become a partner today to speed up our growth rate and land new features faster.
>
> Partners: start here | How to apply

Exponential Copyright © 1998-2026 Exponential Foundation and 7x
Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".
This documentation is free documentation, licensed under the GNU Free Documentation License 1.3 or later:
https://www.gnu.org/licenses/fdl-1.3.html.