# Project: sast-automation

***Visit: http://localhost:9000/dashboard?id=sast-automation for more details***

***5 Security Hotspot(s)***

## Components

| Key | Qualifier | Name | Path |
|---|---|---|---|
| sast-automation:index.ts | FIL | index.ts | index.ts |
| sast-automation:utils/misc.utils.ts | FIL | misc.utils.ts | utils/misc.utils.ts |
| sast-automation:controllers/user.controller.ts | FIL | user.controller.ts | controllers/user.controller.ts |
| sast-automation | TRK | sast-automation | |

## Hotspots

| Key | Component Path | Security Category | Vulnerability Probability | Line |
|---|---|---|---|---|
| 878f4579-14d3-498a-966e-4ec04eddb19d | index.ts | dos | MEDIUM | 20 |
| be2147bd-9d16-4d73-81f9-ffdce70de05e | controllers/user.controller.ts | encrypt-data | LOW | 13 |
| a56af880-2ab6-4e10-87a9-2d932d595148 | index.ts | insecure-conf | LOW | 23 |
| aa336ea9-a78c-42f9-baa4-77757c2a48d0 | index.ts | others | LOW | 18 |
| 8858bbc2-2747-4ce1-a059-02a0e40c21b4 | utils/misc.utils.ts | others | LOW | 79 |

## Hotspots Details

### *878f4579-14d3-498a-966e-4ec04eddb19d*

***Component: index.ts***

***Security Category: dos***

***Vulnerability Probability: MEDIUM***

***Line: 20***

***Message: Make sure the content length limit is safe here.***

***Risk Description:***

Rejecting requests with significant content length is a good practice to control the network traffic intensity and thus resource consumption in order to prevent DoS attacks.

## *be2147bd-9d16-4d73-81f9-ffdce70de05e*

***Component: controllers/user.controller.ts***

***Security Category: encrypt-data***

***Vulnerability Probability: LOW***

***Line: 13***

***Message: Using http protocol is insecure. Use https instead.***

***Risk Description:***

Clear-text protocols such as ftp, telnet, or http lack encryption of transported data, as well as the capability to build an authenticated connection. It means that an attacker able to sniff traffic from the network can read, modify, or corrupt the transported content. These protocols are not secure as they expose applications to an extensive range of risks: sensitive data exposure traffic redirected to a malicious endpoint malware-infected software update or installer execution of client-side code corruption of critical information Even in the context of isolated networks like offline environments or segmented cloud environments, the insider threat exists. Thus, attacks involving communications being sniffed or tampered with can still happen. For example, attackers could successfully compromise prior security layers by: bypassing isolation mechanisms compromising a component of the network getting the credentials of an internal IAM account (either from a service account or an actual person) In such cases, encrypting communications would decrease the chances of attackers to successfully leak data or steal credentials from other network components. By layering various security practices (segmentation and encryption, for example), the application will follow the *defense-in-depth* principle. Note that using the http protocol is being deprecated by major web browsers. In the past, it has led to the following vulnerabilities: CVE-2019-6169 CVE-2019-12327 CVE-2019-11065 Exceptions No issue is reported for the following cases because they are not considered sensitive: Insecure protocol scheme followed by loopback addresses like 127.0.0.1 or localhost.

## *a56af880-2ab6-4e10-87a9-2d932d595148*

***Component: index.ts***

***Security Category: insecure-conf***

***Vulnerability Probability: LOW***

***Line: 23***

***Message: Make sure that enabling CORS is safe here.***

***Risk Description:***

Having a permissive Cross-Origin Resource Sharing policy is security-sensitive. It has led in the past to the following vulnerabilities: CVE-2018-0269 CVE-2017-14460 Same origin policy in browsers prevents, by default and for security-reasons, a javascript frontend to perform a cross-origin HTTP request to a resource that has a different origin (domain, protocol, or port) from its own. The requested target can append additional HTTP headers in response, called CORS, that act like directives for the browser and change the access control policy / relax the same origin policy.

## *aa336ea9-a78c-42f9-baa4-77757c2a48d0*

***Component: index.ts***

***Security Category: others***

***Vulnerability Probability: LOW***

***Line: 18***

***Message: This framework implicitly discloses version information by default. Make sure it is safe here.***

***Risk Description:***

Disclosure of version information, usually overlooked by developers but disclosed by default by the systems and frameworks in use, can pose a significant security risk depending on the production environement. Once this information is public, attackers can use it to identify potential security holes or vulnerabilities specific to that version. Furthermore, if the published version information indicates the use of outdated or unsupported software, it becomes easier for attackers to exploit known vulnerabilities. They can search for published vulnerabilities related to that version and launch attacks that specifically target those vulnerabilities.

## *8858bbc2-2747-4ce1-a059-02a0e40c21b4*

***Component: utils/misc.utils.ts***

***Security Category: others***

***Vulnerability Probability: LOW***

***Line: 79***

***Message: Make sure that expanding this archive file is safe here.***

***Risk Description:***

Successful Zip Bomb attacks occur when an application expands untrusted archive files without controlling the size of the expanded data, which can lead to denial of service. A Zip bomb is usually a malicious archive file of a few kilobytes of compressed data but turned into gigabytes of uncompressed data. To achieve this extreme compression ratio, attackers will compress irrelevant data (eg: a long string of repeated bytes).