Fortify Security Report

2024-6-21

ASUS

Executive Summary

Issues Overview

On 2024-6-21, a source code review was performed over the sumatrapdf code base. 396 files, 27,250 LOC (Executable) were scanned and reviewed for defects that could lead to potential security vulnerabilities. A total of 7 reviewed findings were uncovered during the analysis.

Issues by Fortify Priority Order	
High	4
Critical	3

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/sumatrapdf
Number of Files: 396
Lines of Code: 27250
Build Label: <No Build Label>

Scan Information

Scan time: 03:08
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
example.Example.main
example.MultiThreaded.main
example.MultiThreadedWithPool.main
example.StoryTest.main
example.TraceDevice.main
example.Viewer.main

Environment Variables:
null.null.null
os.null.getenv

File System:
null.null.open
null.file.__init__
null.file.read
null.file.readline
null.file.readlines
io.ioutil.null.ReadFile

GUI Form:
java.awt.TextComponent.getText

Private Information:
null.null.null

Standard Input Stream:
null.null.null
null.null.input
null.file.read

Stream:
com.artifex.mupdf.fitz.BufferInputStream.read
com.artifex.mupdf.fitz.FitzInputStream.read
java.io.InputStream.read
java.io.RandomAccessFile.read
os.null.read

System Information:
null.null.null
null.null.null
null.null.globals
io.ioutil.null.ReadDir
java.lang.System.getProperty
java.lang.Throwable.getMessage
os.null.Getwd
os.null.getcwd
os.null.listdir
os.null.uname
site.null.getsitepackages
sys.null.exc_info
sysconfig.null.get_path

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

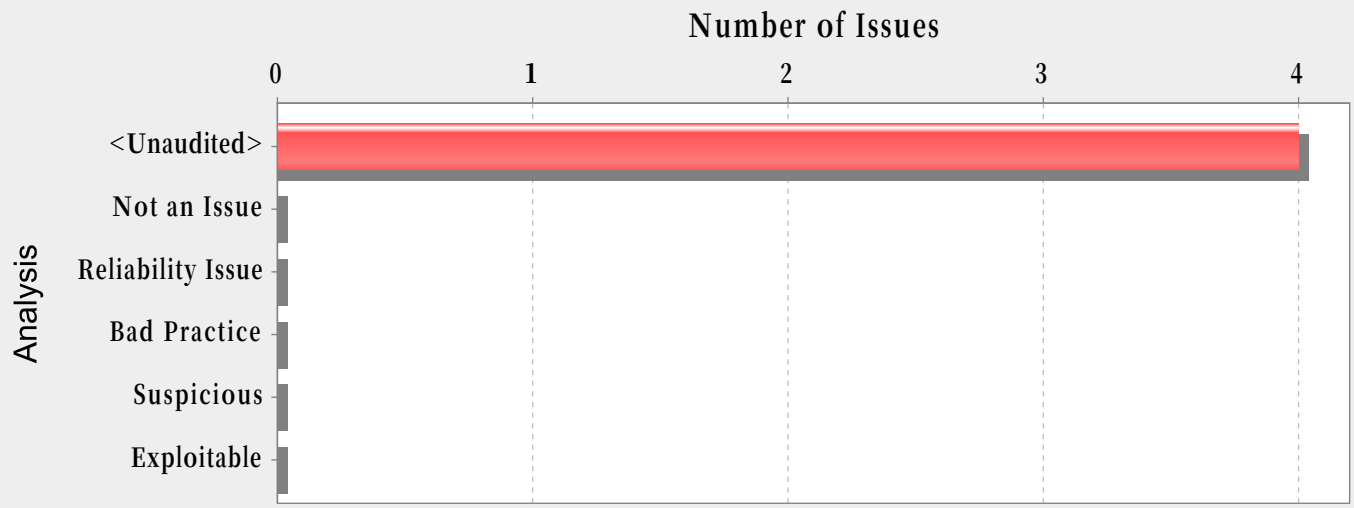
Results Outline

Overall number of results

The scan found 7 issues.

Vulnerability Examples by Category

Category: Password Management: Empty Password (4 Issues)



Abstract:

Empty password 可能会危及系统安全，并且无法轻易修正出现的安全问题。

Explanation:

为密码变量指定空字符串绝非一个好方法。如果使用 empty password 成功通过其他系统的验证，那么相应帐户的安全性很可能会被减弱，原因是其接受了 empty password。如果在为变量指定一个合法的值之前，empty password 仅仅是一个占位符，那么它将给任何不熟悉代码的人造成困惑，而且还可能导致出现意外控制流路径方面的问题。

示例：以下代码尝试使用空密码连接到数据库。

```
...
db = mysql.connect("localhost","scott","","mydb")
...
```

如果此示例中的代码成功执行，则表明数据库用户帐户“ scott ”配置有一个空密码，攻击者可以轻松地猜测到该密码。一旦程序发布，要更新此帐户以使用非空密码，就需要对代码进行更改。

Recommendations:

始终从加密的外部资源读取存储的密码值，并为密码变量指定有意义的值。确保从不使用空密码或 null 密码来保护敏感资源。

Tips:

1. 避免在源代码中使用 empty password，还要避免使用默认密码。

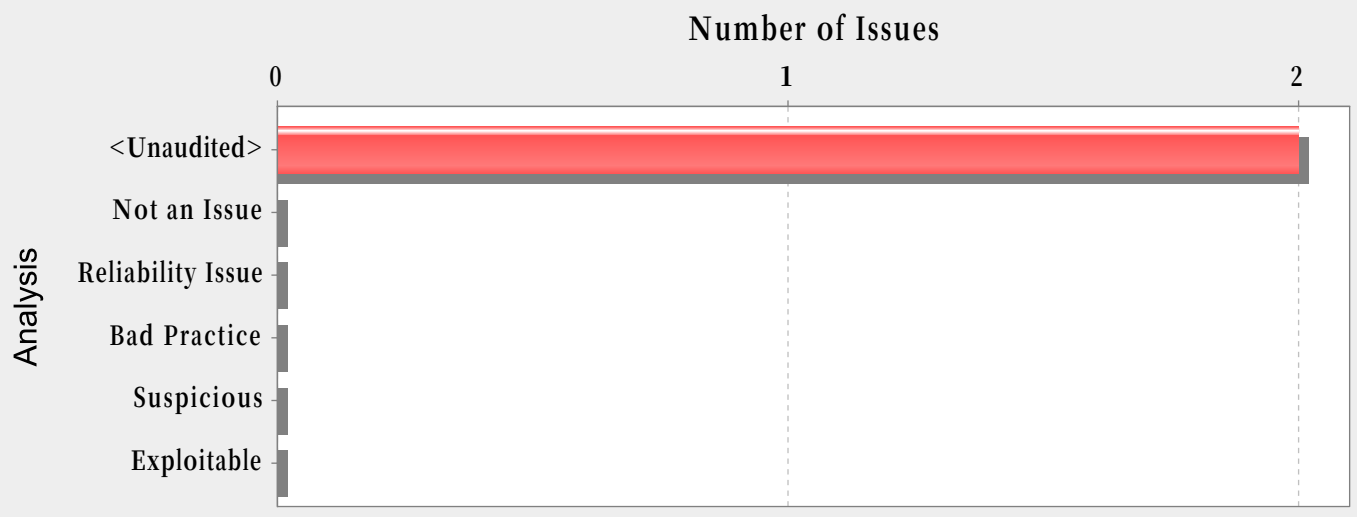
2. 在识别 null 密码、空密码和硬编码密码时，默认规则只会考虑包含 password 一词的字段和变量。但是，使用 Fortify Custom Rules Editor（Fortify 自定义规则编辑器）提供的“ Password Management（密码管理） ”向导可轻松创建用于在自定义命名字段和变量中检测 password management 问题的规则。

mutool.py, line 71 (Password Management: Empty Password)

Fortify Priority:	High	**Folder**	High
Kingdom:	Security Features		
Abstract:	Empty password 可能会危及系统安全，并且无法轻易修正出现的安全问题。		
Sink:	mutool.py:71 VariableAccess: password()		

```
69  def clean(argv):
70    outfile = 'out.pdf'
71    password ="******
72    opts = mupdf.PdfWriteOptions()
73    print( 'opts.do_garbage=%s' % opts.do_garbage)
```

Category: Insecure SSL: Server Identity Verification Disabled (2 Issues)



Abstract:
在进行 SSL 连接时，通过 update_auto_update_ver.py 中的 urlopen() 建立的连接不验证服务器证书。这使得应用程序易受到中间人攻击。

Explanation:
在一些使用 SSL 连接的库中，可以禁用服务器证书验证。这相当于信任所有证书。

例 1：此应用程序在默认情况下不会验证服务器证书：

```
...
import ssl
ssl_sock = ssl.wrap_socket(s)
...
```

当尝试连接到有效主机时，此应用程序将随时接受颁发给 " hackedserver.com " 的证书。此时，当服务器被黑客攻击发生 SSL 连接中断时，应用程序可能会泄漏用户敏感信息。

Recommendations:
当进行 SSL 连接时，不要忘记服务器验证检查。根据所使用的库，一定要验证服务器身份并建立安全的 SSL 连接。

例 2：此应用程序明确地验证服务器证书。

```
...
ssl_sock = ssl.wrap_socket(s, ca_certs="/etc/ca_certs_file", cert_reqs=ssl.CERT_REQUIRED)
...
```

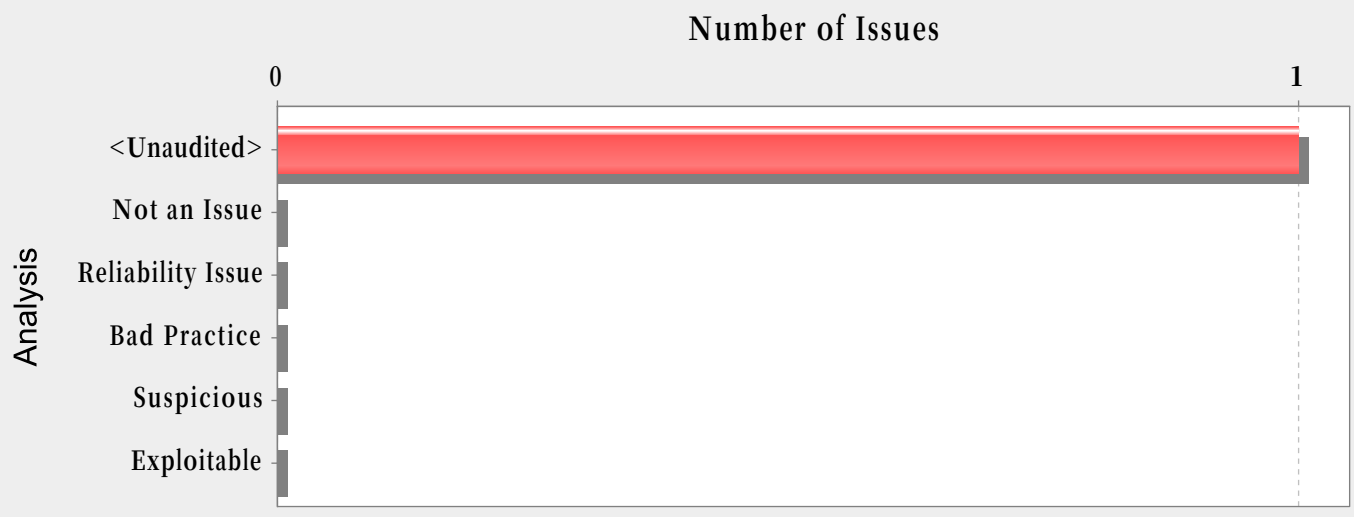
update_auto_update_ver.py, line 59 (Insecure SSL: Server Identity Verification Disabled)

Fortify Priority:	Critical	**Folder**	Critical
Kingdom:	Security Features		
Abstract:	在进行 SSL 连接时，通过 update_auto_update_ver.py 中的 urlopen() 建立的连接不验证服务器证书。这使得应用程序易受到中间人攻击。		

Sink: update_auto_update_ver.py:59 FunctionCall: urlopen()

```
57  def get_update_versions(url):
58    try:
59      data = urllib2.urlopen(url).read()
60      root = SquareTree.Parse(data)
61      node = root.GetChild("SumatraPDF")
```

Category: Dynamic Code Evaluation: Code Injection (1 Issues)



Abstract:
在运行时中解析用户控制的指令，会让攻击者有机会执行恶意代码。

Explanation:
许多现代编程语言都允许动态解析源代码指令。这使得程序员可以执行基于用户输入的动态指令。当程序员错误地认为由用户直接提供的指令仅会执行一些无害的操作时（如对当前的用户对象进行简单的计算或修改用户的状态），就会出现 code injection 漏洞：然而，若不经过适当的验证，用户指定的操作可能并不是程序员最初所期望的。

示例：在这个经典的 code injection 实例中，应用程序可以实施一个基本的计算器，该计算器允许用户指定执行命令。

```
...
userOps = request.GET['operation']
result = eval(userOps)
...
```

如果 operation 参数的值为良性值，程序就可以正常运行。例如，当该值为“ 8 + 7 * 2 ”时，result 变量被赋予的值将为 22。然而，如果攻击者指定的语言操作是有效的，又是恶意的，那么，将在对主进程具有完全权限的情况下执行这些操作。如果底层语言提供了访问系统资源的途径或允许执行系统命令，这种攻击甚至会更加危险。例如，如果攻击者计划将“ os.system('shutdown -h now') ”指定为 operation 的值，主机系统就会执行关机命令。

Recommendations:
在任何时候，都应尽可能地避免动态的代码解析。如果程序的功能要求对代码进行动态的解析，您可以通过以下方式将此种攻击的可能性降低到最小：尽可能的限制程序中动态执行的代码数量，将此类代码应用到特定的应用程序和上下文中的基本编程语言的子集。

如果需要执行动态代码，应用程序绝不应当直接执行和解析未验证的用户输入。而应采用间接方法：创建一份合法操作和数据对象列表，用户可以指定其中的内容，并且只能从中进行选择。利用这种方法，就绝不会直接执行由用户提供的输入。

pypackage.py, line 805 (Dynamic Code Evaluation: Code Injection)

Fortify Priority:	Critical	**Folder**	Critical
Kingdom:	Input Validation and Representation		
Abstract:	在运行时中解析用户控制的指令，会让攻击者有机会执行恶意代码。		
Sink:	pypackage.py:805 FunctionCall: input()		

```
803         except Exception as e:
804             jlib.log('Failed to upload: {e=}')
805             input(jlib.log_text('Press <enter> to retry... ').strip())
806         else:
807             break
```

Issue Count by Category	
Issues by Category	
Password Management: Empty Password	4
Insecure SSL: Server Identity Verification Disabled	2
Dynamic Code Evaluation: Code Injection	1

Issue Breakdown by Analysis

Issues by Analysis