RESPONSIBLE DISCLOSURE of ZENNNN SECURITY VULNERABILITIES

Last updated: March 20, 2020

Responsible Disclosure Policy

The safety of ZENNNN systems is very important to us, and we consider security problems with the highest priority. We do our best every day to protect ZENNNN users from known security threats, and we welcome all reports of security vulnerabilities discovered by our users and contributors. We are committed to handle vulnerability reports with the greatest attention, provided that the following rules are respected.

Reporting an issue

Please share privately the details of your security vulnerability by emailing our Security Team at security@zennnn.com. Make sure to include as much information as possible, including the detailed steps to reproduce the problem, the versions that are affected, the expected results and actual results, and any other information that might help us react faster and more efficiently. We tend to prefer text-based bug descriptions accompanied with a proof-of-concept script/exploit, rather than long videos.

«
Important note: *we receive a majority of security reports that have little to no impact on the security of ZENNNN System, and we ultimately have to reject them. To avoid a disappointing experience when contacting us, please try to put together a proof-of-concept attack and take a critical look at what's really at risk. If the proposed attack scenario turns out unrealistic, your report will probably be rejected. Also be sure to review our list of non-qualifying issues below.*
»

You may send this report from an anonymous email account, although we promise not to disclose your identity if you do not want us to.

Incident Response Procedure

1. You privately share the details of the security vulnerability by reporting an issue via email address (security@zennnn.com);
2. We acknowledge your submission and verify the vulnerability. Our first answer generally comes under 48h;
3. If the vulnerability is valid and in scope, we request a CVE ID and give it to you as soon as it is assigned;
4. We work on a correction in collaboration with you;
5. We write a detailed Security Advisory describing the issue, its impacts, possible workarounds and solution, and we ask you to review it;

6. We privately broadcast the Security Advisory and the correction to stakeholders and customers with an Agreement;
7. We give stakeholders and customers a reasonable delay to apply the correction, before disclosing it publicly (e.g. 2-3 weeks);
8. We disclose and broadcast the Security Advisory and the correction on our web site.

Rules

We ask you to observe the following rules at all times:
1. Exclusively test vulnerabilities on your own deployments before you start normal work;
2. Never attempt to access or modify data that does not belong to you;
3. Never attempt to execute denial of service attacks, or to compromise the reliability and integrity of services that do not belong to you;
4. Do not use scanners or automated tools to find vulnerabilities, as their effects will violate the previous rules;
5. Never attempt non-technical attacks such as social engineering, phishing, or physical attacks against anyone or any system;
6. Do not publicly disclose vulnerabilities without our prior consent (see also the Disclosure Procedure above). During the non-disclosure period you are authorized to use/test any correction we've provided, as long as no emphasis is put on that correction and it is not published in the form of a security report (i.e. using it on production servers is fine).

In return:
1. We will not initiate legal action against you if you followed the rules;
2. We will process your report and respond as quickly as possible;
3. We will provide a fix as soon as possible;
4. We will keep you updated of the progress and disclosure steps (see also the Disclosure Procedure above);
5. We will work diligently with stakeholders and customers in order to help them restore the safety of their system;
6. We will not publicly disclose your identity if you do not want to be credited for your discovery.

What to report?

Qualifying vulnerabilities — DO REPORT!
- SQL injection vectors in public API methods;
- XSS vulnerabilities working in supported browsers
- Broken authentication or session management, allowing unauthorized access
- Broken sandboxing of customizations, allowing arbitrary code execution or access to system resources

NON-Qualifying vulnerabilities — DO NOT REPORT!
- XSS vulnerabilities working only in unsupported/deprecated browsers, or requiring relaxed security settings

- Self-XSS attacks requiring the user to actively copy/paste malicious code into their own browser window
- Pseudo-XSS vulnerabilities.
- Rate-limiting / Brute-forcing / Scripting of components working as designed (e.g. password authentication, password reset, etc.)
- File path disclosures, which do not carry significant risk and do not enable attacks that would be otherwise impossible
- Clickjacking or phishing attacks using social engineering tricks to abuse users, with the system working as intended
- Tabnapping or other phishing attacks conducted by navigating other browser tabs
- Logout CSRF (no plausible attack + not preventable e.g. via cookie tossing or cookie jar overflow)
- Open redirectors, which are simply one vector for phishing among many others (see our detailed explanation)
- Reflected File Downloads, another attack technique that requires social engineering and is not very practical
- Referrer leak (including sensitive tokens) via social media links or ads/analytics requests — very unlikely to be clicked, or to be exploited within validity period by those mainstream companies!
- More generally, attacks relying on physical or social engineering techniques will usually be rejected
- Non-permanent Denial of Service (DoS) and distributed DoS (DDoS) that maintain resource exhaustion (cpu/network/memory) via a sustained stream of requests/packets
- Password policies (length, format, character classes, etc.)
- Missing or partial verification of email addresses
- Disclosure of public information or information that does not carry significant risks (directory listing on our downloads archive is a required feature! ;-))
- Spam-fighting policies and systems such as DKIM, SPF or DMARC
- Absence of HTTP Strict Transport Security (HSTS) headers, HSTS preloading, and HSTS policies
- Weak ciphers or SSL deployments details (our benchmark is an A grade on SSLLab's test)
- Issues in default configuration of access control rules (e.g. ACLs and record rules) - please open regular bug reports instead
- Attack scenarios that rely on a takeover of user email accounts (obviously)

If you have any doubt, please ask us first!