Fortify Security Report

2024-6-21

ASUS

Executive Summary

Issues Overview

On 2024-6-21, a source code review was performed over the PF_RING code base. 16 files, 1,176 LOC (Executable) were scanned and reviewed for defects that could lead to potential security vulnerabilities. A total of 1 reviewed findings were uncovered during the analysis.

Issues by Fortify Priority Order

Critical	1

Recommendations and Conclusions

The Issues Category section provides Fortify recommendations for addressing issues at a generic level. The recommendations for specific fixes can be extrapolated from those generic recommendations by the development group.

Project Summary

Code Base Summary

Code location: C:/Users/ASUS/Desktop/Gitrepo/PF_RING
Number of Files: 16
Lines of Code: 1176
Build Label: <No Build Label>

Scan Information

Scan time: 00:27
SCA Engine version: 20.1.1.0007
Machine Name: DESKTOP-MK5UPFE
Username running scan: ASUS

Results Certification

Results Certification Valid

Details:

Results Signature:

SCA Analysis Results has Valid signature

Rules Signature:

There were no custom rules used in this scan

Attack Surface

Attack Surface:
Command Line Arguments:
null.null.null

File System:
null.null.open
null.file.__init__

System Information:
null.null.null
null.null.null
os.null.getcwd

Filter Set Summary

Current Enabled Filter Set:
Quick View

Filter Set Details:

Folder Filters:

If [fortify priority order] contains critical Then set folder to Critical
If [fortify priority order] contains high Then set folder to High
If [fortify priority order] contains medium Then set folder to Medium
If [fortify priority order] contains low Then set folder to Low
Visibility Filters:
If impact is not in range [2.5, 5.0] Then hide issue
If likelihood is not in range (1.0, 5.0] Then hide issue

Audit Guide Summary

J2EE Bad Practices

Hide warnings about J2EE bad practices.
Depending on whether your application is a J2EE application, J2EE bad practice warnings may or may not apply. AuditGuide can hide J2EE bad practice warnings.
Enable if J2EE bad practice warnings do not apply to your application because it is not a J2EE application.

Filters:
If category contains j2ee Then hide issue
If category is race condition: static database connection Then hide issue

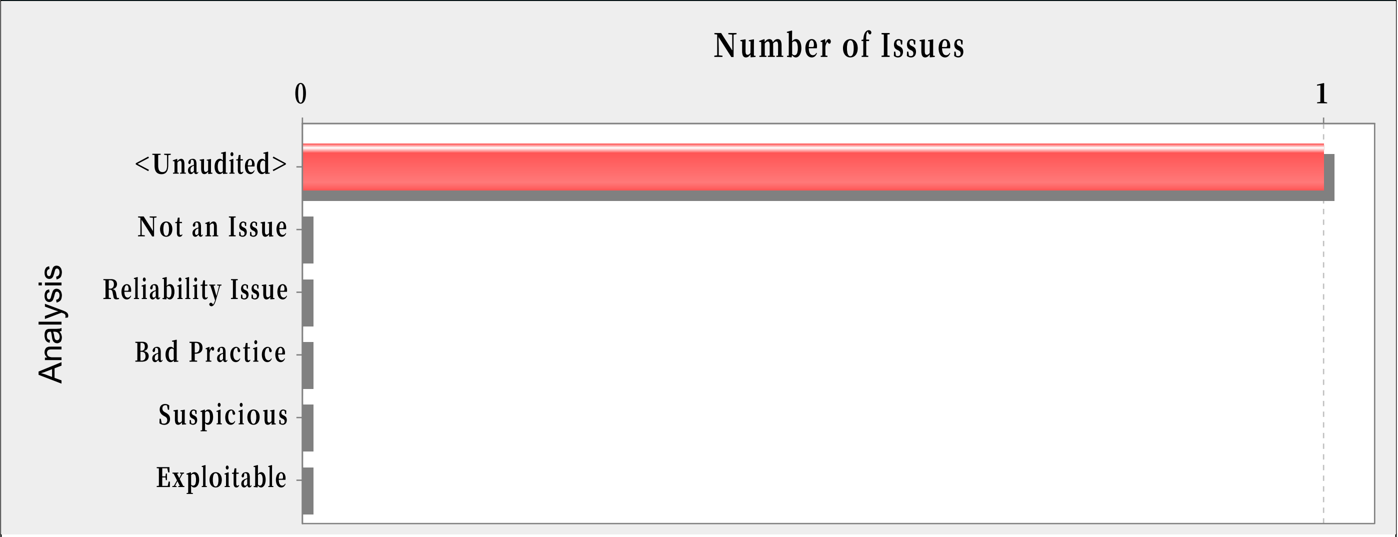
Results Outline

Overall number of results

The scan found 1 issues.

Vulnerability Examples by Category

Category: Key Management: Hardcoded Encryption Key (1 Issues)



Abstract:

Hardcoded 加密密钥可能会削弱系统安全性，一旦出现安全问题将无法轻易修正。

Explanation:

使用硬编码方式处理加密密钥绝非好方法。这不仅是因为所有项目开发人员都可以使用通过硬编码方式处理的加密密钥，而且还会使解决这一问题变得极其困难。在代码投入使用之后，必须对软件进行修补才能更改加密密钥。如果受加密密钥保护的帐户遭受入侵，系统所有者将必须在安全性和可用性之间做出选择。

示例：下列代码使用 hardcoded 加密密钥来加密信息：

```
...
from Crypto.Ciphers import AES
encryption_key = b'_hardcoded__key_'
cipher = AES.new(encryption_key, AES.MODE_CFB, iv)
msg = iv + cipher.encrypt(b'Attack at dawn')
...
```

此代码将成功运行，但任何有权访问此代码的人都可以获得加密密钥。一旦程序发布，除非修补该程序，否则可能无法更改硬编码的加密密钥 _hardcoded__key_。心怀不轨的雇员可以利用其对此信息的访问权限来破坏系统加密的数据。

Recommendations:

绝不能对加密密钥进行硬编码。通常情况下，应对加密密钥加以模糊化，并在外部资源文件中进行管理。如果在系统中采用明文的形式存储加密密钥，任何有足够权限的人即可读取加密密钥，还可能误用这些密码。

adqsetup.py, line 824 (Key Management: Hardcoded Encryption Key)

Fortify Priority:	Critical	**Folder**	Critical
Kingdom:	Security Features		

Abstract: Hardcoded 加密密钥可能会削弱系统安全性，一旦出现安全问题将无法轻易修正。

Sink: adqsetup.py:824 Operation()

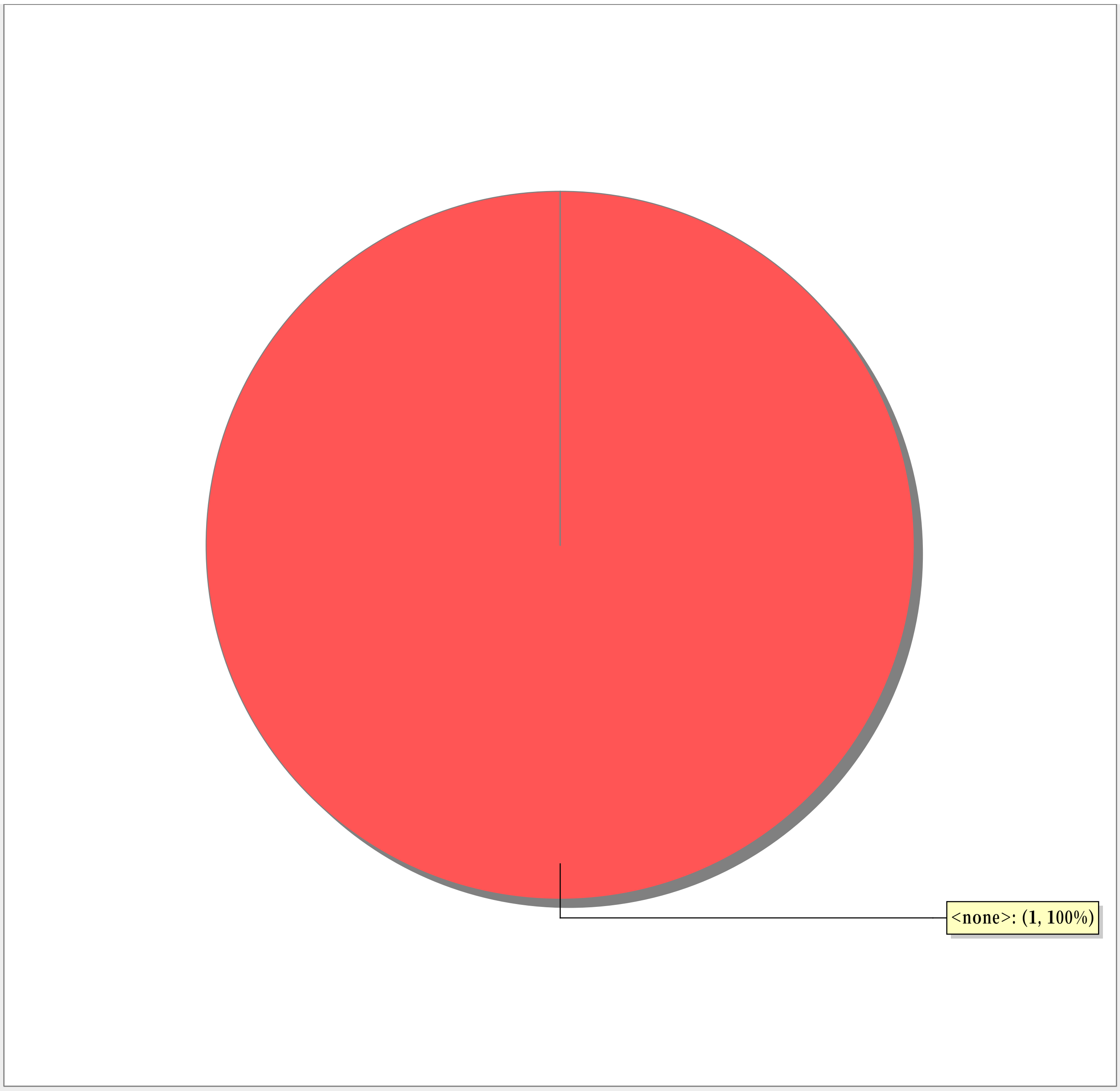
```
822
823         def __getitem__(self, key):
824             if key == 'globals':
825                 return OrderedDict(self.globals)
826             else:
```

Issue Count by Category

Issues by Category	
Key Management: Hardcoded Encryption Key	1

Issue Breakdown by Analysis

Issues by Analysis



● <none>