Launching into Cyber Security, MSc. Cyber Security, Module 1 – End of Module Project

This page presents the background, outline and code comprising my submission of the end of module project. Order of presentation is as follows:

1. Background scenario – describes the situation with client, Queens Medical Centre.
2. Technical Report (individual essay, also with separate link from module home page)
3. Code Outline – submitted for comment by tutor prior to commencing coding. This outline comprises proposed modules and theory of code to address a single issue from the technical report above. (automating the status report and notification of out of date security patches to overcome human error in keeping up to date)
4. A readme file providing an overview of the final approach and code exercise
5. The raw python code with explanatory comments
6. Run Results – documenting successful completion of intended processes and showing output data.
7. A log file demonstrating the successful data output to a designated logging file.
8. A second readme file (README.emailcode.txt) which shows code originally included but removed as it required actual email account login data displayed in code. This file explains the scenario and includes the code with the run results showing that the program attempted to send the requisite email but resulted in a login error as pseudo login data was used.

Project can be viewed on github.com at https://github.com/crypto61/LCYS-Project

1. Background Scenario

The aim of this deliverable is to apply the Cyber Security methods and techniques studied during the module to develop a solution to a business problem. The details of the business problem are as follows:

Queens medical centre is a community clinic and serves as the first point of call for any resident within the catchment area who happens to be unwell. The clinic has specialists in various areas of medicine. However, an appointment is required to schedule a consultation meeting with a specialist, which is done through a telephone call to the receptionist.

The clinic has been experiencing a high volume of calls, causing a lot of problems for residents to get access to care on time. Also, the management of Queens medical centre needs to plan to be able to respond to the rate of growth of the community population.

To address this problem, the clinic management has decided to acquire a web-based appointment and scheduling management information system (ASMIS). This will allow appointments to be booked online by prospective patients. The system will collect vital information from the patient to be able to determine which specialist is best to attend to a given case, considering the availability and workload of the specialist doctor.

The management of the clinic is concerned about the recent high rate of cybercrime and the government's policy on patient data protection. The IT team at the clinic has been tasked to install a secured ASMIS. As the Cyber Security Officer on the team that manages the clinic's IT systems, you are to advise the team on the potential cyber threats and how these can be mitigated.

So, you are to produce a report to management providing the following information:

- Details of the benefits of the ASMIS as well as the potential problems including potential cyber threats to the system.
- The report must include **at least two** UML diagrams. Each diagram should show aspects of the system and a threat modelling technique that can be used to identify and mitigate potential cyber threats
- A background to the UML diagrams and modelling techniques, along with justifications for their use with supporting references.
- The Cyber Security technologies that can be used to address the problem, discussing the strengths and weaknesses of the solutions with references to examples in which the technology has been employed.

2. Technical Report (Essay)

As this report is quite long for inclusion here, you may find it at the following link:
https://crypto61.github.io/eportfolio/module_essay.pdf

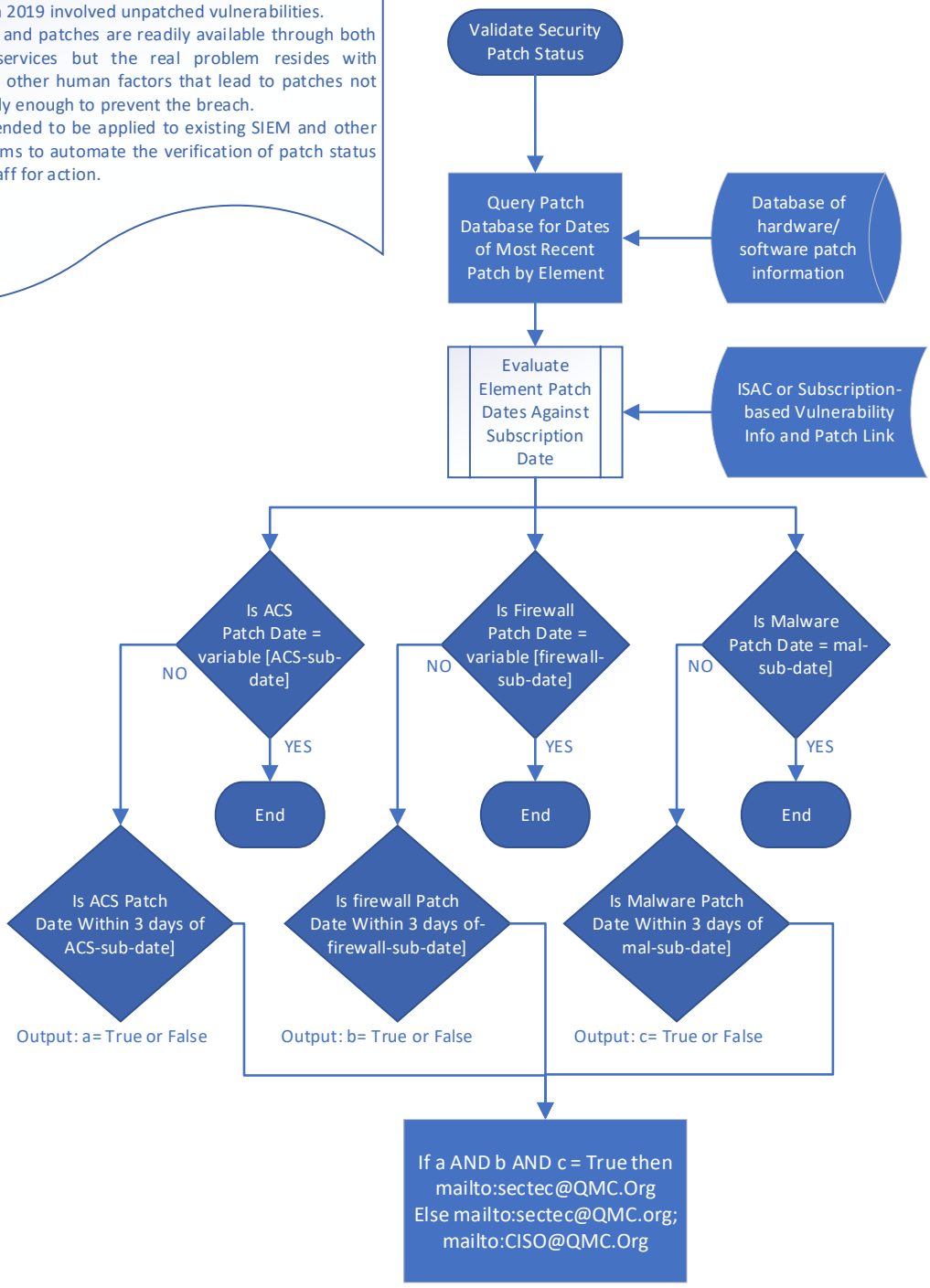
3. Code Outline

Problem Statement

Failure to implement timely updates to security patches is a leading, contributing factor to preventable data breaches. According to (Truta, 2019), 60% of breaches in 2019 involved unpatched vulnerabilities.

Vulnerability information and patches are readily available through both free and subscription services but the real problem resides with workforce shortages and other human factors that lead to patches not being implemented rapidly enough to prevent the breach.

This modular code is intended to be applied to existing SIEM and other security monitoring systems to automate the verification of patch status and notify appropriate staff for action.



4. README.txt

End of Course Assessment Project for LYCS.

Presents python code for addressing one security issue as presented in unit 9 essay (Queen's Medical Centre)

Comments on each section within code. Run results are in file: lcysproject_proof_of_operation.txt.

References used for code syntax verification are at bottom of code section.

Begin lcysproject.py

LYCS Module Assessment - Use python to create code to address one issue identified
in module 9 essay - Queen's Medical Centre, Secure web-based appointment system.
Selected issue is human error in forgetting to update firewall security patches.
Code intends to address the issue by automating a daily script to examine the latest patch date for 2 firewalls
generate warning messages to a log file and send an email to staff.

Approach:

Query last patch date from firewall system file
Determine patch expiration date based on a mandatory 30-day refresh cycle.
Compare today's date to expiration date for each firewall.
Print status of each firewall to console for confirmation
If firewall patch is out of date, send log warning msg to "file.log"

Originally included code to also send an email to CISO and security tech.
But this only worked without error when I included my real email server details including passwords.
So, I have removed the email code to provide a clean run output file free of errors and easy to see proof of concept.
However, to document learning experience that went into that effort, I am including a copy of the code
(with generic email info) for review in README_emailcode.txt.

 # Begin Code

```
from datetime import date, timedelta
today = date.today()


# representing the patch date of firewall 1 pulled (simulated) from firewall 1 system file.

p1 = date (2021, 7, 15)

# representing the patch date of firewall 2 pulled (simulated) from firewall 2 system file.
```

```
p2 = date (2021, 5, 15)

# This calculates the expiration date for security patch upgrade based on 30 days maximum life of patch.

exp1 = (p1 + timedelta(days = 30))
exp2 = (p2 + timedelta(days = 30))


# Prints date info to console to confirm process has completed and is in expected range.
# contains a string description and variable output

print ()
print ("today's date is", (today))
print ()
print ("firewall 1 patch date is", (p1))
print ()
print ("firewall 2 patch date is", (p2))
print ()
print ("Firewall 1 expiration date is", (exp1))
print ()
print ("Firewall 2 expiration date is", (exp2))
print ()


## Enables logging

import logging

# Creates a custom logger

logger = logging.getLogger(__name__)

# Creates handlers (in this case only file handlers as console output is called directly from the code)
# file handler creates file.log and writes event data as specified in msg creation throughout code. logger.warning("string")


f_handler = logging.FileHandler('file.log')
f_handler.setLevel(logging.WARNING)


# Creates formatters and add it to handlers

f_format = logging.Formatter('%(asctime)s - %(name)s - %(levelname)s - %(message)s')
f_handler.setFormatter(f_format)


# Add handlers to the logger as defined in 2 steps above

logger.addHandler(f_handler)
```

```
## provides operations to compare security patch expiration date to today's date and generate output process accordingly.
# variables used have been defined in first section

if today < exp1:
  print ("Patch status of firewall 1 is current", (today))
  print ()
elif today > exp1:
  print ("PATCH STATUS OF FIREWALL 1 IS OUT OF DATE AND REQUIRES IMMEDIATE ATTENTION", (today))
  logger.warning("PATCH STATUS OF FIREWALL 1 IS OUT OF DATE AND REQUIRES IMMEDIATE ATTENTION")


if today < exp2:
  print ("Patch status of firewall 2 is current", (today))
  print ()
elif today > exp2:
  print ("PATCH STATUS OF FIREWALL 2 IS OUT OF DATE AND REQUIRES IMMEDIATE ATTENTION", (today))
  logger.warning("PATCH STATUS OF FIREWALL 1 IS OUT OF DATE AND REQUIRES IMMEDIATE ATTENTION")


## References:

# General, I followed my own original code outline (attached). For style and syntax guide I referred to the following resources:

        Python 3.9.6 documentation available from docs.python.org/3/ [accessed July 2021]

        PEP 8 -- Style Guide for Python Code available from python.org/dev/peps/pep-0008/ [accessed July 2021]

# General, I discovered some of my resources were based on a different version of python and would result in an error. For determining source of the error, I used https://www.tutorialsteacher.com/python/error-types-in-python [accessed July 2021]

# Timedelta syntax help for adding specific number of days to a given date: https://www.geeksforgeeks.org/python-datetime-timedelta-function/ [Accessed July 2021]

# For error resolution with a logging configuration line: https://realpython.com/python-logging/#the-logging-module [Accessed July 2021]

# For send an email code (in separate file - "README_emailcode.txt") base code quoted from: doc.python.org/3/library/email.examples.html [accessed July 2021] code modified to specific needs of this project.
```

5. Raw python code

```
from datetime import date, timedelta
today = date.today()
p1 = date (2021, 7, 15)
p2 = date (2021, 5, 15)
```

```
exp1 = (p1 + timedelta(days = 30))
exp2 = (p2 + timedelta(days = 30))

print ()
print ("today's date is", (today))
print ()
print ("firewall 1 patch date is", (p1))
print ()
print ("firewall 2 patch date is", (p2))
print ()
print ("Firewall 1 expiration date is", (exp1))
print ()
print ("Firewall 2 expiration date is", (exp2))
print ()


import logging

logger = logging.getLogger(__name__)

f_handler = logging.FileHandler('file.log')
f_handler.setLevel(logging.WARNING)
f_format = logging.Formatter('%(asctime)s - %(name)s - %(levelname)s - %(message)s')
f_handler.setFormatter(f_format)
logger.addHandler(f_handler)

if today < exp1:
  print ("Patch status of firewall 1 is current", (today))
  print ()
elif today > exp1:
  print ("PATCH STATUS OF FIREWALL 1 IS OUT OF DATE AND REQUIRES IMMEDIATE
ATTENTION", (today))
  logger.warning("PATCH STATUS OF FIREWALL 1 IS OUT OF DATE AND REQUIRES
IMMEDIATE ATTENTION")

 if today < exp2:
  print ("Patch status of firewall 2 is current", (today))
  print ()
elif today > exp2:
  print ("PATCH STATUS OF FIREWALL 2 IS OUT OF DATE AND REQUIRES IMMEDIATE
ATTENTION", (today))
  logger.warning("PATCH STATUS OF FIREWALL 1 IS OUT OF DATE AND REQUIRES
IMMEDIATE ATTENTION")
```

6. Run Results

Below is output of lcysproject.py when run in Codio. Run results for logging can be seen in file.log.

```
 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage
```

```
 * Super-optimized for small spaces - read how we shrank the memory footprint of MicroK8s to
make it the smallest full K8s around.

   https://ubuntu.com/blog/microk8s-memory-optimisation

 * Canonical Livepatch is available for installation.
   - Reduce system reboots and improve kernel security. Activate at:
     https://ubuntu.com/livepatch
 *
 * Welcome to the Codio Terminal!
 *
 * https://docs.codio.com/project/ide/boxes/#overview
 *
 * Your Codio Box domain is: right-cuba.codio.io
 *
Last login: Thu Jul 22 17:39:20 2021 from 192.168.10.156
codio@right-cuba:~/workspace$ python3 lycsproject.py

today's date is 2021-07-22

firewall 1 patch date is 2021-07-15

firewall 2 patch date is 2021-05-15

Firewall 1 expiration date is 2021-08-14

Firewall 2 expiration date is 2021-06-14

Patch status of firewall 1 is current 2021-07-22

PATCH STATUS OF FIREWALL 2 IS OUT OF DATE AND REQUIRES IMMEDIATE
ATTENTION 2021-07-22
codio@right-cuba:~/workspace$
```

7. File.log

```
PATCH STATUS OF FIREWALL 2 IS OUT OF DATE AND REQUIRES IMMEDIATE
ATTENTION 2021-07-22
```

8. README.emailcode.txt

```
## Below is run result from email Module. This worked with my real email server inserted.
## Pulled from main program for security reasons. Below demonstrates that module attempts
## to send email but connection is rejected due to no login provided.

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

 * Super-optimized for small spaces - read how we shrank the memory
   footprint of MicroK8s to make it the smallest full K8s around.

   https://ubuntu.com/blog/microk8s-memory-optimisation
```

```
 * Canonical Livepatch is available for installation.
   - Reduce system reboots and improve kernel security. Activate at:
     https://ubuntu.com/livepatch
 *
 * Welcome to the Codio Terminal!
 *
 * https://docs.codio.com/project/ide/boxes/#overview
 *
 * Your Codio Box domain is: right-cuba.codio.io
 *
Last login: Thu Jul 22 20:18:53 2021 from 192.168.10.156
codio@right-cuba:~/workspace$ python3 email
Traceback (most recent call last):
  File "email", line 17, in <module>
    s = smtplib.SMTP('localhost')
  File "/usr/lib/python3.6/smtplib.py", line 251, in __init__
    (code, msg) = self.connect(host, port)
  File "/usr/lib/python3.6/smtplib.py", line 336, in connect
    self.sock = self._get_socket(host, port, self.timeout)
  File "/usr/lib/python3.6/smtplib.py", line 307, in _get_socket
    self.source_address)
  File "/usr/lib/python3.6/socket.py", line 724, in create_connection
    raise err
  File "/usr/lib/python3.6/socket.py", line 713, in create_connection
    sock.connect(sa)
ConnectionRefusedError: [Errno 111] Connection refused
codio@right-cuba:~/workspace$
```