SUBGRAPH

FINDINGS REPORT - UWAZI

Uwazi Application

May 13, 2019

Prepared for: Vendor

Subgraph Technologies, Inc.
642 Rue de Courcelle, Suite 309
Montreal, Quebec
https://subgraph.com

Overview

Subgraph performed a security audit of the Uwazi application. This audit consisted of the following:

- Automated and manual testing of the application deployed natively and in Docker
- Code auditing of the application, with a focus on issues such input validation vulnerabilities, file upload safety, authentication, and access control

The audit covered version *1.4* of the application as well as more recent versions from the *master* branch from January/February of 2019.

Subgraph discovered *two* security findings as a result of the audit.

Summary

No.	Title	Severity	Remediation
V-001	Insufficient Document Validation	Medium	Resolved
V-002	Arbitrary File Deletion	Medium	Resolved

Details

V-001: Insufficient Document Validation

Severity	Remediation
Medium	Resolved

Discussion

Uwazi does not sufficiently validate documents by their type and mimetype. In effect, Uwazi does not restrict the type of documents that can be uploaded by a user with *editor* privileges. However, some file content (such as HTML and JavaScript) is inherently dangerous because it may be rendered in the user's browser.

Impact Analysis

An attacker with *editor* privileges can cause arbitrary HTML and JavaScript to be rendered in the session of another user of the application. This could be exploited to gain unauthorized access to an admin's session.

To exploit, the issue the attacker will upload a malicious file and set the mimetype to something (such as *text/html*) that will cause it to be rendered (instead of downloaded) by the victim's browser. At this stage, the attacker can entire the admin user to view the malicious document. But there is also a chance that they will view it inadvertently while using the application.

Remediation Recommendations

Do not accept a user-supplied mimetype (*Content-Type*) during document uploads. The mimetype should be determined based on the document content for supported document types such as PDF and set to a safe default (such as *text/plain*) for all other document types.

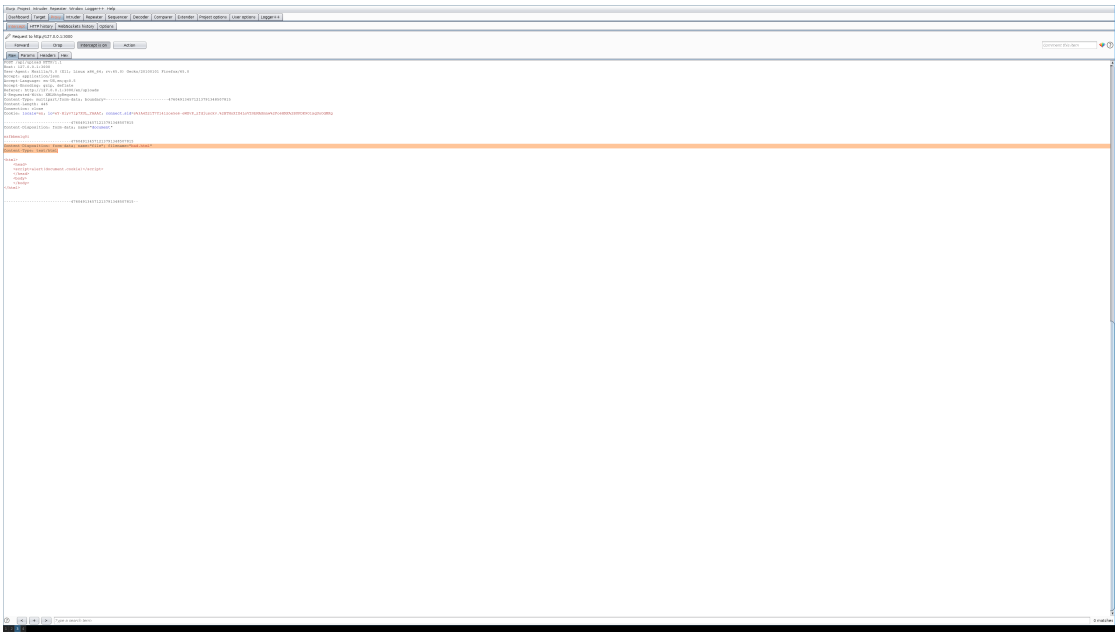
As an additional precaution, consider forbidding the upload of executable files such as *.exe*. These present a danger because users of the site are apt to trust any file hosted by the application – making it more likely that they will run malicious executables hosted there.

Remediation Status

The developers have resolved this issue.

Additional Information

The following screenshot shows the manipulation of a document upload in transit:



The following screenshot shows the browser rendering the document that was uploaded:

V-002: Arbitrary File Deletion

Severity	Remediation
Medium	Resolved

Discussion

Uwazi is prone to a vulnerability that allows users with *editor* privileges to delete files on the host outside of the *uploaded_documents* directory.

This issue is present in the */api/attachments/delete* API call. It is possible to specify a relative path (using directory traversal sequences) to the *filename* argument. As a result, the attacker-specified file will be deleted instead of a valid attachment.

Impact Analysis

An attacker can exploit this issue to delete important application or host files. This can result in a denial-of-service. Any files owned by the web server user can be deleted. In the case of Docker deployments, the web server process runs as *root*. In non-Docker deployments this will likely be a less privileged user – but application files may still be prone to deletion.

Remediation Recommendations

The best to address this issue is for the application to supply the attachment filename from the data model instead of the user. Otherwise, it is possible to implement a check to canonicalize the user-supplied filename and ensure that the canonical path has the same prefix as the configured *uploaded_documents* directory.

Remediation Status

The developers have resolved this issue.

Additional Information

Sending the following proof-of-concept HTTP request as an *editor* will delete the application *README* file:

```
DELETE /api/attachments/delete?entityId=5c64771ececb58001cf0ba2c&
    filename=/../../../home/node/uwazi/README.md
```

For the request to work, the *entityId* field must be valid and the filename must be prefixed with a */* character before the directory traversal sequences.

Appendix

Methodology

Our approach to testing is designed to understand the design, behavior, and security considerations of the assets being tested. This helps us to achieve the best coverage over the duration of the test.

To accomplish this, Subgraph employs automated, manual and custom testing methods. We conduct our automated tests using the industry standard security tools. This may include using multiple tools to test for the same types of issues. We perform manual tests in cases where the automated tools are not adequate or reveal behavior that must be tested manually. Where required, we also develop custom tools to perform tests or reproduce test findings.

The goals of our testing methodology are to:

- Understand the expected behavior and business logic of the assets being tested
- Map out the attack surface
- Understand how authentication, authorization, and other security controls are implemented
- Test for flaws in the security controls based on our understanding
- Test every point of input against a large number of variables and observe the resulting behavior
- Reproduce and re-test findings
- Gather enough supporting information about findings to enable us to classify, report, and suggest remediations

Description of testing activities

Depending on the type and scope of the engagement, our methodology may include any of the following testing activities:

1. **Information Gathering:** Information will be gathered from publicly availble sources to help increase the success of attacks or discover new vulnerabilities
2. **Network discovery:** The networks in scope will be scanned for active, reachable hosts that could be vulnerable to compromise
3. **Host Vulnerability Assessment:** Hosts applications and services will be assessed for known or possible vulnerabilities
4. **Application Exploration:** The application will be explored using manual and automated methods to better understand the attack surface and expected behavior
5. **Session Management:** Session management in web applications will be tested for security flaws that may allow unauthorized access
6. **Authentication System Review:** The authentication system will be reviewed to determine if it can be bypassed

7. **Privilege Escalation:** Privilege escalation checks will be performed to determine if it is possible for an authenticated user to gain access to the privileges assigned to another role or administrator
8. **Input Validation:** Input validation tests will be performed on all endpoints and fields within scope, including tests for injection vulnerabilities (SQL injection, cross-site scripting, command injection, etc.)
9. **Business Logic Review:** Business logic will be reviewed, including attempts to subvert the intended design to cause unexpected behavior or bypass security controls

Reporting

Findings reports are peer-reviewed within Subgraph to produce the highest quality findings. The report includes an itemized list of findings, classified by their severity and remediation status.

Severity ratings

Severity ratings are a metric to help organizations prioritize security findings. The severity ratings we provide are simple by design so that at a high-level they can be understood by different audiences. In lieu of a complex rating system, we quantify the various factors and considerations in the body of the security findings. For example, if there are mitigating factors that would reduce the severity of a vulnerability, the finding will include a description of those mitigations and our reasoning for adjusting the rating.

At an organization's request, we will also provide third-party ratings and classifications. For example, we can analyze the findings to produce *Common Vulnerability Scoring System* (CVSS)[1] scores or *OWASP Top 10*[2] classifications.

The following is a list of the severity ratings we use with some example impacts:

Critical
Exploitation could compromise hosts or highly sensitive information

High
Exploitation could compromise the application or moderately sensitive information

Medium
Exploitation compromises multiple security properties (confidentiality, integrity, or availability)

[1] https://www.first.org/cvss/
[2] https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project

Low

Exploitation compromises a single security property (confidentiality, integrity, or availability)

Informational

Finding does not pose a security risk or represents suspicious behavior that merits further investigation

The severity of a finding is often a product of the impact to general security properties of an application, host, network, or other information system.

These are:

Compromise of confidientiality	Exploitation results in authorized access to data
Compromise of integrity	Exploitation results in the unauthorized modification of data or state
Compromise of availability	Exploitation results in a degradation of performance or an inability to access resources

The actual severity of a finding may be higher or lower depending on a number of other factors that may mitigate or exacerbate it. These include the context of the finding in relation to the organization as well as the likelihood of exploitation. These are described in further detail below.

Contextual factors

Confidentiality, integrity, and availability are one dimension of the potential risk of a security finding. In some cases, we must also consider contextual factors that are unique to the organization and the assets tested.

The following is a list of those factors:

Financial compromise	Exploitation may result in financial losses
Reputation compromise	Exploitation may result in damage to the reputation of the organization
Regulatory liability	Exploitation may expose the organization to regulatory liability (e.g. place them in a state of non-compliance)
Organizational impact	Exploitation may disrupt the operations of the organization

Likelihood

Likelihood measures how probable it is that an attacker exploit a finding.

This is determined by numerous factors, the most influential of which are listed below:

Authentication required	Whether or not the attack must be authenticated
Privilege level	Whether or not an authenticated attacker requires special privileges
Public exploit	Whether or not exploit code is publicly available
Public knowledge	Whether or not the finding is publicly known
Exploit complexity	How complex it is for a skilled attacker to exploit the finding
Local vs. remote	Whether or not the finding is exposed to the network
Accessibility	Whether or not the affected asset is exposed on the public Internet
Discoverability	How easy it is for the finding to be discovered by an attacker
Dependencies	Whether or not exploitation is dependant on other findings such as information leaks

Remediation status

As part of our reporting, remediation recommendations are provided to the client. To help track the issues, we also provide a remediation status rating in the findings report.

In some cases, the organization may be confident to remediate the issue and test it internally. In other cases, Subgraph works with the organization to re-test the findings, resulting in a subsequent report reflecting remediation status updates.

If requested to re-test findings, we determine the remediation status based on our ability to reproduce the finding. This is based on our understanding of the finding and our awareness of potential variants at that time. To reproduce the results, the re-test environment should be as close to the original test environment as possible.

Security findings are often due to unexpected or unanticipated behavior that is not always understood by the testers or the developers. Therefore, it is possible that a finding or variations of the finding may still be present even if it is not reproducible during a re-test. While we will do our best to work with the organization to avoid this, it is still possible.

The findings report includes the following remediation status information:

Resolved
Finding is believed to be remediated, we can no longer reproduce it

In progress
Finding is in the process of being remediated

Unresolved
Finding is not remediated – may indicate the initial report, a finding under investigation, or one that the organization has chosen not to address

Not applicable
There is nothing to resolve, this may be the case with informational findings