

How to secure your system

A twelve-point checklist for leaders, with a question to put to your engineers for each point.

You do not need to be technical to check whether your systems are safe. You need to ask the right questions and notice when the answers are vague. Each point below says why it matters in plain words, then gives one question to ask. A confident, specific answer is a good sign. A pause, or an answer that starts with "I think", is worth following up.

01 Sign-in uses a proven standard

Home-made login code is where many breaches start. A standard such as OAuth 2.0 or OpenID Connect, used with PKCE, has been attacked and hardened for years.

Ask your team: Which standard do we use for sign-in, and do we verify tokens on the server and allow only exact, pre-approved redirect addresses?

02 Sessions can be ended and cannot be reused

A session is the proof that someone is signed in. It should expire when idle, expire after a maximum time, and be revocable the moment you need it to be.

Ask your team: Can we end one person's sessions immediately, and are session tokens stored only as hashes on the server?

03 A stolen cookie is not enough

Tie each session to the browser that created it, and mark cookies HttpOnly, Secure and SameSite. Then a stolen cookie alone does not let an attacker in.

Ask your team: Is each session bound to the device it started on?

04 Everyone with power uses two-factor

Admin, cloud, email and code hosting accounts are the keys to everything. A password alone is not enough for any of them.

Ask your team: Is two-factor switched on for every admin and cloud account, with no exceptions?

05 People get the least access they need

Permissions must be enforced by the server and the database, not by hiding buttons. Anything not explicitly allowed should be refused.

Ask your team: If someone edits the request by hand, does the server still refuse what their role should not do?

06 Cross-site attacks are blocked

CSRF protection stops other websites from making your users' browsers do things. A strict Content Security Policy limits what scripts can run on your pages.

Ask your team: Is every state-changing request protected against CSRF, and what does our Content Security Policy say?

07 All input is checked on the server

Treat everything that arrives as hostile: forms, URLs, API calls, and the contents of uploaded files and spreadsheets. Checks in the browser are only for convenience.

Ask your team: Where is input validated and cleaned on the server, and does that include imported files?

08 Encryption in transit, everywhere

Every connection should use TLS, with HSTS so browsers refuse the unencrypted version, and certificates that renew on their own.

Ask your team: Does anything still run over plain HTTP, and who is warned if a certificate is about to expire?

09 Servers are locked down

Use key-only SSH with no root login, a firewall that allows only the ports you need, automatic banning of repeated bad logins, and automatic security updates. Databases and admin tools never sit on the public internet.

Ask your team: What ports are open to the internet right now, and who can sign in to the servers?

10 Secrets stay secret

Passwords, keys and tokens must never appear in code, web addresses or logs. They live in a proper secrets store and are changed when someone leaves.

Ask your team: Where are our secrets kept, and when were they last rotated?

11 Every important action leaves a record

Sign-ins, approvals, edits and deletions should go into a log nobody can quietly edit, and the log should be searchable, so it is useful at the moment of an incident.

Ask your team: Can we answer who did what, and when, for any sensitive action in the last six months?

12 Abuse is slowed down, and recovery is proven

Rate limits and lockouts blunt password guessing and bots. Backups only count if you have tested restoring from them, and someone should be told when something looks wrong.

Ask your team: When did we last restore from a backup, and who gets an alert when login failures spike?

This list is a starting point, not a full audit. If you would like a second pair of eyes on any of it, get in touch at benedictasare2@gmail.com.