|  |
| --- |
| Nagios fusion XI |
| Vulnerability Report |
| Tuesday, July 5, 2022 |



modifications history

| **Version** | **Date** | **Author** | **Description** |
| --- | --- | --- | --- |
| 1.0 | 07/05/2022 | Ariane BLOW | Initial Version |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

1. General Information 4

1.1 Scope 4

1.2 Organisation 4

2. Executive Summary 5

3. Technical Details 6

3.1 title 6

4. Vulnerabilities summary 8

# General Information

## Scope

ArianeSec.ninja has mandated us to perform security tests on the following scope:

* Nagios Fusion, Nagios XI

## Organisation

The testing activities were performed between 07/01/2022 and 07/04/2022.

# Executive Summary

# Vulnerabilities summary

Following vulnerabilities have been discovered:

|  |  |  |  |
| --- | --- | --- | --- |
| **Risk** | **ID** | **Vulnerability** | **Affected Scope** |
| Critical | IDX-001 | RCE | Application host server. |
| Medium | VULN-003 | LFI | Documents stored on the host server. |
| Medium | VULN-002 | Unsecure FileUpload | host server |

# Technical Details

## RCE

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **CVSS Severity** | **Critical** | | **CVSSv3 Score** | | **9.9** |
| **CVSSv3 criterias** | Attack Vector : | **Network** | Scope : | **Changed** | |
| Attack Complexity : | **Low** | Confidentiality : | **High** | |
| Required Privileges : | **Low** | Integrity : | **High** | |
| User Interaction : | **None** | Availability : | **High** | |
| **Affected scope** | Application host server. | | | | |
| **Description** | A remote system code injection vulnerability allows an attacker to execute system commands by the server via a vulnerability present on the WEB service of the impacted machine. | | | | |
| **Observation** |  | | | | |
| **Test details**  An attacker with access to the application is able to execute SHELL commands by the server hosting an XI application (Nagios, Fusion etc...etc...).  To do so, the authenticated user is able to upload any document by bypassing the security in place present in the "file upload" module of the "dashlet" options (importing a personalized dashboard module).  To do so, the attacker can create malicious PHP code:  (It should be noted that the code includes HTML tags in order to bypass certain filters on the authentication page or will import this malicious script).  image  Image **1** – Code malveillant PHP  When uploading the malicious document, an error message appears:  image  Image **2** – erreur de "file type"  However, it appears that the malicious document has been uploaded to a local folder on the host server in the following directory: /usr/local/nagiosfusion/tmp/SCRIPT.PHP:    image  Image **3** – Téléversement de fichier non sécurisé  This directory is accessible by multiple options present in the application.  By default, the options of "XI" products (Nagios, fusion etc...etc...), offer a component allowing to modify the authentication page with personalized logos for example.  The "Custom Login" component is present and installed by default and is able to access the directory where the malicious script is stored, so it is possible to include malicious code in PHP on the authentication page:  image  Image **4** – Inclusion de documents locaux  This directory is accessible by multiple options present in the application.  By default, the options of "XI" products (Nagios, fusion etc...etc...), offer a component allowing to modify the authentication page with personalized logos for example.  The "Custom Login" component is present and installed by default and is able to access the directory where the malicious script is stored, so it is possible to include malicious code in PHP on the authentication page:  image  Image **5** – Réception du terminal distant | | | | | |
| **Remediation** | Review of the script allowing the file upload of "dashlets" in order to no longer store files and documents with an error during upload.  Include a security review to uploaded documents and files.  Restriction of the directories reachable by the "Custom Login" module so that it is no longer possible to include documents and system files (list of accounts in /etc/passwd for example). | | | | |
| **References** |  | | | | |

## LFI

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **CVSS Severity** | **Medium** | | **CVSSv3 Score** | | **6.5** |
| **CVSSv3 criterias** | Attack Vector : | **Network** | Scope : | **Unchanged** | |
| Attack Complexity : | **Low** | Confidentiality : | **High** | |
| Required Privileges : | **Low** | Integrity : | **None** | |
| User Interaction : | **None** | Availability : | **None** | |
| **Affected scope** | Documents stored on the host server. | | | | |
| **Description** | A local files and documents inclusion vulnerability allows an attacker to extract data that he should not be able to access. | | | | |
| **Observation** |  | | | | |
| **Test details**  An authenticated attacker is able to include any document or system file being stored on the host server.  An attacker having access to the customization of modules and components is able to include local files stored on the host server (eg /etc/passwd, /etc/hosts etc...etc...).  The "Custom Login" component allows you to include any information accessible on the host server on the authentication page:  image  Image **6** – Téléversement de document  This may include previously uploaded files as evidenced by the "remote system command injection" vulnerability and also "SVG" formatted images that may contain XSS-like payloads:  image  Image **7** – XSS stocké | | | | | |
| **Remediation** | Perform a check on the nature of the files and documents searchable by the impacted component.  Make it possible to import files and documents only stored in the files of the WEB server (default web root) in order to make it impossible to import local documents. | | | | |
| **References** |  | | | | |

## Unsecure FileUpload

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **CVSS Severity** | **Medium** | | **CVSSv3 Score** | | **5.4** |
| **CVSSv3 criterias** | Attack Vector : | **Network** | Scope : | **Unchanged** | |
| Attack Complexity : | **Low** | Confidentiality : | **None** | |
| Required Privileges : | **Low** | Integrity : | **Low** | |
| User Interaction : | **None** | Availability : | **Low** | |
| **Affected scope** | host server | | | | |
| **Description** | Unsecured file upload allows an attacker to store malicious documents and files on an application's host server. | | | | |
| **Observation** |  | | | | |
| **Test details**  An attacker with sufficient rights to manage application modules is able to upload any type of documents and files without restriction. The "Dashlets" import module (dashboard customization module) checks the uploaded files and documents but systematically stores the documents and files in a directory even if they contain "MIME, TYPE" type errors, resulting in an insecure file upload vulnerability as demonstrated below: Access to the "File Upload" module of the "dashlets" and import of a malicious script in PHP format:  image  Image **8** – Dashlet management  An error appears during the import, the malicious script has however been uploaded and is located in a directory of the host server:  image  Image **9** – Script téléversé  This vulnerability can be used in a chain of exploitation in order to upload malicious documents which will then be reachable by a visitor. This can result in Stored XSS type vulnerabilities, Remote system command injection, Direct redirection, authentication cookie theft, javascript code injection on visitors' browsers. | | | | | |
| **Remediation** | It is necessary not to store uploaded documents and files permanently on the server. | | | | |
| **References** |  | | | | |