TLS and HTTPS

How a host serves every Exponential site over HTTPS only: who terminates TLS, redirecting plain HTTP, HSTS in the kernel and in Velocity, secure session cookies, proxies and load balancers that terminate TLS...

Source file: `doc/eng-us/help/hosting/security/tls-and-https.md` Last changed 2026-11-04
https://doc.exponential.earth/exphelp/viewer/help/hosting/security/tls-and-https.md

This page is for system administrators who serve Exponential sites over HTTPS, and for support staff who meet the usual symptoms: `http://` links behind a TLS proxy, sign-in loops, or a browser warning. It says who can terminate TLS, which Exponential settings must agree with that choice, and how to check the result. The reference is 13.9 Security headers and HTTPS and 13.10 Behind a proxy.

Contents

1. Who terminates TLS
2. HTTPS everywhere, step by step
3. HSTS
4. Secure cookies
5. Proxies that terminate TLS
6. Watching certificates
7. Checks
8. What goes wrong
9. Checklist

Who terminates TLS

Setup	TLS is ended by	Certificate handled by
Apache with PHP-FPM	Apache `mod_ssl`	the panel or an ACME client on the server
nginx with PHP-FPM	nginx `ssl`	the panel or an ACME client on the server
Velocity facing visitors	Velocity itself, with HTTP/2	Velocity: a certificate you give it, a self-signed stand-in, or ACME (`letsencrypt` mode)
Velocity or PHP-FPM behind a proxy, load balancer or CDN	the proxy	the proxy's operator

Velocity needs no `mod_ssl`, stunnel or proxy for HTTPS; it verifies a certificate pair before using it, swaps a renewed pair in live within `watchInterval` (60 seconds), and serves a self-signed stand-in rather than a broken pair (8.3.9). Details for hosts: Velocity HTTPS certificates; panel certificates: TLS certificates in panels.

HTTPS everywhere, step by step

1. **Serve every name of the site over HTTPS**, including `www.` and the administration's host name.
2. **Redirect plain HTTP to HTTPS** in Apache or nginx (sections 8.5 and 8.6 of Chapter 8). Velocity listens on both ports and has no setting that redirects one to the other; put the redirect in front, or close the plain port to the outside. Exponential's own `[SSLZoneSettings]` in `site.ini` can switch individual subtrees, for a site that cannot be HTTPS-only yet.
3. **Make the site build HTTPS addresses**: `site.ini [SiteSettings] SiteURL` and the siteaccesses' URLs.
4. **Set** `[Session] CookieSecure=true` (see Secure cookies).
5. **Turn HSTS on**, short first (see HSTS).
6. **Watch certificate expiry** (see Watching certificates).

HSTS

A browser that has seen `Strict-Transport-Security` refuses plain HTTP to that host for the whole period, even after you lower the value. Start short, raise it to a year only when every name is served over HTTPS for good.

Who sends it	Setting	Shipped
Velocity, on every HTTPS answer	`velocity.ini [HTTPSSettings] HSTSMaxAge, HSTSIncludeSubDomains, HSTSPreload`	`300`, `disabled`, `disabled`
the kernel, under Apache or nginx	`site.ini [HTTPHeaderSettings] SecurityHeaders[Strict-Transport-Security]`	empty (off); sent only over HTTPS

INI

```
# settings/override/site.ini.append.php, once every name is HTTPS for good
[HTTPHeaderSettings]
SecurityHeaders[Strict-Transport-Security]=max-age=31536000
```

Preload needs `HSTSIncludeSubDomains` and at least a year (8.3.9, Strict-Transport-Security).

> **Warning**
>
> `includeSubDomains` applies to every subdomain of the name, including ones that other teams or other customers may run on plain HTTP. Agree it with the owner of the domain before turning it on.

Secure cookies

`settings/site.ini [Session]`, with the defaults that apply while the keys are unset (13.7):

Key	Default	For an HTTPS-only site
`CookieSecure`	`auto`: Secure when the request itself came over HTTPS	`true`
`CookieHttponly`	`true`	keep

Key	Default	For an HTTPS-only site
CookieSameSite	Lax	keep; None only with Secure

INI

```ini
# settings/override/site.ini.append.php
[Session]
CookieSecure=true
```

`CookieSecure=true` makes sure the session cookie is never sent over plain HTTP, even on a request that wrongly arrives without TLS.

Proxies that terminate TLS

When a proxy ends TLS, PHP sees a plain HTTP request and learns about HTTPS only from headers such as `X-Forwarded-Proto`. Any visitor can send those headers too, so since 6.0.15 the kernel believes them **only when the connection comes from an address in** `site.ini` `[HTTPHeaderSettings]` `TrustedProxies[]`. The default is the loopback addresses `127.0.0.1` and `::1`.

Setup	What to do
No proxy	nothing; forwarded headers a visitor sends are ignored
A proxy on the same machine that sets the headers	nothing; make the proxy set `X-Forwarded-Proto` itself
A proxy, load balancer or CDN on another address	list its addresses or ranges, and only those; firewall the application server so it cannot be reached around them
A front end that rewrites `REMOTE_ADDR` (`mod_remoteip`, nginx `realip`, as in Plesk's nginx in front of Apache)	nothing; `ClientIpByCustomHTTPHeader` stays `false`

INI

```ini
# settings/override/site.ini.append.php: a load balancer on a private subnet
[HTTPHeaderSettings]
TrustedProxies[]
TrustedProxies[]=10.0.0.0/16
ClientIpByCustomHTTPHeader=X-Forwarded-For
```

The visitor's address is read from the right of `X-Forwarded-For`, so a forged entry further left is never used. Entries are addresses or ranges; host names are not resolved, and a comment on the same line makes the entry invalid.

Under Velocity, list the proxies in Velocity's own `Q.webserver.proxy.trusted`, in a snippet under `/etc/vc/conf-available/` enabled with `exp:velocity conf enable <name>`; Velocity then passes the visitor as `REMOTE_ADDR` and sets `$_SERVER['HTTPS']` itself (from engine release 0.0.4.44 on, only for trusted peers). Details: 13.10, Under Velocity and Forwarded headers are trusted only from configured proxies.

Watching certificates

- Velocity: `php bin/php/console exp:velocity ssl show --json --allow-root-user` and watch `daysLeft`. A 90-day CA certificate is renewed when about 30 days are left, so fewer

than 14 days left means renewal has failed for more than two weeks (8.3.9, Inspecting certificates).

- Apache or nginx: your ACME client's timer or the panel's renewal, plus an external expiry check.
- Keep NTP running: certificates are checked against the system clock.

When it has already happened: Certificate expiry.

Checks

Shell

```
# headers on a page
curl -sI https://example.com/ | grep -i -E 'strict-transport|x-content-type|x-frame|referrer|permissions|content-security'
# the session cookie: expect Secure; HttpOnly; SameSite=Lax
curl -sI https://example.com/user/login | grep -i set-cookie
# directly, no proxy: a visitor's own header must change nothing (the redirect stays http)
curl -s -o /dev/null -w '%{redirect_url}\n' -H 'X-Forwarded-Proto: https' http://example.com/user/logout
```

What goes wrong

Symptom	Cause
Absolute URLs and redirects say `http://` behind an HTTPS proxy; a redirect loop on the sign-in page	the proxy is not in `TrustedProxies[]` (or not in Velocity's list), or it does not send `X-Forwarded-Proto`
Signed-in users are signed out on every page with `CookieSecure=true`	the same: the kernel thinks the request is plain HTTP
Every visitor has the proxy's address in the audit and lockout	`ClientIpByCustomHTTPHeader` not set, or the proxy not trusted
A visitor can still choose the scheme or the address	a trusted range far wider than your proxies, or the application server reachable directly
Browser warns about the certificate on a Velocity site	Velocity serves its self-signed stand-in: the real pair is expired, mismatched or not yet issued; `exp:velocity ssl show`

Checklist

☐ Every name of the site serves HTTPS; plain HTTP redirects.

☐ `SiteURL` and the siteaccesses' URLs use `https://`.

☐ `[Session] CookieSecure=true`.

☐ HSTS on, short first; raised to a year once stable; `includeSubDomains` agreed with the domain owner.

☐ Behind a proxy: `TrustedProxies[]` (or `Q.webserver.proxy.trusted`) lists exactly your proxies; the `X-Forwarded-Proto` check above changes nothing.

☐ Certificate expiry monitored; NTP running.

Related pages

- Security for hosts: overview
- Velocity HTTPS certificates
- TLS certificates in panels
- Varnish and reverse proxies
- Certificate expiry
- Chapter 13: Security hardening for production
- 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.