



RADICALLY
OPEN
SECURITY

Penetration Test Report

GlobalLeaks

V 1.0
Amsterdam, August 18th, 2022
Public

Document Properties

Client	GlobaLeaks
Title	Penetration Test Report
Targets	Public Source-Code on GitHub (https://github.com/globaleaks/GlobaLeaks/) Audit of GlobaLeaks server source-code Pentest of GlobaLeaks frontend OpSec for whistleblowers OpSec for server administrators
Version	1.0
Pentesters	Tillmann Weidinger, Stefan Grönke
Authors	Stefan Grönke, Marcus Bointon
Reviewed by	Marcus Bointon
Approved by	Melanie Rieback

Version control

Version	Date	Author	Description
0.1	August 10th, 2022	Stefan Grönke	Initial draft
0.2	August 17th, 2022	Marcus Bointon	Review
1.0	August 18th, 2022	Marcus Bointon	1.0

Contact

For more information about this document and its contents please contact Radically Open Security B.V.

Name	Melanie Rieback
Address	Science Park 608 1098 XH Amsterdam The Netherlands
Phone	+31 (0)20 2621 255
Email	info@radicallyopensecurity.com

Radically Open Security B.V. is registered at the trade register of the Dutch chamber of commerce under number 60628081.

Table of Contents

1	Executive Summary	5
1.1	Introduction	5
1.2	Scope of work	5
1.3	Project objectives	5
1.4	Timeline	6
1.5	Results In A Nutshell	6
1.6	Summary of Findings	7
1.6.1	Findings by Threat Level	9
1.6.2	Findings by Type	10
1.7	Summary of Recommendations	10
2	Methodology	12
2.1	Planning	12
2.2	Risk Classification	12
3	Reconnaissance and Fingerprinting	14
4	Findings	15
4.1	GL-050 — XSS can install a service worker to compromise in-memory sessions	15
4.2	GL-049 — Frontend error contain sensitive information	16
4.3	GL-046 — Page title discloses access to a report	18
4.4	GL-024 — SQL Injection would lead to arbitrary file read	19
4.5	GL-023 — Self-hosted mail notifications sent through mail.globaleaks.org	21
4.6	GL-014 — Administrators Can Upload Arbitrary JavaScript	22
4.7	GL-008 — Administrators Can Download TLS Private Keys	23
4.8	GL-007 — User password reset without old password	25
4.9	GL-006 — Possible Privilege Escalation Through Filesystem Changes	31
4.10	GL-004 — Network Settings Allow Arbitrary Host Name	33
4.11	GL-032 — Session/Token Decorator can be bypassed	35
4.12	GL-036 — Development mode listens on all interfaces	36
4.13	GL-031 — Admin API file listing is available to regular users	37
4.14	GL-027 — Exception Emails Are Sent To Globaleaks.org	38
4.15	GL-026 — 2FA can be disabled	40
4.16	GL-022 — E-Mail Spoofing For Globaleaks.org	41
4.17	GL-016 — Arbitrary File Upload As Administrator	45
4.18	GL-015 — HTML Injection in Questionnaire Options Label	47
4.19	GL-041 — Template keywords from user-data resolve	50
4.20	GL-040 — Export zip file path traversal	53

4.21	GL-033 — Exceptions API endpoint can be used to send arbitrary mail to administrators	55
4.22	GL-029 — Mail content injection in Support Ticket request	57
4.23	GL-018 — The Twisted Webserver On try.globaleaks.org Is Outdated And Has Known Vulnerabilities	59
4.24	GL-017 — try.globaleaks.org Leaks Python Webserver Version	60
4.25	GL-013 — Possible HTML Content Injection Sinks	61
4.26	GL-012 — Simplified Login Leaks Registered Usernames	63
5	Non-Findings	65
5.1	NF-044 — Absence of RCE through subprocess	65
5.2	NF-043 — CSRF protection	65
5.3	NF-042 — Absence of SQL injection in GlobaLeaks backend	65
5.4	NF-035 — Plain text mail only	66
5.5	NF-030 — Submission attachments are never stored in plain text	66
5.6	NF-020 — User-Provided HTML Content Is Properly Sanitized	66
5.7	NF-002 — Properly Configured robots.txt And sitemap.xml	66
6	Future Work	68
7	Conclusion	69
Appendix 1	Testing team	70

1 Executive Summary

1.1 Introduction

Between June 2, 2022 and August 10, 2022, Radically Open Security B.V. carried out a penetration test for GlobaLeaks

This report contains our findings as well as detailed explanations of exactly how ROS performed the penetration test.

1.2 Scope of work

The scope of the penetration test was limited to the following targets:

- Public Source-Code on GitHub (<https://github.com/globaleaks/GlobaLeaks/>)
- Audit of GlobaLeaks server source-code
- Pentest of GlobaLeaks frontend
- OpSec for whistleblowers
- OpSec for server administrators

The scoped services are broken down as follows:

- Scoping effort: 1 days
- Pentest GlobaLeaks frontend (incl. reporting): 5.5 days
- Source code audit GlobaLeaks server (incl. reporting): 5.5-16 days
- Whistleblower OpSec analysis of traces persisting after usage (incl. reporting): 6-8 days
- Retest and fix verification: 2 days
- **Total effort: 20 - 32.5 days**

1.3 Project objectives

ROS will perform an audit and penetration test of GlobaLeaks in order to assess the security of the whistleblowing platform. To do so ROS will assess the source code and the <https://try.globaleaks.org> demo environment and guide GlobaLeaks in attempting to find vulnerabilities, exploiting any such found to try and gain further access and elevated privileges.

1.4 Timeline

The Security Audit took place between June 2, 2022 and August 10, 2022.

1.5 Results In A Nutshell

We discovered 0 Critical, 0 High, 10 Elevated, 8 Moderate, and 8 Low-severity issues during this penetration test.

GlobaLeaks stores an `x-session` authentication credential in memory, so that no traces of Cookies are left after using the platform as administrator, receiver, or whistleblower.

We have identified a routing issue [GL-032](#) (page 35) that allowed bypassing authentication checks, which in turn allowed bypassing of DoS protection, but it did not lead to compromise of user accounts.

Administrators were able to execute arbitrary JavaScript in the authentication page [GL-014](#) (page 22) or upload assets with an executable MIME type [GL-016](#) (page 45). HTML injection in questionnaire option labels [GL-015](#) (page 47) and other page content [GL-013](#) (page 61) by administrators was possible, but was sanitized and not found to be vulnerable to script injection (stored XSS). Because of shared domain names, it would have been possible for tenant administrators to compromise the global application.

An administrator of GlobaLeaks without SSH root access should not be able to download SSL certificates and private keys [GL-008](#) (page 23), further allowing them to compromise user or more powerful administrator sessions with network interception attacks. When running multiple instances on the same IP address the ability to set arbitrary hostnames [GL-004](#) (page 33) allowed compromising other subdomains and download their signed encryption credentials. Both have now been mitigated, so that only the root user of a host is able to do this.

Before GlobaLeaks drops privileges, an adversary with the ability to access the remote filesystem as the `globleaks` user was able to use the service startup routine to escalate local privileges to the root user [GL-006](#) (page 31).

By installing a service worker, any XSS vulnerability would have allowed permanent compromise of user interaction [GL-050](#) (page 15), even after reloading the single-page application.

Because frontend error messages could contain sensitive information [GL-049](#) (page 16) that feature was removed entirely. We also identified that changes in the page title leave traces in browser history [GL-046](#) (page 18).

One API endpoint serving uploaded assets could allow arbitrary file read if database entries could have been injected into the database [GL-024](#) (page 19). The upload feature itself was always only available to administrators though and did not allow inserting data with path traversal the regular way. Additionally, GlobaLeaks uses an ORM layer with hardening against SQL injection vulnerabilities, which made this vector unlikely to be exploitable, confirmed by our assessment of existing vulnerabilities. Unauthenticated clients were able to list all assets uploaded by an administrator [GL-031](#) (page 37) even though they were not referenced yet and potentially not meant to be public.

Several minor vulnerabilities were identified in relation to the default mail service offered by GlobaLeaks. In the default configuration the server sends outgoing mail [GL-023](#) (page 21), including user and administrator and receiver password reset mails. Although GlobaLeaks emphasizes decentralization, the hosted mail infrastructure could have been a single point of failure. This situation was hardened by improving the 2FA enforcement and leaving a clear

warning message to operators of an appliance in the administrator backend when the default mail setup is in use. We found fault with frontend exception reporting through the default mail setup [GL-027](#) (page 38), as it could undermine a whistleblower's anonymity if it contained sensitive information. In response to this, the frontend error reporting feature has been removed entirely. Furthermore we have found that mail was sent from `@globeleaks.org` aliases, allowing spoofing of mail [GL-022](#) (page 41) that appears to come from members of the organization. Lastly, it was possible to inject template keywords [GL-041](#) (page 50) or arbitrary content into support tickets [GL-029](#) (page 57) and exception notifications [GL-033](#) (page 55) – all potentially useful in phishing attacks.

Although no reflected or stored XSS (exploitable by non-admin users) was found, we flagged the ability to take over accounts without providing 2FA or the original account password [GL-007](#) (page 25) as an important matter to address. A similar issue applied to administrators that were previously able to reset another user's 2FA without further confirmation [GL-027](#) (page 38), which is particularly relevant because application admins are not allowed to read submissions.

Of course we were eager to find vulnerabilities through anonymous submissions by whistleblowers. In this regard it was found possible to upload attachments that can cause ZIP file path traversal [GL-040](#) (page 53), although exploitation would have been strongly dependent on the receiver's tooling to extract and handle archive files.

Because the `try.globeleaks.org` server leaked its Python Twisted server version [GL-017](#) (page 60) we were able to identify potential Denial of Service vulnerabilities [GL-018](#) (page 59) in the version deployed from Debian packages.

Even though GlobaLeaks aims to protect whistleblower privacy, we had to mention that a simplified login form leaks sensitive information about existing user accounts [GL-012](#) (page 63) of which only administrator accounts are not public by default anyway.

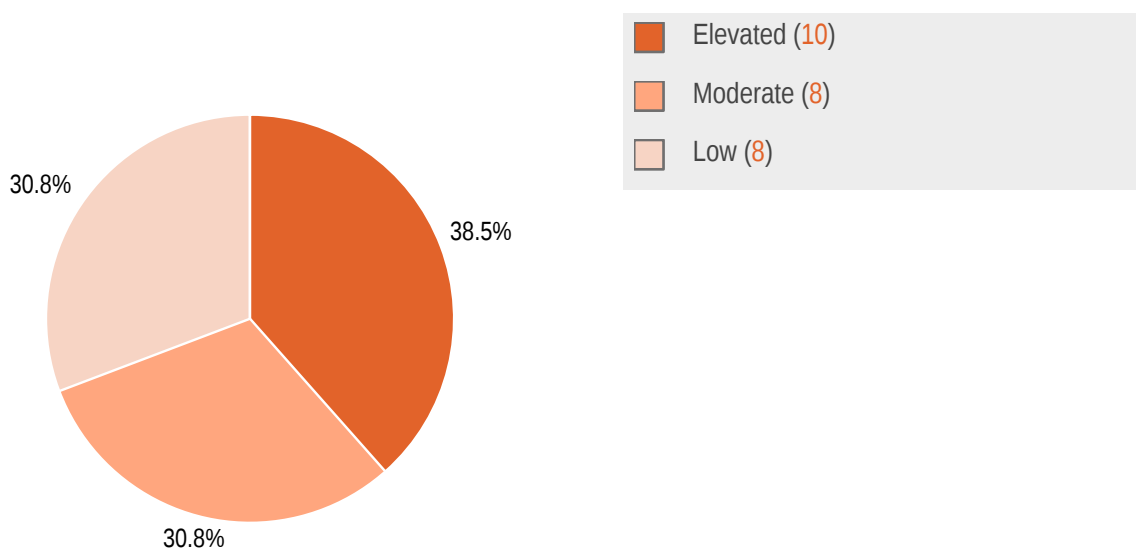
1.6 Summary of Findings

ID	Type	Description	Threat level
GL-050	Missing Hardening	GlobaLeaks holds session credentials in memory only, mitigating access through directly loaded resources that trigger XSS. When installing a service worker though, the next session opened by an administrator or receiver might still be compromised by injecting malicious scripts.	Elevated
GL-049	Information Disclosure	Frontend exception alert mails contain sensitive information about the client, such as submitted data and the user-agent string.	Elevated
GL-046	Unknown	The GlobaLeaks frontend application updates the page title based on user activity, so that the last user activity is stored in the browsing history.	Elevated
GL-024	Information Disclosure	An SQL Injection vulnerability in SQLite would lead to arbitrary file read on the remote server.	Elevated
GL-023	Information Disclosure	By default, outgoing mails from self-hosted instances of GlobaLeaks are sent through mail.globeleaks.org.	Elevated

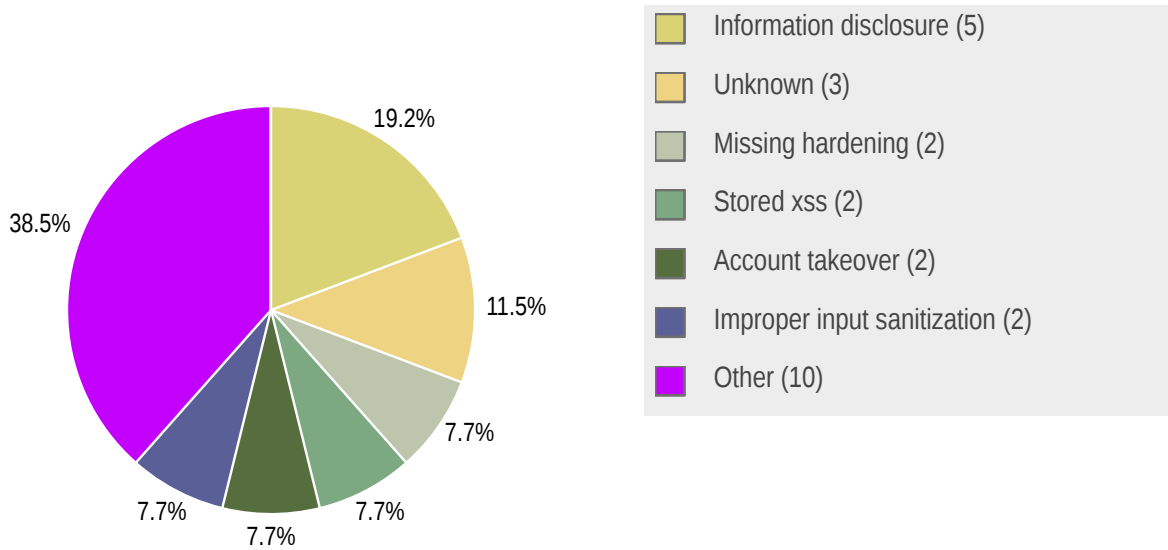
GL-014	Stored XSS	The theme customization feature allows administrators to upload and embed arbitrary JavaScript into the application.	Elevated
GL-008	Exposed Encryption Private Keys	The admin UI provides the TLS encryption key-pair (for instance signed by Let's Encrypt). These can be downloaded by any administrative account.	Elevated
GL-007	Account Takeover	It is possible to change another user's account password with access to only their x-session, for instance when leaving a browser session unattended or falling victim to an XSS vulnerability.	Elevated
GL-006	Local Privilege Escalation	Before the GlobaLeaks daemon drops privileges, a cleanup task assigns filesystem ownership to the executing user, but recurses over symbolic links.	Elevated
GL-004	Subdomain Takeover	Administrators can configure arbitrary host names as a custom domain name, allowing them to generate and download certificates for all (sub-)domains configured on the same IP address.	Elevated
GL-032	DoS Protection Bypass	The request decorator requiring either a request session or token can be bypassed by appending /api/token to any API URL.	Moderate
GL-036	Exposed Service	When the GlobaLeaks daemon is run in development mode, it exposes HTTP services on port 8080 on all available interfaces.	Moderate
GL-031	Unknown	Non-administrators can use the GET /api/admin/files API endpoint to list files that were uploaded by administrators, leaking meta information and file content to low-privilege users.	Moderate
GL-027	Unknown	Exception logs are delivered to the configured administrative email addresses. If enable_developers_exception_notification is enabled, they are also delivered to exceptions@globaleaks.org.	Moderate
GL-026	Account Takeover	A logged-in user can always disable 2FA without further authentication (old password or a valid 2FA token).	Moderate
GL-022	Leaked Credentials	Due to shared SMTP server credentials for mail.globaleaks.org it was possible to spoof mail from legitimate aliases of the @globaleaks.org domain.	Moderate
GL-016	Stored XSS	Administrators can use the Site Settings > Files feature to upload arbitrary files.	Moderate
GL-015	Improper Input Sanitization	When a select option of a questionnaire has more than 10 possible answers, the option labels are rendered as trusted HTML, allowing the author to inject HTML into the questionnaire.	Moderate
GL-041	Recursion	Template keywords contained in user-data are rendered by the templating engine.	Low
GL-040	Path Traversal	Report attachment uploads accepting path traversal in flowFilename. GlobaLeaks does not use user-defined file	Low

		names in storage paths, but was found to embed relative paths in report export .zip files.	
GL-033	Missing Hardening	The /api/exception API endpoint allows sending mail with arbitrary content to administrators.	Low
GL-029	Improper Input Sanitization	Missing input validation in support requests allows injection of arbitrary text content into support ticket notifications.	Low
GL-018	Denial Of Service	The Twistedd webserver is outdated and has known (critical) vulnerabilities.	Low
GL-017	Information Disclosure	The version number of the Twisted webserver is transmitted in the server response and allows searching for version specific (publicly known) vulnerabilities.	Low
GL-013	HTML Injection	We found several sinks where site administrators could embed HTML content into the site layout.	Low
GL-012	Information Disclosure	The simplified login exposes registered usernames. This could represent a privacy leak, depending on the threat model.	Low

1.6.1 Findings by Threat Level



1.6.2 Findings by Type



1.7 Summary of Recommendations

ID	Type	Recommendation
GL-050	Missing Hardening	<ul style="list-style-type: none"> Disable service workers using a CSP policy.
GL-049	Information Disclosure	<ul style="list-style-type: none"> Remove exact User-Agent from log messages and alerts. Avoid reflecting user-data through stack traces and error messages.
GL-046	Unknown	<ul style="list-style-type: none"> Do not change the page title dynamically.
GL-024	Information Disclosure	<ul style="list-style-type: none"> Validate that served files are stored in a trusted path.
GL-023	Information Disclosure	<ul style="list-style-type: none"> Disable email notifications until an administrator enables them. Consider enforcing the use of an external SMTP server. Log password resets in audit logs so administrators can review when resets were issued.
GL-014	Stored XSS	<ul style="list-style-type: none"> Reduce file upload capabilities. Consider removing or further restricting site customization. Allow disabling script injection by administrators in the global configuration.
GL-008	Exposed Encryption Private Keys	<ul style="list-style-type: none"> Only allow the host system administrator to download certificates. Prevent downloading of private key files for any administrator.
GL-007	Account Takeover	<ul style="list-style-type: none"> Require password to show Account recovery key. Require password to change email address. Require valid 2FA for password-reset (when enabled). Let administrators confirm 2FA reset request issued by a user.
GL-006	Local Privilege Escalation	<ul style="list-style-type: none"> Skip symbolic links and files named ' * ' when assigning ownership and permissions.

		<ul style="list-style-type: none"> • Open the file before path validation and change its ownership with <code>os.fchown()</code> on the already-open file descriptor. • Use <code>os.fwalk()</code> to read <code>fileno</code> and <code>dir_fd</code>. • Let GlobalLeaks check ownership and permissions on startup and fail with a descriptive error message if they need to be changed by an administrator.
GL-004	Subdomain Takeover	<ul style="list-style-type: none"> • Separate hosts and domains per service. • Document the risk for shared services.
GL-032	DoS Protection Bypass	<ul style="list-style-type: none"> • Omit query parameters from URL matching. • Check that the URL path fully matches <code>^/api/token\$</code>.
GL-036	Exposed Service	<ul style="list-style-type: none"> • Listen only on localhost when in development mode.
GL-031	Unknown	<ul style="list-style-type: none"> • Make the <code>/api/admin/files</code> route available to administrators only.
GL-027	Unknown	<ul style="list-style-type: none"> • Default to options that protect privacy. • Require opt-in for software exception reporting.
GL-026	Account Takeover	<ul style="list-style-type: none"> • Ask the user for a 2FA token or recovery key before disabling 2FA. • Administrators should be able to override users 2FA settings in order to support users who forgot their credentials.
GL-022	Leaked Credentials	<ul style="list-style-type: none"> • Revoke the credentials. • Consider a dedicated domain for public SMTP instances. • Consider removing the public SMTP instance.
GL-016	Stored XSS	<ul style="list-style-type: none"> • Consider preventing upload of the <code>script</code> file. • Consider adding <code>Content-Disposition: attachment;</code> for further risk reduction.
GL-015	Improper Input Sanitization	<ul style="list-style-type: none"> • Consider allowing only plaintext input and deny HTML input.
GL-041	Recursion	<ul style="list-style-type: none"> • Resolve only one pass of template variables.
GL-040	Path Traversal	<ul style="list-style-type: none"> • Do not allow relative paths in uploaded file names. • (Additional) Resolve paths and check their prefix before adding them to an export archive.
GL-033	Missing Hardening	<ul style="list-style-type: none"> • Consider disabling frontend error reporting.
GL-029	Improper Input Sanitization	<ul style="list-style-type: none"> • Sanitize email content. • Consider manually approving copies before sending to user defined recipients.
GL-018	Denial Of Service	<ul style="list-style-type: none"> • Update to the latest version of Twisted.
GL-017	Information Disclosure	<ul style="list-style-type: none"> • Remove the <code>server</code> header.
GL-013	HTML Injection	<ul style="list-style-type: none"> • Consider allowing only plaintext input and deny HTML input.
GL-012	Information Disclosure	<ul style="list-style-type: none"> • Document the privacy implications for the simplified login. • Consider a warning next to the login configuration interface. • Consider deprecating of this feature.

2 Methodology

2.1 Planning

Our general approach during penetration tests is as follows:

1. **Reconnaissance**

We attempt to gather as much information as possible about the target. Reconnaissance can take two forms: active and passive. A passive attack is always the best starting point as this would normally defeat intrusion detection systems and other forms of protection afforded to the app or network. This usually involves trying to discover publicly available information by visiting websites, newsgroups, etc. An active form would be more intrusive, could possibly show up in audit logs and might take the form of a social engineering type of attack.

2. **Enumeration**

We use various fingerprinting tools to determine what hosts are visible on the target network and, more importantly, try to ascertain what services and operating systems they are running. Visible services are researched further to tailor subsequent tests to match.

3. **Scanning**

Vulnerability scanners are used to scan all discovered hosts for known vulnerabilities or weaknesses. The results are analyzed to determine if there are any vulnerabilities that could be exploited to gain access or enhance privileges to target hosts.

4. **Obtaining Access**

We use the results of the scans to assist in attempting to obtain access to target systems and services, or to escalate privileges where access has been obtained (either legitimately through provided credentials, or via vulnerabilities). This may be done surreptitiously (for example to try to evade intrusion detection systems or rate limits) or by more aggressive brute-force methods. This step also consist of manually testing the application against the latest (2017) list of OWASP Top 10 risks. The discovered vulnerabilities from scanning and manual testing are moreover used to further elevate access on the application.

2.2 Risk Classification

Throughout the report, vulnerabilities or risks are labeled and categorized according to the Penetration Testing Execution Standard (PTES). For more information, see: <http://www.pentest-standard.org/index.php/Reporting>

These categories are:

- **Extreme**

Extreme risk of security controls being compromised with the possibility of catastrophic financial/reputational losses occurring as a result.

- **High**
High risk of security controls being compromised with the potential for significant financial/reputational losses occurring as a result.
- **Elevated**
Elevated risk of security controls being compromised with the potential for material financial/reputational losses occurring as a result.
- **Moderate**
Moderate risk of security controls being compromised with the potential for limited financial/reputational losses occurring as a result.
- **Low**
Low risk of security controls being compromised with measurable negative impacts as a result.

3 Reconnaissance and Fingerprinting

We were able to gain information about the software and infrastructure through the following automated scans. Any relevant scan output will be referred to in the findings.

- nmap – <https://nmap.org>
- testssl.sh - <https://testssl.sh/>
- Semgrep - <https://semgrep.dev/>
- Chrome DevTools - <https://developer.chrome.com/docs/devtools/>
- Gobuster - <https://github.com/OJ/gobuster>

4 Findings

We have identified the following issues:

4.1 GL-050 — XSS can install a service worker to compromise in-memory sessions

Vulnerability ID: GL-050

Status: Resolved

Vulnerability type: Missing Hardening

Threat level: Elevated

Description:

GlobalLeaks holds session credentials in memory only, mitigating access through directly loaded resources that trigger XSS. When installing a service worker though, the next session opened by an administrator or receiver might still be compromised by injecting malicious scripts.

Technical description:

This could be the case when a malicious asset (for instance an SVG image) is uploaded and served from the same subdomain, as shown in [GL-016](#) (page 45).

The service worker could modify any JavaScript asset that is served to the client in order to inject malicious code that either leaks the authentication token to an adversary, or directly interacts with the API.

Impact:

Stored XSS from resources with an executable MIME type (HTML, SVG, etc) can compromise in-memory browser sessions by installing a [service worker](#).

Recommendation:

- Disable service workers using a CSP policy.

Update :

By disabling `worker-src` in the CSP header, it is no longer possible to configure a service worker, mitigating persistence through stored or reflected XSS. The `worker-sec` is set implicitly through `default-src 'none'`;

4.2 GL-049 — Frontend error contain sensitive information

Vulnerability ID: GL-049	Status: Resolved
Vulnerability type: Information Disclosure	
Threat level: Elevated	

Description:

Frontend exception alert mails contain sensitive information about the client, such as submitted data and the user-agent string.

Technical description:

The following mail body was received in an exception alert e-mail:

```
Platform: GlobaLeaks-2022
Host: 127.0.0.1 (3mxdlcszxyt44702c7hw6vb6ruitf2qmmnw4uhyshkqc7rnyz3osb2ad.onion)
Version: 4.9.9

URL: /login

User Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/102.0.5005.61 Safari/537.36

Error Message: Error: [$parse:syntax] Syntax Error: Token '<' not a primary expression at column 1
of the expression [<u>Hello</u> world] starting at [<u>Hello</u> World].
https://errors.angularjs.org/1.8.2/$parse/syntax?p0=%3C&p1=not%20a%20primary%20expression&p2=1&p3=
%3Cu%3EHello%3C%2Fu%3E%20World&p4=%3Cu%3EHello%3C%2Fu%3E%20World

Stacktrace:
[
  {
    "columnNumber": 836,
    "lineNumber": 8,
    "fileName": "http://127.0.0.1:8082/js/scripts.min.js",
    "source": "    at http://127.0.0.1:8082/js/scripts.min.js:8:836"
  },
  {
    "columnNumber": 88771,
    "lineNumber": 8,
    "fileName": "http://127.0.0.1:8082/js/scripts.min.js",
    "functionName": "Sa.throwError",
    "source": "    at Sa.throwError (http://127.0.0.1:8082/js/scripts.min.js:8:88771)"
  }
]
```



```
{
  "columnNumber": 86965,
  "lineNumber": 8,
  "fileName": "http://127.0.0.1:8082/js/scripts.min.js",
  "functionName": "Sa.primary",
  "source": "    at Sa.primary (http://127.0.0.1:8082/js/scripts.min.js:8:86965)"
},
{
  "columnNumber": 86476,
  "lineNumber": 8,
  "fileName": "http://127.0.0.1:8082/js/scripts.min.js",
  "functionName": "Sa.unary",
  "source": "    at Sa.unary (http://127.0.0.1:8082/js/scripts.min.js:8:86476)"
},
{
  "columnNumber": 86222,
  "lineNumber": 8,
  "fileName": "http://127.0.0.1:8082/js/scripts.min.js",
  "functionName": "Sa.multiplicative",
  "source": "    at Sa.multiplicative (http://127.0.0.1:8082/js/scripts.min.js:8:86222)"
},
{
  "columnNumber": 86048,
  "lineNumber": 8,
  "fileName": "http://127.0.0.1:8082/js/scripts.min.js",
  "functionName": "Sa.additive",
  "source": "    at Sa.additive (http://127.0.0.1:8082/js/scripts.min.js:8:86048)"
},
{
  "columnNumber": 85882,
  "lineNumber": 8,
  "fileName": "http://127.0.0.1:8082/js/scripts.min.js",
  "functionName": "Sa.relational",
  "source": "    at Sa.relational (http://127.0.0.1:8082/js/scripts.min.js:8:85882)"
},
{
  "columnNumber": 85706,
  "lineNumber": 8,
  "fileName": "http://127.0.0.1:8082/js/scripts.min.js",
  "functionName": "Sa.equality",
  "source": "    at Sa.equality (http://127.0.0.1:8082/js/scripts.min.js:8:85706)"
},
{
  "columnNumber": 85556,
  "lineNumber": 8,
  "fileName": "http://127.0.0.1:8082/js/scripts.min.js",
  "functionName": "Sa.logicalAND",
  "source": "    at Sa.logicalAND (http://127.0.0.1:8082/js/scripts.min.js:8:85556)"
},
{
  "columnNumber": 85402,
  "lineNumber": 8,
  "fileName": "http://127.0.0.1:8082/js/scripts.min.js",
  "functionName": "Sa.logicalOR",
  "source": "    at Sa.logicalOR (http://127.0.0.1:8082/js/scripts.min.js:8:85402)"
}
]
```

The message contains the client's user-agent string, which could be used for fingerprinting and identification:

```
User Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/102.0.5005.61 Safari/537.36
```

Although the user-agent is important to help reproduce issues, an advanced level of paranoia seems appropriate for this platform.

The error message itself contains data that was sent to the server, in this case the string `<u>Hello</u> World`:

```
Error Message: Error: [$parse:syntax] Syntax Error: Token '<' not a primary expression at column 1
of the expression [<u>Hello</u> World] starting at [<u>Hello</u> World].
```

With automatic error reporting to exceptions@globaleaks.org enabled, sensitive information about whistleblower reports might be disclosed to a third party that was not supposed to receive the information.

Impact:

When a JavaScript error occurs in the frontend, an email to the site administrators (and potentially [Globleaks GL-027](#) (page 38)) is sent containing the client's user-agent and data that was submitted to the backend in private.

Recommendation:

- Remove exact User-Agent from log messages and alerts.
- Avoid reflecting user-data through stack traces and error messages.

Update :

After evaluating the benefits and downsides of frontend error reporting the `/api/exception` route has been removed entirely in commit [415aaa58](#), so that there is no longer a risk of leaking this potentially sensitive information.

4.3 GL-046 — Page title discloses access to a report

Vulnerability ID: GL-046

Status: Resolved

Vulnerability type: Unknown

Threat level: Elevated

Description:

The GlobaLeaks frontend application updates the page title based on user activity, so that the last user activity is stored in the browsing history.

Technical description:

After viewing a tip on a demo tenant (on <https://try.globaleaks.org>), the browser history discloses a viewer's last activity. When the last activity was viewing a report, the entry in Firefox browsing history appears as:

```
% firefed -p default-release history

https://foobarbarbar.try.globaleaks.org/#/
  Title:      foobarbarbar - Report
  Last visit: 2022-06-09 10:11:28
  Visits:     1
```

Impact:

Users that visit GlobaLeaks in a non-private browser session leave the name of their last activity stored in the browser history, potentially enabling forensics to identify the given visitors role.

Recommendation:

- Do not change the page title dynamically.

Update :

A - `Report` suffix is no longer added to window titles since commit [58bab15c](#), so the browser history no longer discloses whether a user has submitted or viewed a report.

4.4 GL-024 — SQL Injection would lead to arbitrary file read

Vulnerability ID: GL-024

Status: Resolved

Vulnerability type: Information Disclosure

Threat level: Elevated

Description:

An SQL Injection vulnerability in SQLite would lead to arbitrary file read on the remote server.

Technical description:

The SQLite3 `file` table contains references from uploaded file names to their storage location.

```
sqlite> .schema file
CREATE TABLE file (
  tid INTEGER,
  id TEXT(36) NOT NULL,
  name TEXT NOT NULL,
  PRIMARY KEY (id),
  FOREIGN KEY(tid) REFERENCES tenant (id) ON DELETE CASCADE DEFERRABLE INITIALLY DEFERRED,
  UNIQUE (tid, name)
);
```

When an adversary manages to insert or update any file's `id` parameter, downloading the file through the `/s/<filename>` route reveals the file content:

```
INSERT INTO file (tid, id, name) VALUES (1, '../../../../../tmp/flag.txt', 'flag.txt');
```

```
% echo secret > /tmp/flag.txt
% curl http://127.0.0.1:8082/s/flag.txt
secret
```

This happens because the file handler concatenates the asset storage directory with the file id stored in the database to form an absolute path, and serves the file's contents ([backend/globaleaks/handlers/file.py#L38](#)):

```
path = os.path.abspath(os.path.join(self.state.settings.files_path, id))
```

Impact:

An adversary's ability to control data inserted into the SQLite3 database `file` table can lead to remote disclosure of arbitrary system files.

Recommendation:

- Validate that served files are stored in a trusted path.

Update :

AppArmor is enabled by default and enforced in [8f151f2b](#).

Directory traversal check added in [85daab21](#).

4.5 GL-023 — Self-hosted mail notifications sent through mail.globaleaks.org

Vulnerability ID: GL-023

Status: Resolved

Vulnerability type: Information Disclosure

Threat level: Elevated

Description:

By default, outgoing mails from self-hosted instances of GlobaLeaks are sent through `mail.globaleaks.org`.

Technical description:

The entire outgoing mail traffic of the appliance is sent through the default SMTP remote server, including password reset notifications that allow taking over the instance.

Administrators can change the configuration to their own SMTP server.

Impact:

(Administrator) password-reset emails of self-hosted appliances are sent through GlobaLeaks, so that access to the mail content may grant administrative access to the remote appliance.

Recommendation:

- Disable email notifications until an administrator enables them.
- Consider enforcing the use of an external SMTP server.
- Log password resets in audit logs so administrators can review when resets were issued.

Update :

In response to [GL-022](#) (page 41), commit [a6ae4ab6](#) adds a message in the admin backend that notifies administrators of the implications of using a shared mail server.

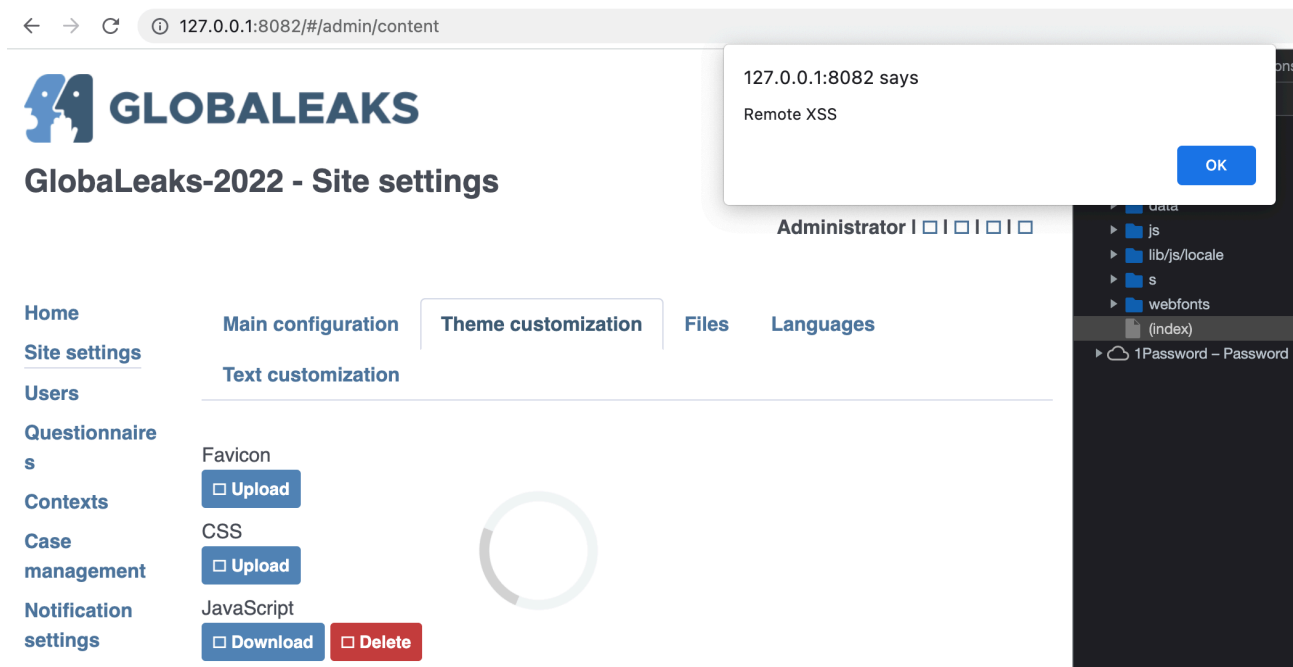
4.6 GL-014 — Administrators Can Upload Arbitrary JavaScript

Vulnerability ID: GL-014	Status: Resolved
Vulnerability type: Stored XSS	
Threat level: Elevated	

Description:

The theme customization feature allows administrators to upload and embed arbitrary JavaScript into the application.

Technical description:



This feature has several implications, as it allows all web interface administrative accounts to take over the application entirely.

Site administrators should not be able to inject script into other users' browser contexts or otherwise elevate their privileges to edit other user passwords that would otherwise only be granted subject to permissions. The permission to reset other account's passwords can be disabled for administrators in advanced settings:

Give this admin ability to change user passwords

It was possible for administrators with the above account restrictions to upload global JavaScript that can affect whistleblower and administrator browser sessions.

Impact:

Administrators can bypass advanced limitations (e.g. set passwords for other users) by injecting JavaScript into other administrators' sessions. Additionally, specialized browser exploits or tracking scripts can be injected at will into the application, as the provided CSP nonce is only enforced for `script.min.js`.

Recommendation:

- Reduce file upload capabilities.
- Consider removing or further restricting site customization.
- Allow disabling script injection by administrators in the global configuration.

Update :

Global frontend JavaScript can no longer be managed by the administrator UI, but requires direct disk access (for instance via SSH or SFTP). This prevents (tenant-) admins without this privileged host access from compromising whistleblowers, other admins, or client browser sessions by injecting malicious JavaScript into the login page. Amongst other changes, the `ScriptHandler` was introduced in commit [07c16546](#).

4.7 GL-008 — Administrators Can Download TLS Private Keys

Vulnerability ID: GL-008

Status: Resolved

Vulnerability type: Exposed Encryption Private Keys

Threat level: Elevated

Description:

The admin UI provides the TLS encryption key-pair (for instance signed by Let's Encrypt). These can be downloaded by any administrative account.

Technical description:

Site administrators are able to manage TLS certificates. In the admin backend they can upload and download certificates and their private keys:

Administrator |  |  |  | 

HTTPS **Tor** **Access control** **URL redirects**

Hostname	127.0.0.1	Save
----------	-----------	-------------

Private Key (PEM)

Download **Delete**

A site administrator is not necessarily a *domain* administrator and should not have access to the private key. When obtained from LetsEncrypt, the key can be re-generated instead of being restored from a backup. In manual configurations, the administrator who uploaded a private key should be the only administrator able to access that key.

Impact:

Any administrator can download private keys. With machine-in-the-middle capabilities, it is possible to intercept or modify all encrypted traffic for the GlobaLeaks instance.

Recommendation:

- Only allow the host system administrator to download certificates.
- Prevent downloading of private key files for any administrator.

Update :

Commit [b160c16a](#) removes the ability to download private keys via the admin API.

4.8 GL-007 — User password reset without old password

Vulnerability ID: GL-007

Status: Resolved

Vulnerability type: Account Takeover

Threat level: Elevated

Description:

It is possible to change another user's account password with access to only their `x-session`, for instance when leaving a browser session unattended or falling victim to an XSS vulnerability.

Technical description:

Steps to reproduce

Step 1: Change account email to an attacker-controlled alias

Navigate to `/#/recipient/preferences` and change the account email to an attacker-controlled one:



GlobaLeaks-2022 - Preferences

Stefan Grönke | | | | |

Home
Reports

Preferences

Password

Username: gronke

Role: Recipient

Name: Stefan Grönke

Public name: Stefan Grönke

Email address:

Enable email notifications

Enable two factor authentication

Set up encryption by providing a PGP public key

Email address:

gronke@radicallyopensecurity.com

Validation of email address change in progress.
Please check your inbox for further instructions.

Step 2: Obtain user's recovery key

Click the "Account recovery key" button and copy the user's recovery key for later use.

The screenshot shows a web browser window with the address bar displaying `127.0.0.1:8082/#/recipient/preferences?src=%2Frecipient%2Fpreferences`. The page title is "GlobaLeaks-". A modal dialog titled "Account recovery key" is centered on the screen. The dialog contains a "Copy to clipboard" button, a key string `J4GC-M3E3-ZHCS-37S5-PPTP-XFCI-C6EC-6CUV-SO4G-3SAQ-H22D-4IVE-56VQ`, and a warning: "Make a copy and store it in a safe place. It will be necessary if you lose your password to recover the access to your account without data loss." Below the dialog, the user's profile information is visible: "Public name: Stefan Grönke" (with an "Edit" button), "Email address: gronke+hacked@radicallyopensecurity.com" (with an "Edit" button), and two checked options: "Enable email notifications" and "Enable two factor authentication". There is a section for "Set up encryption by providing a PGP public key" with a "Select" button. At the bottom of the page, there is a "Save" button and a button labeled "Account recovery key".

Step 3: Confirm new link

Attacker receives the verification email and confirms the password change request:



GlobaLeaks-2022

Your new email address has been validated.

Step 4: Reset password

Attacker requests a password reset on `127.0.0.1:8082/#/login/passwordreset` for the newly activated email address:

A screenshot of a web browser window. The address bar shows the URL `127.0.0.1:8082/#/login/passwordreset`. The page content includes the GLOBALEAKS logo and the heading "GlobaLeaks-2022 - Password reset". Below the heading, there is a text input field containing the email address `gronke+hacked@radicallyopensecurity.com`, a blue "Submit" button, and a small square checkbox to the right of the button. The text above the input field reads: "Enter your account's username or your email address to request a password reset."

Step 5: Enter account recovery key

The password reset link is sent to the attacker controlled email address:

From: GlobaLeaks-2022 <giovanni.pellerano@globaleaks.org>
Subject: GlobaLeaks-2022 - Password reset instructions
Date: 4. June 2022 at 18:06
To: gronke+hacked@radicallyopensecurity.com

Dear Stefan Grönke,

You're receiving this email because a password reset has been requested for the account: gronke

If you didn't make this request, you may safely ignore and delete this email.

You can confirm your request by clicking the following link:

<http://127.0.0.1/#/password/reset?token=7ff6624b73ffd8d7ab54f89c34b94258f78b3d361f33c1970b89dbfcc083524e>

Kind regards,
GlobaLeaks-2022

Here the adversary is asked to enter the "Account recovery key" obtained in step 2:

← → ↻ ⓘ 127.0.0.1:8082/#/password/reset/recovery?token=7ff6624b73ffd8d7ab54f89c34b94258f78b3d361f33c1970b89dbfcc083524e



GlobaLeaks-2022 - Password reset

Enter your account recovery key to complete the password reset procedure



Step 6: Set new password

The adversary can follow the link and set an arbitrary password without entering the old password:



GlobaLeaks-2022 - Change your password

Before proceeding, please set a new password.

New password *

I-was-hacked-2022

Strong

Type your new password again *

.....

Proceed □

Impact:

With knowledge of a valid `x-session` credential (like an unattended browser window), an adversary can take over an account completely, including bypassing 2FA.

Recommendation:

- Require password to show Account recovery key.
- Require password to change email address.
- Require valid 2FA for password-reset (when enabled).
- Let administrators confirm 2FA reset request issued by a user.

Update :

With commit [ed2216fe](#), a valid 2FA token is always required before resetting an account or viewing the account recovery key (if the user has enabled this feature).

4.9 GL-006 — Possible Privilege Escalation Through Filesystem Changes

Vulnerability ID: GL-006	Status: Resolved
Vulnerability type: Local Privilege Escalation	
Threat level: Elevated	

Description:

Before the GlobaLeaks daemon drops privileges, a cleanup task assigns filesystem ownership to the executing user, but recurses over symbolic links.

Technical description:

During startup the GlobaLeaks backend runs a cleanup task that updates `workdir` and `ramdisk` filesystem ownership and permissions to match the executing user as seen in [backend/globaleaks/backend.py#L92-L96](#):

```
fix_file_permissions(Settings.working_path,
                    Settings.uid,
                    Settings.gid,
                    0o700,
                    0o600)
```

A modified version of `fix_file_permissions()` prints the affected directories

```
import os.path
import glob
def fix_file_permissions(path, uid, gid, dchmod, fchmod):
    """
    Recursively fix file permissions on a given path
    """
    exceptions = ['lost+found']
    def fix(path):
        if os.path.basename(path) in exceptions:
            return
        print(f"chown {uid}:{gid} {path}")
        #os.chown(path, uid, gid)
        if os.path.isfile(path):
            print(f"chmod 600 {path}")
            #os.chmod(path, 0o600)
        else:
            print(f"chmod 600 {path}")
            #os.chmod(path, 0o700)
```

```
fix(path)
for item in glob.glob(path + '/*'):
    fix_file_permissions(item, uid, gid, dchmod, fchmod)

fix_file_permissions("/tmp/demo/userdir", 1000, 1000, None, None)
```

For demonstration we create a secret file `/tmp/demo/etc/shadow` and a `workdir` that the user GlobalLeaks drops privilege to can control:

```
% mkdir -p /tmp/demo/etc
% echo "secret" > /tmp/demo/etc/shadow
% chown root: /tmp/demo/etc/shadow
% chmod 600 /tmp/demo/etc/shadow

% mkdir /tmp/demo/workdir
% ln -s /tmp/demo/etc /tmp/demo/workdir/etc
```

On execution of the modified `fix_file_permissions` function, the protected file is made accessible to the target user (1000):

```
% python3 simulate.py
chown 1000:1000 /tmp/demo/workdir
chmod 600 /tmp/demo/workdir
chown 1000:1000 /tmp/demo/workdir/etc
chmod 600 /tmp/demo/workdir/etc
chown 1000:1000 /tmp/demo/workdir/etc/shadow
chmod 600 /tmp/demo/workdir/etc/shadow
```

An adversary would need to wait for the daemon to restart to become owner of these files.

Impact:

Users able to write to the GlobalLeaks backend `workdir` can elevate their privileges by taking ownership of protected system files.

Recommendation:

- Skip symbolic links and files named `'*'` when assigning ownership and permissions.
- Open the file before path validation and change its ownership with `os.fchown()` on the already-open file descriptor.
- Use `os.fwalk()` to read `fileno` and `dir_fd`.
- Let GlobalLeaks check ownership and permissions on startup and fail with a descriptive error message if they need to be changed by an administrator.

Update :

The `fix_file_permission` method was removed in commit [11418804](#), so a malicious actor can no longer gain root privileges by symlinking to system files such as `/etc/sudoers`, `/etc/shadow`, etc.

4.10 GL-004 — Network Settings Allow Arbitrary Host Name

Vulnerability ID: GL-004

Status: Resolved

Vulnerability type: Subdomain Takeover

Threat level: Elevated

Description:

Administrators can configure arbitrary host names as a custom domain name, allowing them to generate and download certificates for all (sub-)domains configured on the same IP address.

Technical description:

Administrators are able to configure arbitrary hostnames that are used as subdomain of `try.globaleaks.org`, allowing them to claim certificates for existing appliances.

This is a demo platform, please do not use it for real submissions.

project logo

foobarbarbar - Network settings

pen"pen" tester | | | |

- Home
- Site settings
- Users
- Questionnaires
- Contexts
- Case management
- Notification settings
- Network settings**
- Advanced settings
- Audit log

HTTPS
 Tor
 Access control
 URL redirects

Hostname: foobarbarbar2.try.globaleaks.org Save

Private Key (PEM) Download Delete

Certificate (PEM) Valid until: 01-09-2022 16:03 Issuer: Let's Encrypt Download Delete

Intermediate Certificates (PEM) Valid until: 15-09-2025 18:00 Issuer: Internet Security Research Group Download Delete

Enable Reset

This affects the demo environment (<https://try.globaleaks.org>) and all other hosts, where multiple services are provided from the same host, for example where the host's IP address matches multiple (sub-)domains.

In combination with [GL-008](#) (page 23), the generated key and certificate can be downloaded to spoof or intercept traffic to the subdomain.

Impact:

Adversaries with machine-in-the-middle capabilities and access to these certificates can intercept or modify TLS encrypted traffic from other services on the same domain or host.

Recommendation:

- Separate hosts and domains per service.

- Document the risk for shared services.

Update :

With commit [a811b094](#), GlobalLeaks only allows root tenants to configure subdomains, so that subordinate tenant administrators can no longer claim subdomains they should not have access to.

4.11 GL-032 — Session/Token Decorator can be bypassed

Vulnerability ID: GL-032	Status: Resolved
Vulnerability type: DoS Protection Bypass	
Threat level: Moderate	

Description:

The request decorator requiring either a request session or token can be bypassed by appending `/api/token` to any API URL.

Technical description:

The `decorator_require_session_or_token` request decorator verifies whether either a token or session property was set.

```
def decorator_require_session_or_token(f):
    # Decorator that ensures a token or a session is included in the request
    def wrapper(self, *args, **kwargs):
        if not self.request.uri.endswith(b"/api/token") and not self.token and not self.session:
            raise errors.InternalServerError("Invalid request: No token and no session")

        return f(self, *args, **kwargs)

    return wrapper
```

It does not perform this test when the URL ends with `/api/token`. Because the URL query parameters are considered part of the URL, an adversary can craft one that ends with the given suffix without changing the URL path. For example:

```
/api/exception?/api/token
```

This behavior was confirmed in finding [GL-033](#) (page 55).

Impact:

API endpoints that require authentication but do not access the `X-Session` are vulnerable to DoS-protection bypass based on user accounts.

Recommendation:

- Omit query parameters from URL matching.
- Check that the URL path fully matches `^/api/token$`.

Update :

Fixed in [2dd9755a](#) by matching the full URL for `/api/token`.

4.12 GL-036 — Development mode listens on all interfaces

Vulnerability ID: GL-036	Status: Resolved
Vulnerability type: Exposed Service	
Threat level: Moderate	

Description:

When the GlobaLeaks daemon is run in development mode, it exposes HTTP services on port 8080 on all available interfaces.

Technical description:

When launching the GlobaLeaks backend daemon in development mode (`-z`) and optionally in the foreground (`-n`), the application listens on all available interfaces.

```
globaleaks -z -n
```

This exposes a local developer's instance to public networks, including the wizard page right after activation.

```
% ss -tulpn | grep globaleaks
tcp LISTEN 0      1024      127.0.0.1:8082      0.0.0.0:*    users:
(("globaleaks",pid=16851,fd=13),("globaleaks",pid=16851,fd=7))
tcp LISTEN 0      1024      127.0.0.1:8083      0.0.0.0:*    users:
(("globaleaks",pid=16851,fd=15),("globaleaks",pid=16851,fd=8))
tcp LISTEN 0      1024          *:8443             *:*         users:
(("globaleaks",pid=16851,fd=17),("globaleaks",pid=16851,fd=10))
```

```
tcp LISTEN 0 1024 *:8080 *:* users:
(("globaleaks",pid=16851,fd=16),("globaleaks",pid=16851,fd=9))
```

Impact:

GlobaLeaks is accessible on public interfaces in development mode by default. Adversaries may gain administrative privilege through the wizard or access information they are not supposed to see. Vulnerabilities in GlobaLeaks could lead to compromise of the developer's system.

Recommendation:

- Listen only on localhost when in development mode.

Update :

In developer mode the server no longer binds to remote interfaces. This change was introduced in commit [f684373a](#).

4.13 GL-031 — Admin API file listing is available to regular users

Vulnerability ID: GL-031

Status: Resolved

Vulnerability type: Unknown

Threat level: Moderate

Description:

Non-administrators can use the `GET /api/admin/files` API endpoint to list files that were uploaded by administrators, leaking meta information and file content to low-privilege users.

Technical description:

It was possible to obtain a list of files uploaded by (tenant) administrators using a low-privilege user's (whistleblower or recipient) session credential:

```
const lowPrivilegedUserSession = "31abba9361c8f4dcd1e9e9142df7f1b486e12828f7291180de463b0f954e2ea7";
fetch("/api/admin/files", {
  headers: {
    "X-Session": lowPrivilegedUserSession
  }
}).then(res => res.text()).then(console.log)
```

The response contains a JSON list of file IDs and original file names, even if the files are not referenced publicly.

```
[
  {"id": "../../../../../../../../tmp/flag.txt", "name": "abc.txt"},
  {"id": "87f68c0a-e241-4d00-ba17-a1d41eadfb70", "name": "xss.js"}
]
```

Files uploaded by administrators for internal sharing or later publication might be found by users that should not (yet) have access to them.

Impact:

Users can list all administrators file uploads, even if they are not yet referenced anywhere and are not intended for public access.

Recommendation:

- Make the `/api/admin/files` route available to administrators only.

Update :

An access permission check in commit [15963bcd](#) prevents listing files for users without administrative privileges.

4.14 GL-027 — Exception Emails Are Sent To Globaleaks.org

Vulnerability ID: GL-027

Status: Resolved

Vulnerability type: Unknown

Threat level: Moderate

Description:

Exception logs are delivered to the configured administrative email addresses. If `enable_developers_exception_notification` is enabled, they are also delivered to `exceptions@globaleaks.org`.

Technical description:

The exception mail scheduler sends a copy of the error notification to `exceptions@globaleaks.org` (in [backend/globaleaks/state.py#L229](#)):

```
mail_subject = "Globleaks Exception"
    delivery_list = self.tenants[1].cache.notification.admin_list + \
        self.tenants[tid].cache.notification.admin_list

    if self.tenants[1].cache.enable_developers_exception_notification:
        delivery_list.append(('exceptions@globaleaks.org', ''))
```

The email address is hard-coded and leaks exception details to the GlobalLeaks mailbox.

During the Setup Wizard this option is enabled by default (in [backend/globaleaks/handlers/signup.py#L141](#)):

```
wizard = {
    'node_language': signup.language,
    'node_name': node_name,
    'admin_username': 'admin',
    'admin_name': signup.name + ' ' + signup.surname,
    # ...
    'enable_developers_exception_notification': True
}
```

Globally the option is enabled by default (in [backend/globaleaks/models/config_desc.py#L52](#)):

```
ConfigDescriptor = {
    # ...
    'enable_developers_exception_notification': Bool(default=True),
    # ...
}
```

An example of a notification mail arriving would also be sent to `exceptions@globaleaks.org`, disclosing the locally hosted appliance Tor hidden service URL and IP address to a central entity. The real IP address of the machine is known to the GlobalLeaks mail server when connecting to the default SMTP server [GL-023](#) (page 21).

Impact:

Self-hosted appliances report copies of exception notification emails to `exceptions@globaleaks.org`. This discloses the (tor) domain or IP address of the service to GlobalLeaks. See [GL-033](#) (page 55) for reference.

Recommendation:

- Default to options that protect privacy.
- Require opt-in for software exception reporting.

Update :

The feature has been disabled by default in [d1c5b6ca](#), so that administrators have to opt in to the feature in safe environments.

4.15 GL-026 — 2FA can be disabled

Vulnerability ID: GL-026	Status: Resolved
Vulnerability type: Account Takeover	
Threat level: Moderate	

Description:

A logged-in user can always disable 2FA without further authentication (old password or a valid 2FA token).

Technical description:

Username: gronke

Role: Recipient

Name: Stefan Grönke

[Edit](#)

Public name: Stefan Grönke

[Edit](#)

Email address:

gronke+hacked@radicallyopensecurity.com

[Edit](#)

Enable email notifications

Enable two factor authentication

Set up encryption by providing a PGP public key

[Select](#)

[Save](#)

[Account recovery key](#)

When enabled, the "Enable two factor authentication" checkbox can be disabled without additional credentials, such as the old password, a valid 2FA secret, or the account recovery key.

With this vulnerability an adversary can immediately disable 2FA on a user's account that was compromised through XSS or an unattended browser session [GL-007](#) (page 25).

Because a user session does not become active until a valid 2FA code is entered, it was not possible to disable 2FA with known username and password only.

Impact:

XSS in a user session can lead to account compromise by disabling 2FA without further input.

Recommendation:

- Ask the user for a 2FA token or recovery key before disabling 2FA.
- Administrators should be able to override users 2FA settings in order to support users who forgot their credentials.

Update :

Disabling 2FA now requires entry of a valid 2FA credential belonging to the user performing the action, which is either the user themselves or an administrator disabling the second factor for another account. This measure was introduced in commit [0f00457a](#).

4.16 GL-022 — E-Mail Spoofing For Globaleaks.org

Vulnerability ID: GL-022

Status: Resolved

Vulnerability type: Leaked Credentials

Threat level: Moderate

Description:

Due to shared SMTP server credentials for `mail.globaleaks.org` it was possible to spoof mail from legitimate aliases of the `@globaleaks.org` domain.

Technical description:

Steps to Reproduce

1. Install GlobaLeaks locally
2. Set Admin > Site Setting > Page title to "Giovanni Pellerano"

3. Set Admin > Notification settings > SMTP email address to `giovanni.pellerano@globaleaks.org`
4. Edit notification E-Mail template `admin_test_mail_template`
5. Set own admin account E-Mail to recipient of spoofed mail
6. Send test mail

← → ↻ ⓘ 127.0.0.1:8082/#/admin/notifications



Giovanni Pellerano - Notification settings

Home

Site settings

Users

Questionnaires

Contexts

Case management

Notification settings

Network settings

Advanced settings

Audit log

Main configuration

Notification templates

Template

admin_test_mail_template

This email appears to be sent by Giovanni while in fact it comes from some hackers!

Regards,
ROS

Save

← → ↻ ⓘ 127.0.0.1:8082/#/admin/notifications

Users**Questionnaires****Contexts****Case management****Notification settings****Network settings****Advanced settings****Audit log**

SMTP email address

giovanni.pellerano@globaleaks.org

SMTP server address

mail.globaleaks.org

SMTP server port

587

Security

SMTP/TLS

 Require authentication

Username

globaleaks

Password

.....

 Disable notifications to administrators Disable notifications to recipients

Number of hours before sending a report expiration alert

72

 Save Test the configuration Reset SMTP configuration

Result

From: Giovanni Pellerano <giovanni.pellerano@globaleaks.org>
Subject: Giovanni Pellerano - Test email
Date: 4. June 2022 at 16:05
To: gronke+admin@radicallyopensecurity.com

This email appears to be sent by Giovanni while in fact it comes from some hackers!

Regards,
ROS

The mail headers show a valid DKIM signature for @globaleaks.org and a match for the SPF record.

```
Return-Path: <giovanni.pellerano@globaleaks.org>
Delivered-To: gronke@radicallyopensecurity.com
Received: from mail.radicallyopensecurity.com
  by mail.radicallyopensecurity.com with LMTP
  id aJ9AJp9mm2LKaAEAK09+lg
  (envelope-from <giovanni.pellerano@globaleaks.org>)
  for <gronke@radicallyopensecurity.com>; Sat, 04 Jun 2022 14:05:19 +0000
Received: from mail.globaleaks.org (mail.globaleaks.org [95.174.23.113])
  by mail.radicallyopensecurity.com (Postfix) with ESMTMP id 4LFhNC3gMZzD1D1
  for <gronke+admin@radicallyopensecurity.com>; Sat, 4 Jun 2022 14:05:15 +0000 (UTC)
Received: from mail.globaleaks.org (localhost [127.0.0.1])
  by mail-relay.globaleaks.org (Postfix) with ESMTMP id 0C9364060F
  for <gronke+admin@radicallyopensecurity.com>; Sat, 4 Jun 2022 16:05:08 +0200 (CEST)
Content-Type: multipart/alternative;
  boundary="=====  
DKIM-Signature: v=1; a=rsa-sha256; c=relaxed/simple; d=globaleaks.org;
  s=globaleaks; t=1654351507;
  bh=c6MoipIk6wnQwCn0WRfu5nQGEfRTpHJ9WlM04eHuu1I=;
  h=Subject:Date:To:From:From;
  b=bZacjUlvfS1lKqNXZAE1rk9v3CiZqg8lIG1WvyHfraAVIF/JBECU2ma9hC20go9kk
  CarG5CbUPMDN9oP66qy6n9DAUxdITzQr9v0zb0kFicLqBBXhgNxKTFioD1N3gZfXSn
  xjqlulukYdmjiDDJdg30NLjsnszuvCCw8d/uYKkamJYI2elmnQeSfeD70SvLByNU1Xz+
  dQzc97UJq2MgzUndd8f5xTTPads5uY0dtl0lsQKDGwHq/qj8im4KseyAoPKMqb557
  SftZz6dhGyHaB0619PQRD89k85mbwiXqM7EwhFEP1pDfpRHaPs+ijczgxbUDD3Ry4y
  dAxU26z6JKFYQ==
MIME-Version: 1.0
Message-Id: <20220604140506.12315.964815977.15@b'GlobaLeaks'>
Subject: =?utf-8?q?Giovanni_Pellerano_-_Test_email?=-
Date: Sat, 04 Jun 2022 14:05:06 -0000
To: =?utf-8?q?gronke+admin=40radicallyopensecurity=2Ecom?=-
  <gronke+admin@radicallyopensecurity.com>
From: =?utf-8?q?Giovanni_Pellerano?=- <giovanni.pellerano@globaleaks.org>
X-Mailer: fnord
Authentication-Results: mail.radicallyopensecurity.com;
  dkim=pass header.d=globaleaks.org header.s=globaleaks header.b=bZacjUlv;
  dmarc=pass (policy=none) header.from=globaleaks.org;
  spf=pass (mail.radicallyopensecurity.com: domain of giovanni.pellerano@globaleaks.org designates
  95.174.23.113 as permitted sender) smtp.mailfrom=giovanni.pellerano@globaleaks.org
X-Spam-Bar: ++++
```

```
% dig TXT +short globaleaks.org
"t-verify=3b150ef586a984695708822fe9114c97"
"google-site-verification=_xdsVTsCLcDm8c3zVd-GXH_76IVbyoAHu-gLf6JIC74"
"v=spf1 include:mail.globaleaks.org include:_spf.google.com ~all"
```

Impact:

Anyone with access to the leaked credentials can impersonate a GlobaLeaks sender. As the email is properly signed and sent from the correct server it is nearly impossible to distinguish between phishing or legitimate emails. The only indication could be the `X-Mailer: fnord` header.

Recommendation:

- Revoke the credentials.
- Consider a dedicated domain for public SMTP instances.
- Consider removing the public SMTP instance.

Update :

In response to this finding, commit [a6ae4ab6](#) introduces a message in the admin backend that notifies administrators of the implications of using a shared mail server. Additionally, the GlobaLeaks MTA was configured to restrict accepted sender addresses.

4.17 GL-016 — Arbitrary File Upload As Administrator

Vulnerability ID: GL-016

Status: Resolved

Vulnerability type: Stored XSS


Threat level: Moderate

Description:

Administrators can use the `Site Settings > Files` feature to upload arbitrary files.

Technical description:

← → ↻ 127.0.0.1:8082/#/admin/content

Administrator | □ | □ | □ | □

GlobaLeaks-2022 - Site settings

- Home
- Site settings
- Users
- Questionnaires
- Contexts
- Case management
- Notification settings
- Network settings
- Advanced settings
- Audit log

Main configuration Theme customization **Files** Languages Text customization

Files

xss.js	<input type="button" value="Download"/>	<input type="button" value="Delete"/>
--------	---	---------------------------------------

```
% curl -s -i http://127.0.0.1:8082/s/xss.js
HTTP/1.1 200 OK
Transfer-Encoding: chunked
Server: GlobaLeaks
Date: Sat, 04 Jun 2022 12:18:37 GMT
Content-Security-Policy: base-uri 'none';default-src 'self';style-src 'self' 'sha256-
fwoy2zCGlh85Nfn4rQUlpLM7MB5cry/1AEDA/G9mQJ8=';script-src 'self' 'sha256-IYBZitj/
YwbzjFFnwLPjJJmMGdSj923kzu2tdCxLKdU=';img-src 'self' data:;font-src 'self' data:;form-action
'self';frame-ancestors 'none';
Cross-Origin-Embedder-Policy: require-corp
Cross-Origin-Opener-Policy: same-origin
Cross-Origin-Resource-Policy: same-site
Permissions-Policy: camera=(),document-domain=(),fullscreen=(),geolocation=(),microphone=()
X-Frame-Options: deny
X-Content-Type-Options: nosniff
Cache-Control: no-store
Referrer-Policy: no-referrer
X-Robots-Tag: noarchive
X-Check-Tor: False
Content-Language: en
Content-Type: application/octet-stream

alert("Remote XSS");
```

The server sets the Content-Type: application/octet-stream header, preventing execution of scriptable content.

Uploading a file with the name `script` will lead to execution of the file on every page load. This is the same functionality as [GL-014](#) (page 22).

Impact:

Administrators can upload arbitrary files served from `https://domain/s/...`, which are publicly available. The impact is broadly limited due to the `Content-Type: application/octet-stream` header.

Recommendation:

- Consider preventing upload of the `script` file.
- Consider adding `Content-Disposition: attachment;` for further risk reduction.

Update :

An allow-list of non-executable MIME types was added in [7c352e1b](#). In addition to this, service workers as seen in [GL-050](#) (page 15) have been blocked.

4.18 GL-015 — HTML Injection in Questionnaire Options Label

Vulnerability ID: GL-015

Status: Resolved

Vulnerability type: Improper Input Sanitization

Threat level: Moderate

Description:

When a select option of a questionnaire has more than 10 possible answers, the option labels are rendered as trusted HTML, allowing the author to inject HTML into the questionnaire.

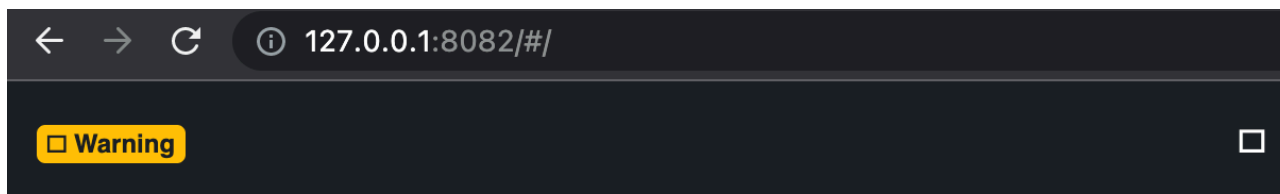
Technical description:

Under the condition that a questionnaire select input has more than 10 possible answers, the option labels are rendered as trusted HTML in `client/app/views/whistleblower/form_field_input.html#L37`, leading to HTML injection

Options **+ Add**

1	HTML <u><u>injection</u></u> works	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
2	Sanitized <script src="http://127.0.0.	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
3	Also Sanitized <script>alert(1)</scrip	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
4	Embed <embed type="text/plain" src	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
5	Object SVG <object type="image/sv	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
6	SVG <svg width="400" height="110"	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
7	Some Label	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
8	Some Label	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
9	Some Label	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
10	Some Label	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
11	Some Label	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Script elements, as seen above, are filtered out. Unsafe-inline would be blocked by the Content-Security Policy, although it could be bypassed with the admin file upload feature [GL-016](#) (page 45). The general HTML injection can be confirmed with underlined text visible in the rendered output:



GlobaLeaks-2022

Can this be hacked? *

Search
HTML <u>injection</u> works
Sanitized
Also Sanitized
Embed
Object SVG
SVG
Some Label

Impact:

Once `ng-sanitizer` bypasses are discovered and publicly known, administrators could embed arbitrary JavaScript into questionnaires. This also requires [GL-014](#) (page 22) to host the JavaScript.

Recommendation:

- Consider allowing only plaintext input and deny HTML input.

Update :

In commit [e404932b](#), the `ng-bind-html` directive was changed to `ng-bind` and no longer allows injection of HTML content.

4.19 GL-041 — Template keywords from user-data resolve

Vulnerability ID: GL-041	Status: Resolved
Vulnerability type: Recursion	
Threat level: Low	

Description:

Template keywords contained in user-data are rendered by the templating engine.

Technical description:

Users can include templating keywords (like `{SubmissionDate}`) in [backend/globaleaks/utils/templating.py#L43](#). We demonstrate this behavior by including the `{SubmissionDate}` string in a report label (that is normally inserted as `{TipLabel}`):



GlobaLeaks-2022 - Report



x{SubmissionDate}y				
#	Context	Date	Last update	Expiration date
1	Default	06-06-2022 17:08	06-06-2022 17:27	05-09-2022 02:00

The label `x{SubmissionDate}y` is rendered into the `client/app/data_src/txt/export_template.txt#L3`:

```
ID: {TipNum}
Date: {SubmissionDate}
Label: {TipLabel}
Status: {TipStatus}

{QuestionnaireAnswers}
{Comments}
{Messages}
```

The result (when downloaded) shows that `{SubmissionDate}` in the `{TipLabel}` is indeed replaced:

```
ID: 1
Date: Monday 06 June 2022 15:08 (UTC)
Label: xMonday 06 June 2022 15:08 (UTC)y
Status: Opened
...
```

The template keywords are also replaced when whistleblowers send messages to recipients:



GlobaLeaks-2022 - Report



#	Context	Date
2	Default	08-06-2022 19:40

Questionnaire answers

How much is the fish?
The fish is {SubmissionDate}!

Private messages:

From: Whistleblower
Date: Wednesday 08 June 2022 17:43 (UTC)

Hello Wednesday 08 June 2022 17:43 (UTC)!

Keywords available in each template are scoped for each template individually, so the opportunities to use this for exploitation are very limited. Regardless, it does not seem appropriate that non-administrators (like whistleblowers) can make use of templating keywords in their message content.

Impact:

Template variable content can contain nested variables that are resolved by the template renderer. The template only resolves known variables in the according scope, which severely limits the impact.

Recommendation:

- Resolve only one pass of template variables.

Update :

With commit [f76b09b6](#) template keywords are no longer replaced recursively. Instead, curly braces around keywords are replaced with round ones, avoiding the recursion.

4.20 GL-040 — Export zip file path traversal

Vulnerability ID: GL-040

Status: Resolved

Vulnerability type: Path Traversal

Threat level: Low

Description:

Report attachment uploads accepting path traversal in `flowFilename`. GlobaLeaks does not use user-defined file names in storage paths, but was found to embed relative paths in report export .zip files.

Technical description:

It was possible to upload report attachments with a user-defined `flowFilename` that can contain relative paths, here demonstrated with an attempt to override `~/ .bashrc`:

```
curl 'http://127.0.0.1:8082/api/rtips/867ef33b-e733-4d90-9403-c50c8fb8895d/wbfile' \  
-H 'Content-Type: multipart/form-data; boundary=----WebKitFormBoundaryQKswabWhbzA9hqeB' \  
-H 'X-Session: f967cb739f12e8d6140a974a21d24d2efdc2a24a4315e3dc0a79495b2855e620' \  
--data-raw '$' \  
-----WebKitFormBoundaryQKswabWhbzA9hqeB
```

```

Content-Disposition: form-data; name="description"

date_2
-----WebKitFormBoundaryQKswabWhbzA9hqeB
Content-Disposition: form-data; name="flowChunkNumber"

1
-----WebKitFormBoundaryQKswabWhbzA9hqeB
Content-Disposition: form-data; name="flowChunkSize"

1024000
-----WebKitFormBoundaryQKswabWhbzA9hqeB
Content-Disposition: form-data; name="flowCurrentChunkSize"

30
-----WebKitFormBoundaryQKswabWhbzA9hqeB
Content-Disposition: form-data; name="flowTotalSize"

30
-----WebKitFormBoundaryQKswabWhbzA9hqeB
Content-Disposition: form-data; name="flowIdentifier"

1012927.6821511965
-----WebKitFormBoundaryQKswabWhbzA9hqeB
Content-Disposition: form-data; name="flowFilename"

../.bashrc
-----WebKitFormBoundaryQKswabWhbzA9hqeB
Content-Disposition: form-data; name="flowRelativePath"

../.bashrc
-----WebKitFormBoundaryQKswabWhbzA9hqeB
Content-Disposition: form-data; name="flowTotalChunks"

1
-----WebKitFormBoundaryQKswabWhbzA9hqeB
Content-Disposition: form-data; name="file"; filename="../.bashrc"
Content-Type: text/plain

Sat Jun  4 19:21:30 CEST 2022\n
-----WebKitFormBoundaryQKswabWhbzA9hqeB-- '

```

```

% unzip -l report.zip
Archive:  report.zip
  Length      Date    Time    Name
-----
   26  06-06-2022  15:27  files/style.css
   21  06-06-2022  15:27  files_attached_from_recipients/xss.js
   30  06-06-2022  15:27  files_attached_from_recipients/date_1.txt
   30  06-06-2022  15:27  files_attached_from_recipients/../.bashrc
  563  06-06-2022  15:27  report.txt
-----
  670
          5 files

```

Notice the `files_attached_from_recipients/../../bashrc` path in the zip archive. GlobaLeaks does not control the downloader's local extraction tool, so the `report.zip` file downloaded in `~/Downloads/report.zip` might create or overwrite the user's `~/.bashrc` (for example).

Impact:

Tip recipients might trust a report export download from GlobaLeaks that exploits zip path traversal in their local extraction tool.

Recommendation:

- Do not allow relative paths in uploaded file names.
- (Additional) Resolve paths and check their prefix before adding them to an export archive.

Update :

Fixed by resolving the files basename in commit [8e8a2a57](#).

4.21 GL-033 — Exceptions API endpoint can be used to send arbitrary mail to administrators

Vulnerability ID: GL-033

Status: Resolved

Vulnerability type: Missing Hardening

Threat level: Low

Description:

The `/api/exception` API endpoint allows sending mail with arbitrary content to administrators.

Technical description:

```
fetch("/api/exception?api/token", {
  method: "POST",
  headers: {
    "Content-Type": "application/json"
  },
  body: JSON.stringify({
    'errorUrl': 'https://www.globaleaks.org/exception',
    'errorMessage': 'EXCEPTION!'
  })
})
```

```
'stackTrace': [],  
'agent': "Antani 1.3.3.7"  
  })  
}).then(res => res.text()).then(console.log)
```

(sent with auth-bypass [GL-032](#) (page 35))

From: GlobaLeaks-2022 <notifications@globaleaks.org>
Subject: GlobaLeaks-2022 - GlobaLeaks Exception
Date: 5. June 2022 at 15:25
To: gronke+admin@radicallyopensecurity.com

Platform: GlobaLeaks-2022
Host: 127.0.0.1 (3mxdlcszxyt447o2c7hw6vb6ruitf2qmmnw4uhyshkqc7rnyz3osb2ad.onion)
Version: 4.9.9

URL: <https://www.globaleaks.org/exception>

User Agent: Antani 1.3.3.7

Error Message: EXCEPTION!

Stacktrace:
[]

Impact:

Administrators may receive valid-looking email alerts with user-controlled content that might be misleading and contain phishing links.

Recommendation:

- Consider disabling frontend error reporting.

Update :

The API endpoint has been removed in [415aaa58](#).

4.22 GL-029 — Mail content injection in Support Ticket request

Vulnerability ID: GL-029

Status: Resolved

Vulnerability type: Improper Input Sanitization

Threat level: Low

Description:

Missing input validation in support requests allows injection of arbitrary text content into support ticket notifications.

Technical description:

The E-Mail address, Text and GlobalLeaks appliance URL in support request tickets can be selected by the submitting user.

Insufficient filtering of the input data can be used to inject content into the mail body. As Proof of Concept the following URL is submitted to the `/api/support` endpoint:

```
https://radicallyopensecurity.com
```

```
+++Hacked+++
```

This can be achieved by using the following cURL command:

```
curl 'http://127.0.0.1:8082/api/support' \  
-H 'Content-Type: application/json;charset=UTF-8' \  
-H 'X-Session: <CENSORED>' \  
--data-raw '{"mail_address":"gronke+admin@radicallyopensecurity.com","text":"Please help","url":"https://radicallyopensecurity.com\n\n+++Hacked+++\\n"}'
```

The above example could be used to lure `gronke+admin@radicallyopensecurity.com` to visit a phishing site. Emails are sent to support recipients as well as to the email address defined in the API request:

From: GlobaLeaks-2022 <notifications@globaleaks.org>
Subject: GlobaLeaks-2022 - Support request
Date: 4. June 2022 at 19:10
To: gronke+admin@radicallyopensecurity.com

From: gronke+admin@radicallyopensecurity.com

Site: <https://radicallyopensecurity.com>

+++Hacked+++

**Request:
Please help**

Newlines after the URL are injected into the text email body without further sanitization.

Impact:

Legitimate-looking support request email messages can be manipulated and used to phish administrators.

Recommendation:

- Sanitize email content.
- Consider manually approving copies before sending to user defined recipients.

Update :

Commit [2a138de0](#) matches the URL string against a regular expression, preventing content injection into the plaintext mail body.

4.23 GL-018 — The Twisted Webserver On try.globaleaks.org Is Outdated And Has Known Vulnerabilities

Vulnerability ID: GL-018

Status: Unresolved

Vulnerability type: Denial Of Service

Threat level: Low

Description:

The Twisted webserver is outdated and has known (critical) vulnerabilities.

Technical description:

<https://github.com/twisted/twisted/security/advisories> The version in use was leaked in [GL-017](#) (page 60) and is known to be `18.9.0`. The critical-rated vulnerability [CVE-2019-9512](#) seems to apply to the GlobalLeaks demo platform, but exploitation was not verified.

Impact

Twisted web servers that utilize the optional HTTP/2 support suffer from the following flow-control related vulnerabilities:

Ping flood: <https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2019-9512>

Reset flood: <https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2019-9514>

Settings flood: <https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2019-9515>

A Twisted web server supports HTTP/2 requests if you've installed the `http2` optional dependency set.

The GlobalLeaks GitHub repository dependencies file references a newer version, which is not affected by the above vulnerabilities.

Impact:

The `try.globaleaks.org` environment and other outdated instances could be taken down with limited resources.

Recommendation:

- Update to the latest version of Twisted.

Update :

Ubuntu APT repositories have patched this vulnerability in [USN-4308-1](#), while there is [no mention of patches on the Debian package](#). Given the low severity of a Denial of Service caused by this repository circumstance, ROS suggested notifying the Debian APT maintainers and taking no further action.

4.24 GL-017 — try.globaleaks.org Leaks Python Webserver Version

Vulnerability ID: GL-017	Status: Resolved
Vulnerability type: Information Disclosure	
Threat level: Low	

Description:

The version number of the Twisted webserver is transmitted in the server response and allows searching for version specific (publicly known) vulnerabilities.

Technical description:

The interface exposes TwistedWeb/18.9.0 as version (stable is 22.04)

```
curl -i -s 'https://try.globaleaks.org/'
```

```
HTTP/2 400
server: TwistedWeb/18.9.0
date: Fri, 03 Jun 2022 12:34:14 GMT
content-length: 0
```

Impact:

An adversary gains the ability to search for version-specific exploits instead of manually fingerprinting the application.

Recommendation:

- Remove the `server` header.

Update :

Server responses no longer disclose the Python Twisted version. Instead, the server identifies as G1obaLeaks:

```
% curl -I -s 'https://try.globaleaks.org/' | grep -i "server:"  
server: G1obaLeaks
```

4.25 GL-013 — Possible HTML Content Injection Sinks

Vulnerability ID: GL-013

Status: Resolved

Vulnerability type: HTML Injection

Threat level: Low

Description:

We found several sinks where site administrators could embed HTML content into the site layout.

Technical description:

```
<div data-markdown-to-html="public.node.disclaimer_text">  
  <div ng-bind-html="trustedHtml">  
    <p></p>  
  </div>  
</div>
```

The custom privacy and solicit question sections render partial HTML tags.

Network settings

HTTPS

Tor

Access control

URL redirects

Tor Onion Service

si6rpo2elwuktb63sfqaujzlw5by736xifxyw56z3q63srhmbh5

- Disable the privacy panel
- Enable custom privacy panel

Custom privacy panel

```
<img src=x onload=alert('roswashere');/>
```



Proceed

Close

Question to solicit possible whistleblowers

```
OK? <img src=x onerror=alert('roswashere question');/>
```

OK?

**File a report" **

Content served with the tag `ng-bind-html` is automatically sanitized but is deprecated, and once bypasses are publicly known stay unfixed. The default content security policy prevents external resource loading.

Impact:

Once `ng-sanitizer` bypasses are discovered and publicly known, administrators could embed arbitrary JavaScript into questionnaires. This also requires [GL-014](#) (page 22) to host the JavaScript.

Recommendation:

- Consider allowing only plaintext input and deny HTML input.

Update :

A `stripHtml` introduced in commit [b80a9d14](#) to the frontend prevents the browser from interpreting HTML content.

4.26 GL-012 — Simplified Login Leaks Registered Usernames

Vulnerability ID: GL-012

Status: Resolved

Vulnerability type: Information Disclosure

Threat level: Low

Description:

The simplified login exposes registered usernames. This could represent a privacy leak, depending on the threat model.

Technical description:

The normal login flow for the GlobaLeaks platform requires knowing a username and password combination. The simplified login mask shows a dropdown list of available login usernames. For successful login, a password and (if configured) a 2FA token is needed. This seems to be intended behavior but could be used to gain information about the people handling the reports.

Impact:

It is possible to obtain the names of users engaged on the platform, which can be (depending on the threat model) lead to privacy-relevant leaks. This information could aid further social engineering attacks.

Recommendation:

- Document the privacy implications for the simplified login.
- Consider a warning next to the login configuration interface.
- Consider deprecating of this feature.

Update :

For convenience of some GlobaLeaks appliances in production use it was decided to keep the Simplified Login feature supported. Instead of disclosing usernames commit [e5f830db](#) changes to listing the user's UUID, which is already public information.

5 Non-Findings

In this section we list some of the things that were tried but turned out to be dead ends.

5.1 NF-044 — Absence of RCE through subprocess

Subprocess execution was not found to be present in the GlobaLeaks backend source code. Third-party dependencies installed through pip were inspected for the potential to execute code through child processes, but this was not found in any active code paths.

5.2 NF-043 — CSRF protection

The GlobaLeaks frontend is protected from CSRF by preventing embedding by setting the HTTP response header `X-Frame-Options: deny` and a Content-Security-Policy header that includes `frame-ancestors 'none'`;

Session credentials are not persisted in the browser but are kept in memory and sent as an `X-Session` HTTP request header to authenticate each request.

5.3 NF-042 — Absence of SQL injection in GlobaLeaks backend

Access to the SQLite3 database is handled through a custom ORM framework based on [SQLAlchemy](#).

Throughout the GlobaLeaks backend no raw SQL queries with user input are executed. If an SQL injection was present it would have to be located in the SQLAlchemy third-party library, as there are no occurrences in the GlobaLeaks source code.

To further mitigate exploitability, GlobaLeaks restricts the available SQL statements and functions in [backend/globaleaks/orm.py#L54-L67](#):

```
def authorizer_callback(action, table, column, sql_location, ignore):
    if action in [SQLITE_DELETE,
                  SQLITE_INSERT,
                  SQLITE_READ,
                  SQLITE_SELECT,
                  SQLITE_TRANSACTION,
                  SQLITE_UPDATE] or \
        (action == SQLITE_FUNCTION and column in ['count',
                                                  'lower',
                                                  'min',
                                                  'max']):
        return sqlite3.SQLITE_OK
    else:
        return sqlite3.SQLITE_DENY
```

5.4 NF-035 — Plain text mail only

The mailer used by GlobaLeaks uses plain text exclusively, so that any potential content injection would be visible. If HTML mails were supported, which they are not, adversaries with the ability to inject content might gain more control of the rendered result. In the current design, fragments of the original template would always be visible (if the client does not render control characters).

5.5 NF-030 — Submission attachments are never stored in plain text

The `/api/submission/attachment` file upload encrypts uploaded files on the fly and never stores them in plain text. On one hand this prevents artifacts on the server disk, but also prevents the use of those files for stored XSS.

5.6 NF-020 — User-Provided HTML Content Is Properly Sanitized

Although `angular-sanitize` is marked as end-of-life and might not be fixed if future vulnerabilities are discovered in it, the mitigation is used in front of `ng-bind-html`. No occurrences of `deliberatelyTrustDangerousSnippet` (that would bypass the sanitizer) were found in the source code.

See also: [https://docs.angularjs.org/api/ngSanitize/service/\\$sanitize](https://docs.angularjs.org/api/ngSanitize/service/$sanitize)

`ngSanitize` follows the [WHATWG standard recommendations for content sanitization](#) which is over-restrictive. Given that AngularJS no longer receives updates, this behavior ensures that new HTML elements and attributes will not expand the attack surface.

ROS cannot entirely rule out eventual flaws in the sanitizer, which is why we recommend strict server-side input validation (in addition to type checks) for user-provided data.

5.7 NF-002 — Properly Configured robots.txt And sitemap.xml

Indexing and bot access for the `https://try.globaleaks.org` environment is sensibly configured.

Search engine crawlers are only allowed to index the index page:

```
% curl https://try.globaleaks.org/robots.txt
User-agent: *
Allow: /$
Disallow: *
Sitemap: https://try.globaleaks.org/sitemap.xml
```

Accordingly the `sitemap.xml` only references the index page as well, but offers different languages:

```
% curl https://try.globaleaks.org/sitemap.xml
<?xml version='1.0' encoding='UTF-8' ?>
```

```
<urlset xmlns='http://www.sitemaps.org/schemas/sitemap/0.9' xmlns:xhtml='http://www.w3.org/1999/
xhtml'>
  <url>
    <loc>https://try.globaleaks.org/#/</loc>
    <changefreq>weekly</changefreq>
    <priority>1.00</priority>
    <xhtml:link rel='alternate' hreflang='am' href='https://try.globaleaks.org/#/?lang=am' />
    <xhtml:link rel='alternate' hreflang='ar' href='https://try.globaleaks.org/#/?lang=ar' />
    <xhtml:link rel='alternate' hreflang='az' href='https://try.globaleaks.org/#/?lang=az' />
    <!-- ... -->
  </url>
</urlset>
```

This configuration prevents accidental indexing of user content and does not disclose any information.

6 Future Work

- **Retest of findings**

When mitigations for the vulnerabilities described in this report have been deployed, a repeat test should be performed to ensure that they are effective and have not introduced other security problems.

- **Regular security assessments**

Security is an ongoing process and not a product, so we advise undertaking regular security assessments and penetration tests, ideally prior to every major release or every quarter.

- **Client-Side Encryption**

GlobaLeaks already encrypts whistleblower tips and assets on the filesystem with a public key mechanism to protect from compromise of server hardware. The data itself is sent to the remote server in clear-text though, which leaves new submissions after infiltration of a server vulnerable to be intercepted.

Modern browsers allow encrypting data and assets in the whistleblowers client and decrypt them at the recipient side instead of funneling through the server (in memory).

In order to harden client-side encryption against script injection through a compromised server, GlobaLeaks frontend code should be delivered through a trusted third-party (for instance by hashing the assets to be loaded by a client).

Combination of client-side encryption and delivery/validation of scriptable frontend code through a third-party would prevent a user with control of the remote hardware or root access to the system from accessing confidential or compromising information.

7 Conclusion

We discovered 0 Critical, 0 High, 10 Elevated, 8 Moderate, and 8 Low-severity issues during this penetration test.

ROS performed a combined audit of the GlobaLeaks source code and a pentest on <https://try.globaleaks.org> in order to find vulnerabilities in the GlobaLeaks platform. One major objective of this exercise was ensuring privacy and plausible deniability for whistleblowers, and to assess the security posture of the GlobaLeaks server and web frontend.

When studying the GlobaLeaks source code, it became obvious that a security-in-depth approach had been used, which contributed to the low severity of findings and number of non-findings we reported. This attitude is much appreciated, considering that a time-boxed pentest is only a one-time snapshot, and even if performed conscientiously can always miss a vulnerability. We do recommend reading the non-finding sections to get an impression of the thoughtfulness put into defense mechanisms. Given the importance of anonymity for whistleblowers and the confidentiality of their submissions we conclude that the tool is well suited for a decentralized infrastructure platform for handling leaks.

While the server-side API based on Python 3 Twisted shows no signs of obsolescence, the AngularJS framework used by the frontend has been deprecated and replaced by a re-write based on newer technology. This does not mean vulnerabilities in HTML sanitization discovered in the future would not be found, but can be considered a technical debt that we recommend addressing in the near future.

Efforts to protect installations from compromise of a server with storage encryption can be modernized by switching to client-side encryption instead, so that whistleblowers' submissions never pass through server-side memory in clear text. This is a recommended step on the project's roadmap.

Thanks to combined efforts with GlobaLeaks we are pleased to publish this report knowing that all the findings we reported have been addressed in release version 4.10.0.

We recommend fixing all of the issues found and then performing a retest in order to ensure that mitigations are effective and that no new vulnerabilities have been introduced.

Finally, we want to emphasize that security is a process – this penetration test is just a one-time snapshot. Security posture must be continuously evaluated and improved. Regular audits and ongoing improvements are essential in order to maintain control of your corporate information security. We hope that this pentest report (and the detailed explanations of our findings) will contribute meaningfully towards that end.

Please don't hesitate to let us know if you have any further questions, or need further clarification on anything in this report.

Appendix 1 Testing team

Tillmann Weidinger	Tillmann Weidinger is a trained full-stack developer with a strong emphasis on security. He started tinkering with computers in his early teens. Due to this he has multiple years of experience in (reverse-)engineering hard- and software, software architecture and breaking things. His main interests evolve around Secure Coding, Automation, Web Applications, WiFi, DMA attacks and other topics between hard- and software with a focus on red-teaming. He enjoys programming in multiple languages and recently has chosen rust as his new favorite. He started studying IT-Security at Ruhr University Bochum and switched to Computer Science at FH Bochum and will graduate in 2022. Due to his broad experience in application development and system's security he can quickly adapt to new IT-Security related topics and is always happy to learn lesser known facts.
Stefan Grönke	Stefan is a highly adaptable senior security consultant, pentester and code auditor. He has over a decade of experience in (reverse) engineering, architecture and quality assurance, with a large focus on security and simplicity. He commits most of his free time to development projects that enable him and others to run secure infrastructure. As a full-stack developer he has always enjoyed learning from and with open source code; Stefan has contributed to a variety of projects, often on GitHub. Stefan can be a terrible chaos monkey in the ROS infra, but always cleans up behind him. In fact he likes constructing more than disruption. Therefore he went over from setting things on fire to participating in the ROS development and infra team. Apart from that he enjoys speaking at conferences like the Chaos Communication Congress or hosting workshops at local hackerspaces. He was one of the winning participants of team proTRon at the Shell Eco Contest in 2013/14 for building a CAN-Bus based telemetry system for a lightweight fuel-cell driven car.
Melanie Rieback	Melanie Rieback is a former Asst. Prof. of Computer Science from the VU, who is also the co-founder/CEO of Radically Open Security.