Address Vulnerabilities in "3D slicer" Slicing Software

IT20028046
Gunathilaka S.B.M.B.S.A
It20028046@my.sliit.lk
shehanbuddhika43@gmail.com
SLIIT- cyber security (undergraduate)
Student at the Sri Lanka Institute of Information Technology

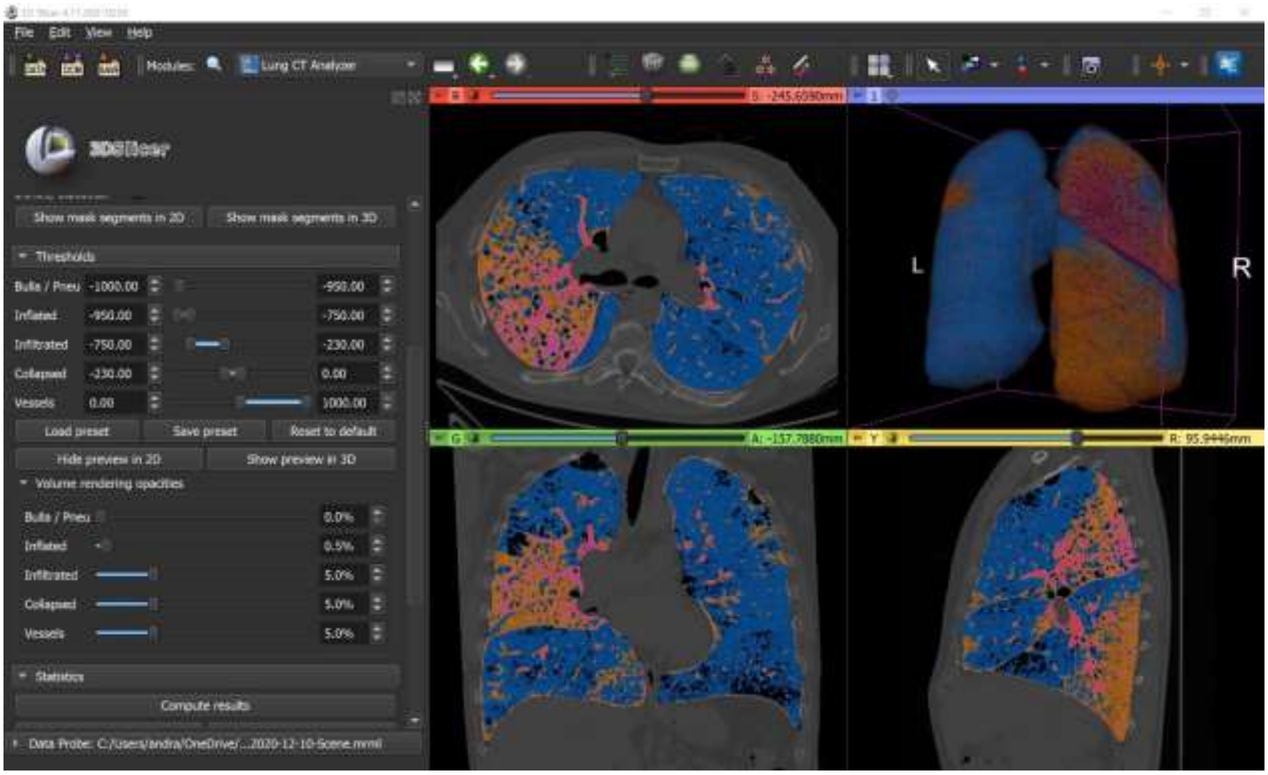
Video link: *https://drive.google.com/drive/folders/1ZsSGBbt8xmnSLdXJLLfRiGxi-DzMdXYo?usp=sharing*

ABSTRACT

In this era, we can see different kind of software built for different kind of purposes when we are browsing the internet. Among those applications the healthcare related application take a special place because with the technological improvement in the healthcare sector threats are also emerging that can harm to the healthcare sector.as the cyber security community we need to mitigate those threats by fix bugs and vulnerabilities in those applications and come up with more secure applications.in this report we are trying to find a solution to fix vulnerabilities in 3d slicer software and we analyze the source code from the code level.in this work we discuss what methods we can use and what tools and the bug fixing procedure.

INDEX TERMS

cybersecurity, healthcare, bugs, vulnerabilities, 3D slicer, procedure



Bulla / Pneu -1000.00 -950.00
Inflated -950.00 -750.00
Infiltrated -750.00 -230.00
Collapsed -230.00 0.00
Vessels 0.00 1000.00

Bulla / Pneu 0.0%
Inflated 0.5%
Infiltrated 5.0%
Collapsed 5.0%
Vessels 5.0%

Data Probe: C:/users/andra/OneDrive/...2020-12-10-Scene.mrml

Figure.1 3D slicer software

I. INTRODUCTION

the "3d slicer" is basically a slicing software. Assume If someone has a 3d printer and has assembled that printer, he will print his first 3d model. Then he tries to give that 3d model to the printer, but it doesn't take that file. Instead of that, it needs to be converted into a machine code. (3D Slicer, n.d.) Then, that machine code can tell the printer how and where to move and extrude filament. So, this is the place slicer Software come into play. A slicer processes the 3D model and generates a gcode file that is machine instructions that the 3D printer can understand. (what-is-3D-slicer-software, n.d.) cruising speed and the settings you want to print with. This can include layer height, nozzle temperature, support to use, margin settings, and any other settings you need to get the perfect print. At a conceptual level, what a slicer does is pretty simple. It takes 2D "slices" of your 3d version and calculates the satisfactory direction on your printer to take to create that shape. It does this for every layer it desires to print and bundles all the commands into one file. Easy, right? Not pretty. There is much stuff the slicer desires to parent out. Where does the infill go? Does this segment want to support? How speedy must the fringe of the version be printed, and how fast must the infill? It receives complex very quickly. That's why multiple slicers were evolved to deal with those questions. Therefore, this 3D slicer is an application for visualizing and analyzing computer data sets of medical images. All standard data sets such as images, slicers, surfaces, annotations, transformations, etc., are supported in 2D, 3D, and 4D. The visualization is available on desktop, and in V.R. The analysis includes segmentation, logging, and various quantifications (build instructions, n.d.).

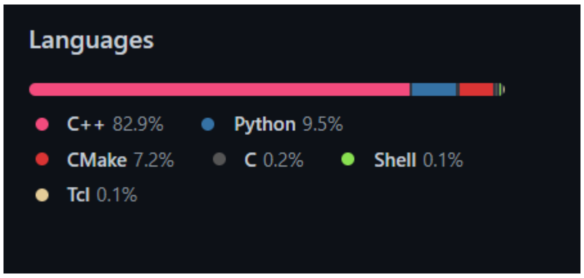
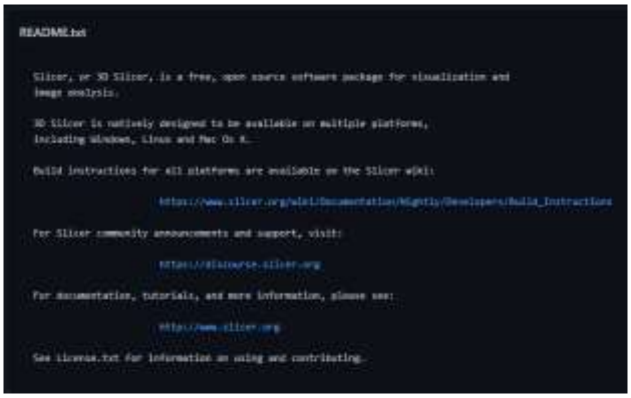


Figure.2 used languages

II. LITERATURE

This research software program platform lets researchers speedy expand and compare new strategies and distribute them to scientific users. All functions are to be had and extensible in Python and C++. A complete Python surrounding is supplied wherein any Python programs may be set up and mixed with integrated functions. Slicer has an integrated Python console and might act as a Jupyter notebook kernel with remote 3D rendering capabilities. (user_guide, n.d.) This software is available for free and open-source applications also, and it can run on multiple operating systems such as Linux, macOS, and windows. It has every organ from head to toe in its database.

Users can add greater abilities without difficulty by installing extra modules from the Extensions manager, walking custom Python scripts within the integrated Python console, running any executables from the application's consumer interface, or putting in force custom modules in Python or C++.now we can look into the source code and other stuff (what-is-a-3d-slicer-simply-explained, n.d.).

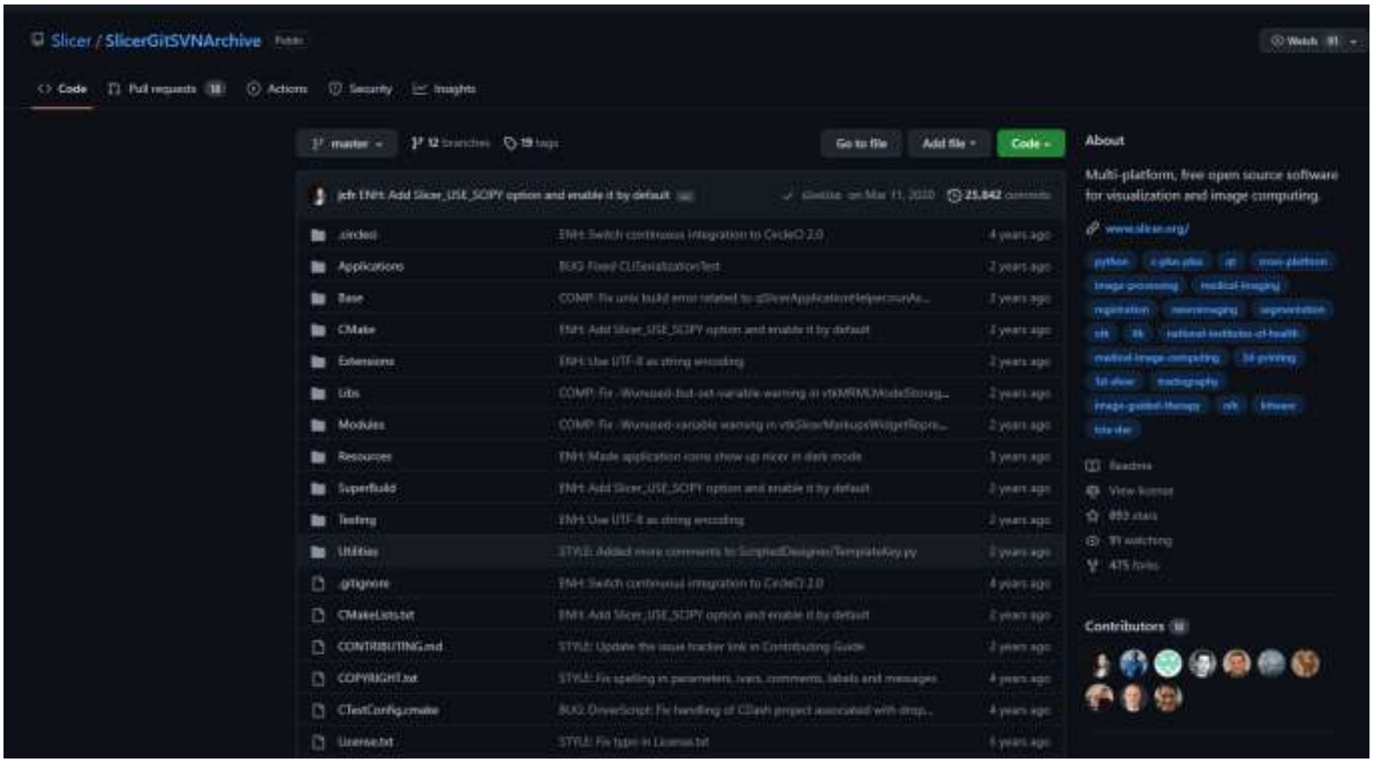


README.txt

```
Slicer, or 3D Slicer, is a free, open source software package for visualization and
image analysis.

3D Slicer is natively designed to be available on multiple platforms,
including Windows, Linux and Mac Os X.

Build instructions for all platforms are available on the Slicer wiki:

        https://www.slicer.org/wiki/Documentation/Nightly/Developers/Build_Instructions

For Slicer community announcements and support, visit:

        https://discourse.slicer.org

For documentation, tutorials, and more information, please see:

        https://www.slicer.org

See License.txt for information on using and contributing.
```

Figure.3 readme information

I get this source code from a git hub repository. It has every file, including the source code. Most of the software is written in c++ language, and it uses some other languages such as python, CMake, shell, and TCL to get its functionalities.


Slicer / SlicerGitSVNArchive Public

Code · Pull requests · Actions · Security · Insights

master · 12 branches · 19 tags · Go to file · Add file · Code

jcfr ENH: Add Slicer_USE_SCIPY option and enable it by default · sliceline on Mar 11, 2020 · 25,842 commits

.circleci	ENH: Switch continuous integration to CircleCI 2.0	4 years ago
Applications	BUG: Fixed CLISerializationTest	2 years ago
Base	COMP: Fix unix build error related to qSlicerApplicationHelper...	2 years ago
CMake	ENH: Add Slicer_USE_SCIPY option and enable it by default	2 years ago
Extensions	ENH: Use UTF-8 as string encoding	2 years ago
Libs	COMP: Fix -Wunused-but-set-variable warning in vtkMRMLModelStorag...	2 years ago
Modules	COMP: Fix -Wunused-variable warning in vtkSlicerMarkupsWidgetRepre...	2 years ago
Resources	ENH: Made application icons show up nicer in dark mode	3 years ago
SuperBuild	ENH: Add Slicer_USE_SCIPY option and enable it by default	2 years ago
Testing	ENH: Use UTF-8 as string encoding	2 years ago
Utilities	STYLE: Added more comments to ScriptedDesignerTemplateKey.py	2 years ago
.gitignore	ENH: Switch continuous integration to CircleCI 2.0	4 years ago
CMakeLists.txt	ENH: Add Slicer_USE_SCIPY option and enable it by default	2 years ago
CONTRIBUTING.md	STYLE: Update the issue tracker link in Contributing Guide	2 years ago
COPYRIGHT.txt	STYLE: Fix spelling in parameters, ivars, comments, labels and messages	4 years ago
CTestConfig.cmake	BUG: DriverScript: Fix handling of CDash project associated with drop...	4 years ago
License.txt	STYLE: Fix typo in License.txt	6 years ago

About

Multi-platform, free open source software for visualization and image computing.

www.slicer.org/

python · c-plus-plus · qt · cross-platform · image-processing · medical-imaging · registration · neuroimaging · segmentation · vtk · itk · national-institutes-of-health · medical-image-computing · 3d-printing · 3d-slicer · tractography · image-guided-therapy · nih · kitware · tcia-dac

Readme · View license · 893 stars · 91 watching · 475 forks

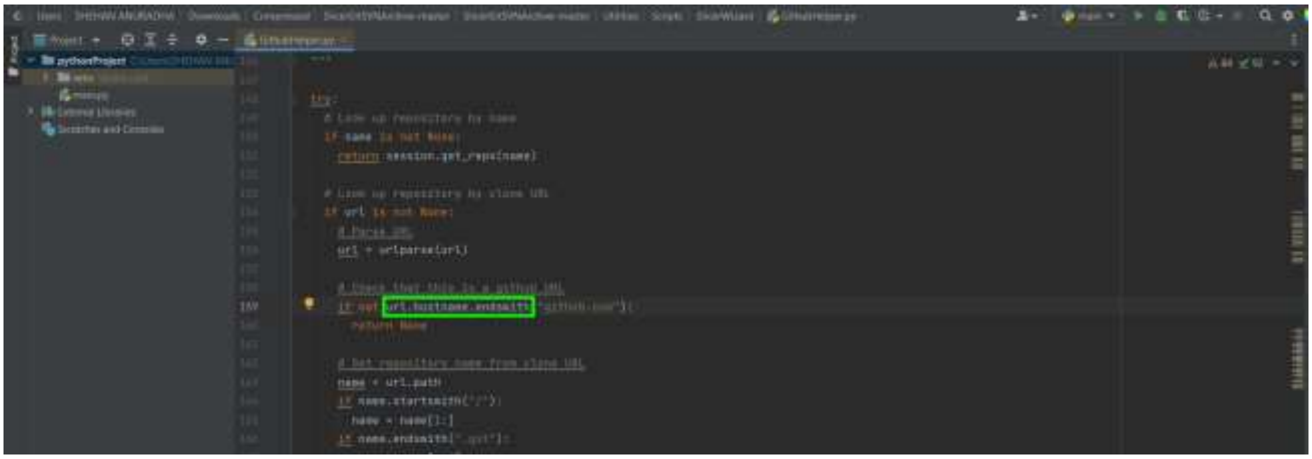
Contributors

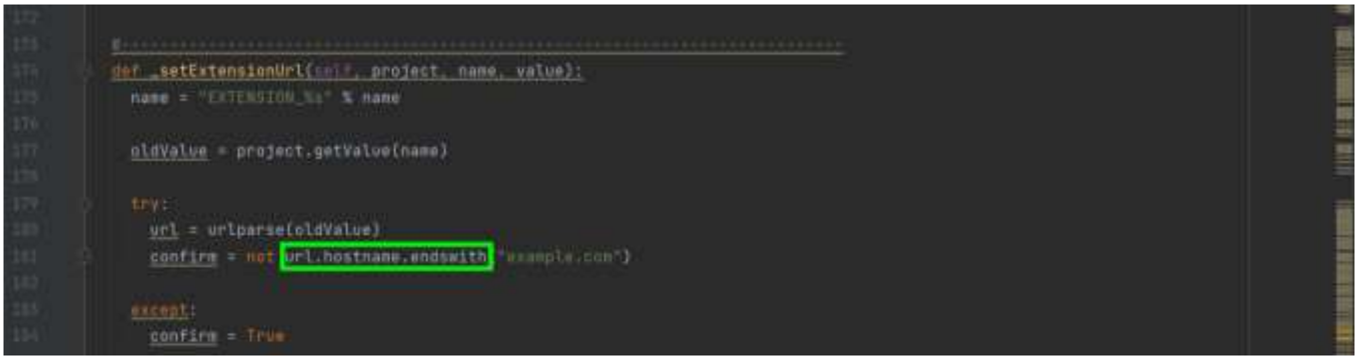
Figure.4 3d slicer GitHub repository

III. DISCOVERED VULNERABILITIES

- Incomplete URL sanitization
- Insecure Hash
- Insecure Xml Parser
- Command Injection

A. *Incomplete URL sanitization*


```
try:
    # Look up repository by name
    if name is not None:
        return session.get_repo(name)

    # Look up repository by clone URL
    if url is not None:
        # Parse URL
        url = urlparse(url)

        # Check that this is a github URL
        if not url.hostname.endswith("github.com"):
            return None

        # Get repository name from clone URL
        name = url.path
        if name.startswith("/"):
            name = name[1:]
        if name.endswith(".git"):
```



```
#-----------------------------------------------------------------------
def _setExtensionUrl(self, project, name, value):
  name = "EXTENSION_%s" % name

  oldValue = project.getValue(name)

  try:
    url = urlparse(oldValue)
    confirm = not url.hostname.endswith("example.com")

  except:
    confirm = True
```

The reason is to implement input sanitization mechanisms in a application filter the user inputs and allow only the properly formed data to enter to the workflow in the application. (Input Sanitization, n.d.) If there is no any input sanitization mechanisms and an user enters malicious data set to the application and that data set goes to the database of the application and persisting in the database. These kinds of malformed data can trigger some of vulnerabilities in the application and change the main functionality. because of that the input sanitization should happen as soon as possible when the data received from the user.

User inputs which are came from unreliable sources including internet ,web applications and also other inputs that are coming from other private networks ,stakeholders such as vendors,partners,suppliers as well as operators all those inputs should be validated. By doing this input validation it doesn’t means that we can fully control attacks such that XSS ,SQLi and other types of malicious attacks but it can reduce the attack impact in a considerable level if we implement the input validation mechanism correctly (Maury, n.d.).

We can find this vulnerability from the 3d slicer software’s githubhelper.py file and it is a Incomplete URL substring sanitization type of vulnerability.

Except other common type of attacks url sanitization helps to mitigate attacks like request forgeries and redirection type of attacks.we can create a whitelist(list of allowed hosts)then a host is check against this whitelist and if the host is in the whitelist it will allow.check the urls as just strings and allow or reject them can be lead to errors because the string can contain a part of url

which is in the whitelist then it can be allowed. Using this method forged URLs can bypass the security countermeasures by embed with a allowed URL. Even if the substring check isn't utilized in a security-critical situation, the incomplete check can result in unwanted behavior if it succeeds by accident.

In the given code sector Data is 'tainted' if it comes from an insecure source such as a file, the network, or the user. This tainted data is passed to a sensitive sink. Sinks are the operations that must receive clean data and that you wouldn't want an attacker to be able to manipulate (Smith, n.d.).

Fix

Before checking the host value of a URL, parse it and make sure the validation handles random subdomain sequences properly.

This is the unchanged code segment

```
# Check that this is a github URL
if not
url.hostname.endswith("github.com"):
  return None
```

We can change this to prevent bypassing Security checks on the substrings of an unparsed URL.

```
# Check that this is a github URL
if not
url.hostname.endswith(".github.com"):
  return None
```

Now the user cannot embed the URL with other component and bypass the security checking.to add more security we can use a explicit whitelist of allowed hosts and then the redirecting happens it will be more secured.

Ex: let allowed host = ['example.com ', 'beta.example.com','www.example.com'];

B. *Insecure Hash*

A hashing algorithm is a special kind of algorithm that get a data array as a input and convert that data array to a fixed length string.this programming function get arbitrary length of data and convert a string which has a fixed length .this hashing is used to store large files as compressed files in small amount of memory.

The reason is to use this hashing is compress large data and then the memory required for store files is lesser than earlier.and also it speed up file search performance then it is used to simplify the database management .one file always create a unique hash digest and two file's hash digests cannot be same.if the two hash digests are same we call it as a collision (insecure-default-password-hashing-cms, n.d.).

This hashing mechanism mainly used for store passwords and helps to authenticate users to a system.because it is lightweight and faster and attackers cannot get the real password if the password file compromised .In our software it uses md5 hashing algorithm but it is already compromised.instead of that we can suggest a new much secure hashing algorithm and change the code to use that algorithms instead of old one.

MD5 has the possibility for message collisions if message hash codes are accidentally duplicated, which is a big worry. The length of MD5 hash code strings is similarly limited to 128 bits. This makes them easier to crack than subsequent hash coding techniques.

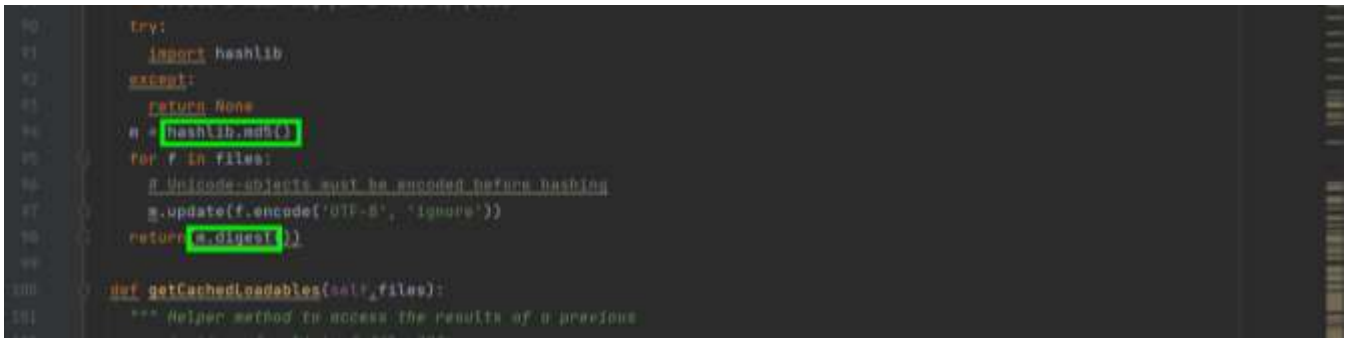
Fix

hashlib.md5 is not enough security and it is already implemented in the original code.

```
m = hashlib.md5()
for f in files:
  # Unicode-objects must be encoded
before hashing
  m.update(f.encode('UTF-8', 'ignore'))
return(m.digest())
```

it is recommended to use sha256 instead of md5 because it supplies security in many ways.
The security of SHA-256 is due to three factors (sha-256-encryption, n.d.). First, reconstructing the original data from the hash value is nearly impossible.To generate the initial data, a brute-force attack would need to conduct 2256 attempts.Second, a collision (two messages with the same hash value) is exceedingly improbable With 2256 potential hash values (more than the number of atoms in the known universe), the chances of two hash values matching are infinitesimally minuscule.

```
m = hashlib.sha256()
for f in files:
  # Unicode-objects must be encoded
before hashing
  m.update(f.encode('UTF-8', 'ignore'))
return(m.digest())
```

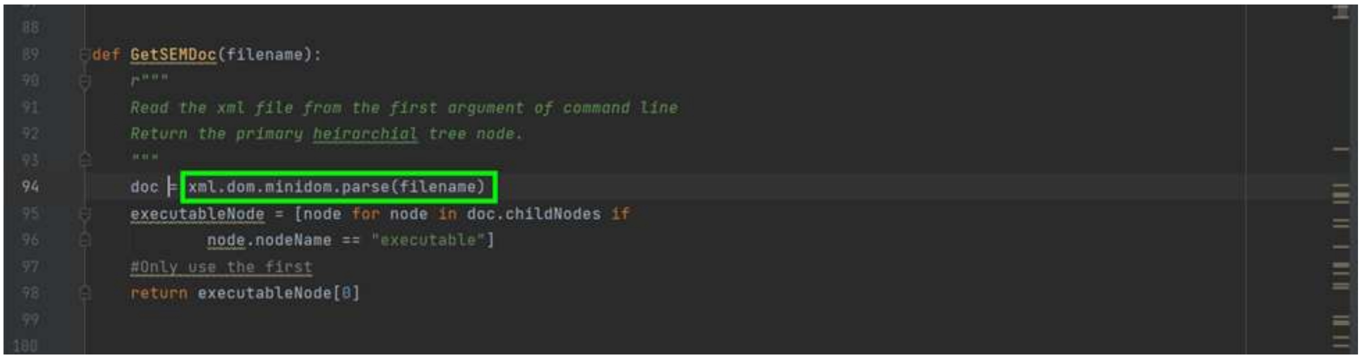


```
    try:
      import hashlib
    except:
      return None
    m = hashlib.md5()
    for f in files:
      # Unicode-objects must be encoded before hashing
      m.update(f.encode('UTF-8', 'ignore'))
    return(m.digest())

  def getCachedLoadables(self, files):
    """ Helper method to access the results of a previous
```

now the hashing algorithm is changed to sha256.

C. *Insecure Xml Parser*

An XML External Entity (XXE) attack can be launched by parsing insecure XML files with a poorly configured XML parser.External entity references are used in this sort of attack to get access to random files on a system, perform denial of service, or forge server side requests.Out-of-band data gathering methods may allow attackers to obtain sensitive data even if the outcome of parsing is not provided to the user.In this circumstance, service denial is also an option (XML External Entity Prevention Cheat Sheet, n.d.).



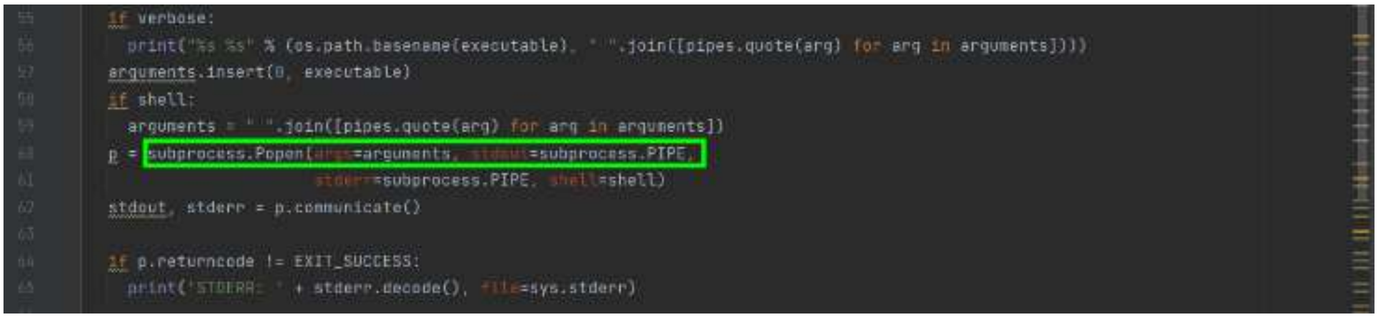
```
def GetSEMDoc(filename):
    r"""
    Read the xml file from the first argument of command line
    Return the primary heirarchial tree node.
    """
    doc = xml.dom.minidom.parse(filename)
    executableNode = [node for node in doc.childNodes if
            node.nodeName == "executable"]
    #Only use the first
    return executableNode[0]
```

Fix

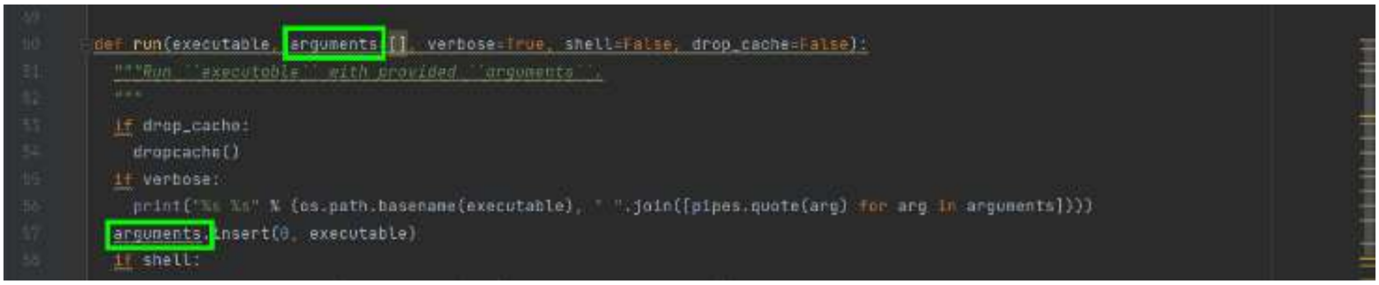
Disabling the parsing of any Document Type Declarations (DTDs) in untrusted data is the best technique to prevent XXE assaults. External general entities and external parameter entities should be disabled if this is not possible.Although this increases security, the code remains vulnerable to denial of service and server-side request forgery attacks.Setting entity growth limitations, which is done by default in modern JDK and JRE implementations, can also be used to protect against denial of service attacks.We chose to specifically check for the OWASP recommended technique to prevent external entity retrieval for a particular parser in this query because there are numerous possible ways to disable external entity retrieval with varying support amongst various providers.Other methods of making a parser safe that do not follow these criteria may exist, in which case this query will continue to mark the parser as possibly dangerous (Resolving XML external entity in user-controlled data, n.d.).

D. *Command Injection*

A cyber assault involving the execution of arbitrary commands on a host operating system is known as command injection (OS). Typically, the threat actor injects the orders via taking advantage of an application flaw, such as a lack of input validation (Command Injection, n.d.).
For example, a threat actor can inject a command into a web server's system shell using insecure transmissions of user data like cookies and forms. The attacker can then compromise the server using the privileges of the vulnerable application.Direct execution of shell commands, injecting malicious files into a server's runtime environment, and exploiting vulnerabilities in configuration files, such as XML external entities, are all examples of command injection (XXE) (Command Injection, n.d.).



```
  if verbose:
    print("%s %s" % (os.path.basename(executable), " ".join([pipes.quote(arg) for arg in arguments])))
  arguments.insert(0, executable)
  if shell:
    arguments = " ".join([pipes.quote(arg) for arg in arguments])
  p = subprocess.Popen(args=arguments, stdout=subprocess.PIPE,
                       stderr=subprocess.PIPE, shell=shell)
  stdout, stderr = p.communicate()

  if p.returncode != EXIT_SUCCESS:
    print('STDERR: ' + stderr.decode(), file=sys.stderr)
```



```
def run(executable, arguments=[], verbose=True, shell=False, drop_cache=False):
  """Run ``executable`` with provided ``arguments``.
  """
  if drop_cache:
    dropcache()
  if verbose:
    print("%s %s" % (os.path.basename(executable), " ".join([pipes.quote(arg) for arg in arguments])))
  arguments.insert(0, executable)
  if shell:
```

This attack varies from Code Injection in that it allows the attacker to inject their own code into the program, which is subsequently executed. In Command Injection, the attacker enhances the application's default capability by executing system commands without having to inject code.

Fix

Never calling out to OS commands from application-layer code is by far the most effective technique to prevent OS command injection issues. In almost every scenario, there are safer platform APIs that can be used to implement the desired functionality (How To Prevent Command Injection, n.d.). If using user-supplied input to call out to OS commands is unavoidable, then strong input validation is required. Here are some examples of effective validation:

- Checking against a whitelist of acceptable values.
- Checking to see if the input is a number.
- Checking for only alphanumeric characters and no other syntax or whitespace in the input.

Attempting to sanitize input by escaping shell metacharacters is a bad idea. In fact, this is far too error-prone and vulnerable to a skilled attacker's bypass.

IV. CONCLUSION

This activity discussed the healthcare-related slicing software, which is a 3D slicer, and how it can be analyzed for vulnerabilities and code bugs. The bug fixing process consists of information gathering about the software and its functionality, code analysis, searching for more information about the bugs and solutions, and finally, bug fixing. In the first stage, we use some tools to identify vulnerabilities in the source code after identifying the vulnerable points in the source code. We have to gather more information about that particular vulnerability and a good solution. In the final stage, we implemented those solutions into the source code.

V. REFERENCE

3D Slicer

*A retrospective impact analysis of the WannaCry cyberattack on the NHS*2019

*A survey on various cyber attacks and their classification*2013

*Alaska DHSS facing potential breach after two Trojan malware attacks*2017

*Analyzing Man-in-the-Browser (MITB) Attacks*2015sans whitepaper

*Attack and Defend: Linux Privilege Escalation Techniques of 2016*2017

build instructions

California hospital makes rare admission of hack, ransom payment

CISA2009*Understanding Denial-of-Service Attacks*

cisco*What Is the Difference: Viruses, Worms, Trojans, and Bots?*

*Classification of Security Threats in Information Systems*2014

Command Injection

Command Injection

*common vectors cyber attacks*2009

*cryptography*britannica

*Cybersecurity in healthcare: A systematic review of modern threats and trends*2017San MarcosTexas StateUSA

Cybersecurity in Hospitals: A Systematic, Organizational Perspective

*Cybersecurity in Hospitals: A Systematic, Organizational Perspective*2018

*Cyber-Security Issues in Healthcare Information Technology*2017springerlink

*Cybersecurity Risks in a Pandemic*2020pubmed

Cybersecurity vulnerabilities in medical devices2015

Davis.J2017*Hackers breach New York's largest provider with phishing attacks*

*Delays in Emergency Care and Mortality during Major U.S. Marathons*2017

*Digital Tracing during the COVID-19 Pandemic: User Appraisal, Emotion, and Continuance Intention*2021researchgate

*Do Hospital Data Breaches Reduce Patient Care Quality?*2019

*Do Hospital Data Breaches Reduce Patient Care Quality?*2019

*Emerging Cyber Threats and the Challenges Associated with them*2018

Epiphany cardio server is vulnerable to SQL and LDAP injection2015

*Exploring the Cybersecurity Measures Healthcare Managers Use to Reduce Patient Endangerment Resulting from Backdoor Intrusions into Medical Devices*2019

*Health Care and Cybersecurity: Bibliometric Analysis of the Literature*2019

HealthITSecurity2017NY Computer Virus Raises Healthcare Data Security Concerns

*Hollywood hospital pays $17,000 in bitcoin to hackers; FBI investigating*2016

How To Prevent Command Injection

How to Prevent Web Attacks Using Input Sanitization

Hummel.R2017*Securing Against the Most Common Vectors of Cyber Attacks*

Input Sanitization

insecure-default-password-hashing-cms

*Investigation: WannaCry cyber attack and the NHS*2019

*Lessons learned review of the WannCry ransomware cyber attack*2018London

Merriam-Webster

Merriam-Webster*spyware*Merriam-Webster

NIST2013*Glossary of key information security terms*

Primary health care

*Quantum Spy Games*2014science.org

*Ransomware Case Studies: Hollywood Presbyterian & The Ottawa Hospital*2016

*Recognizing and avoiding spyware*2009

Resolving XML external entity in user-controlled data

*risk management-based security evaluation model for telemedicine systems*2020

Sanitize Your Inputs?

*Securing cyber resilience in health and care*2019

sha-256-encryption

*Social Engineering Attacks During the COVID-19 Pandemic*2021

symantec2016*What is the difference between viruses, worms, and Trojans?*

*The Digital Transformation of Healthcare Technology Management*2018
*Transforming Healthcare Cybersecurity from Reactive to Proactive: Current Status and Future Recommendations*2020researchagate
US-CERT
user_guide
*WannaCry, Cybersecurity and Health Information Technology: A Time to Act*2017pubmed
*What is a Man-in-the-Middle Attack?*2013Kaspersky
what is digital transformation
*What is Privilege Escalation?*2016ICANN
what-is-3D-slicer-software
what-is-a-3d-slicer-simply-explained
When 'Hacktivists' Target Your Hospital
XML External Entity Prevention Cheat Sheet