17 June 2020

|  |  |  |  |
| --- | --- | --- | --- |
| Date: | 17 June 2020 | Name: | Srinidhi J C |
| Course: | **Introduction to Cyber Security** | USN: | 4al16ec078 |
| Topic: | |  | | --- | | Vulnerabilities & Password Security | | What is Cryptography? | | Message integrity | | Semester & Section: | 8th & b |
| Github Repository: | SrinidhiJC078 |  |  |
| FORENOON SESSION DETAILS | | | | | |
| Image of session        A screenshot of a cell phone  Description automatically generated | | | | | |

Report:

Mistakes happen, even in the process of building and coding technology. What’s left behind from these mistakes is commonly referred to as a bug. While bugs aren’t inherently harmful (except to the potential performance of the technology), many can be taken advantage of by nefarious actors—these are known as vulnerabilities. Vulnerabilities can be leveraged to force software to act in ways it’s not intended to, such as gleaning information about the current security defences in place.

Once a bug is determined to be a vulnerability, it is registered by MITRE as a [CVE](https://cve.mitre.org/), or common vulnerability or exposure, and assigned a Common Vulnerability Scoring System (CVSS) score to reflect the potential risk it could introduce to your organization. This central listing of CVEs serves as a reference point for [vulnerability scanners](https://www.rapid7.com/products/insightvm/).

Generally speaking, a vulnerability scanner will scan and compare your environment against a [vulnerability database](https://www.rapid7.com/db/), or a list of known vulnerabilities; the more information the scanner has, the more accurate its performance. Once a team has a report of the vulnerabilities, developers can use [penetration testing](https://www.rapid7.com/fundamentals/penetration-testing/) as a means to see where the weaknesses are, so the problem can be fixed and future mistakes can be avoided. When employing frequent and consistent scanning, you'll start to see common threads between the vulnerabilities for a better understanding of the full system.[Learn more about vulnerability management and scanning here.](https://www.rapid7.com/fundamentals/vulnerability-management-and-scanning/)

## Security Vulnerability Examples

A Security Vulnerability is a weakness, flaw, or error found within a security system that has the potential to be leveraged by a threat agent in order to compromise a secure network.

There are a few Security Vulnerabilities, but some common examples are:

* **Broken Authentication:** When authentication credentials are compromised, user sessions and identities can be hijacked by malicious actors to pose as the original user.
* **SQL Injection:** As one of the most prevalent security vulnerabilities, SQL injections attempt to gain access to database content via malicious code injection. A successful SQL injection can allow attackers to steal sensitive data, spoof identities, and participate in a collection of other harmful activities.
* **Cross-Site Scripting:** Much like an SQL Injection, a Cross-site scripting (XSS) attack also injects malicious code into a website. However, a Cross-site scripting attack targets website users, rather than the actual website itself, which puts sensitive user information at risk of theft.
* **Cross-Site Request Forgery:** A Cross-Site Request Forgery (CSRF) attack aims to trick an authenticated user into performing an action that they do not intend to do. This, paired with social engineering, can deceive users into accidentally providing a malicious actor with personal data.
* **Security Misconfiguration:** Any component of a security system that can be leveraged by attackers due to a configuration error can be considered a “Security Misconfiguration.”

Passwords are hacked in a variety of ways and for a variety of reasons.

Sometimes the hacker is someone you know. It could be a “frenemy” who wants to access your information. If so, he or she might be able to guess your password if you use one that’s as obvious as your dog’s name or child’s name. Also, if you are not using a proven [password management solution](https://www.avatier.com/blog/category/password-management/) like Avatier, an insider may be able to access your account if the answers to your password recovery questions are known. If a hacker is unknown to you, a brute force attack is the most common strategy for cracking your password. In this method, a program systematically tries every password combination until it gains access. That is why the simpler your password is, the easier it is for someone to gain access.

The Art of the Difficult Password

A difficult password is your stronghold against hackers. Examples of high-security passwords include:

* K5,747.;Sj24f9m/
* -UP8cjCS!`S8″rA8
* sP5}V97^Wru&E:!

A difficult password is created by using variables, such as upper and lower cases, numbers, and symbols, along with a password length of at least 16 characters.  
When creating your password, don’t include dictionary words, usernames or IDs, repetition, any predefined number or letter sequence, your birthday, people’s names, personal information (license plate number, phone number), words or phrases from popular culture (Ihaveadream; Game\_over,\_man!), or dictionary words with simple algorithms applied (backwards spelling, combining words with punctuation in between).

With many complex passwords for your many different accounts, how can you keep it all straight? Some experts suggest using a phrase as a keycode. For example, the password “T5&10i@M&ASt.s” could be remembered using the keycode “The 5-and-10 is at Main and Ash Streets.” Similarly, the password “LaIsBo11#187!” could be remembered with this sentence about a favorite song: “Madonna’s La Isla Bonita, which spent 11 weeks at #1 on the music charts in 1987.”

Additionally, when we each have so many user accounts, password managers become useful tools to help keep your passwords straight. Another option is to store your passwords on a USB thumb drive.

Some even suggest you go low-tech and write your passwords on paper or Post-its. Surprisingly, it’s less risky than using the same password on multiple sites, re-using old passwords, using easy-to-guess passwords, letting your software remember your passwords, or not frequently changing your passwords.

Cryptography is a method of protecting information and communications through the use of codes, so that only those for whom the information is intended can read and process it. The prefix "crypt-" means "hidden" or "vault" -- and the suffix "-graphy" stands for "writing."

In computer science, cryptography refers to secure information and communication techniques derived from mathematical concepts and a set of rule-based calculations called algorithms, to transform messages in ways that are hard to decipher. These deterministic algorithms are used for cryptographic key generation, digital signing, verification to protect data privacy, web browsing on the internet, and confidential communications such as credit card transactions and email.

### Cryptography techniques

### Cryptography is closely related to the disciplines of [cryptology](https://searchsecurity.techtarget.com/definition/cryptology) and [cryptanalysis](https://searchsecurity.techtarget.com/definition/cryptanalysis). It includes techniques such as microdots, merging words with images, and other ways to hide information in storage or transit. However, in today's computer-centric world, cryptography is most often associated with scrambling [plaintext](https://searchsecurity.techtarget.com/definition/plaintext) (ordinary text, sometimes referred to as cleartext) into [ciphertext](https://whatis.techtarget.com/definition/ciphertext) (a process called [encryption](https://searchsecurity.techtarget.com/definition/encryption)), then back again (known as decryption). Individuals who practice this field are known as cryptographers.

### Modern cryptography concerns itself with the following four objectives:

1. **Confidentiality**: the information cannot be understood by anyone for whom it was unintended
2. **Integrity:**the information cannot be altered in storage or transit between sender and intended receiver without the alteration being detected
3. **Non-repudiation**: the creator/sender of the information cannot deny at a later stage his or her intentions in the creation or transmission of the information
4. **Authentication**: the sender and receiver can confirm each other's identity and the origin/destination of the information

Message Integrity and Digital Signatures go hand in hand in our daily communication. Here we talk about **message integrity**.

If Bob receives a message (either be encrypted or be in plaintext) from Alice, he needs to verify:

1. The message indeed originated from Alice.
2. The message was not tempered with on its way to Bob.

Before talking about how to achieve the above goals, we need to introduce cryptographic hash functions.

A hash function takes an input, m, and computes a fixed-size string H(m). A cryptographic hash function also needs to satisfy an additional property:

* It is computationally infeasible to find any two different messages x and y such that H(x) = H(y).

Common hash algorithms are MD5 [RFC 1321] and SHA-1 [FIPS 1995].

After we are having the cryptographic hash function. A naive step to perform message integrity would be:

1. Alice creates message m and send Bob (m, H(m)).
2. After Bob receives (m, H(m)), he uses the same hash function to check whether after hashing m, the result is equal to the H(m) he received from Alice.

The approach is flawed because other party can also send message to bob (m’, H(m’)) claiming that this is from Alice, and Bob has no way to tell even the hash result is correct.

In order to prove the message is indeed from Alice, Alice and Bob should have a shared secret s. This shared secret is called **authentication key**. Using this shared secret, message integrity can be performed as:

1. Alice creates message m, concatenates s to m, calculates H(m+s), sends Bob (m, H(m+s)). H(m+s) here is called the **message authentication code (MAC)**.
2. Bob receives (m, H(m+s)), calculates the MAC using his s, check whether it is the same H(m+s) he received from the message.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Date: | 17 June 2020 | Name: | Srinidhi J C | |
| Course: | IOT in Python with Raspberry Pi | USN: | 4al16ec078 | |
| Topic: | Atmosphere Monitoring System | Semester & Section: | 8th & b | |
| AFTERNOON SESSION DETAILS | | | |
| Image of session | | | |

Report:

The Raspberry Pi acts as a central server that runs the following components:

* **balenaOS** is a minimal Linux-based OS which mainly packages balenaEngine, an engine designed to run Docker-like containers. [BalenaCloud](https://www.balena.io/cloud/) allows us to push applications to the Raspberry Pi and other devices fully remotely and over the air. Their cloud platform is free for up to 10 devices and lets you push updates to applications running in containers on devices.
* **Sensor** is the python library and code for accessing the sensor, taking readings and saving them in the database.
* [**InfluxDB**](https://www.influxdata.com/) is a time-series database.
* [**Grafana**](https://grafana.com/) is an open-source platform for data visualization. Grafana reads the data in **InfluxDB** and manages the dashboard to visualize such information.

A combination of these tools can provide you with outstanding monitoring and analysis abilities of your environment monitoring dashboard. There are many options to install both of these tools separately or together, but the most efficient is by using the **Docker** technology. It takes just a few minutes to spin up these tools together.

### Hardware required

The components you will be required for this project are as follows:

* Raspberry Pi board
* **CJMCU-680** module
* A 16GB or more microSD card.
* Power Adapter for Raspberry Pi.
* A laptop or desktop computer for flashing BalenaOS image on the SD card.

### Software required

* A tool to flash your SD card, such as [balenaEtcher](https://www.balena.io/etcher/)

### The BME680 sensor

The **CJMCU-680** module is based on the BME680 sensor from Bosch. It allows you to measure environmental parameters such as gas content in the air, as well as pressure, humidity and air temperature.

It also allows you to detect volatile organic compounds (VOC) in the air, so it can be used to monitor gases and alcohols, such as ethyl alcohol, carbon monoxide and perform air quality measurements. At the same time, the sensor cannot measure the content of a specific gas or alcohol, since it gives the value of the total content of VOC compounds in the air.

BME680 has **I2C** and **SPI** interfaces for working with the microcontrollers.

There are 6 pins on the module:

* Vcc - Voltage input pin
* GND - Ground pin
* SCL - SPI Clock pin
* SDO - Serial Data Out pin
* SDA - Serial Data In pin
* CS - Chip Select Pin for SPI communication

### Schematic connection

The BME680 sensor communicates with the Raspberry Pi over a bus called I2C which is a serial communication bus that requires 2 wires. These two communication wires are referred to as serial clock (SCL) and serial data (SDA). In addition to the two communication wires, we also need to provide the sensor with power (3.3V) and ground.

The 40 pin GPIO header is standard across the Pi 2 and later. You will need to connect 4 pins with the Raspberry PI Board:

* SDA ==> pin 3
* SCL ==> pin 5
* Vcc ==> pin 1
* GND ==> pin 9

Here's the my Raspberry PI 4 Board connected with BME680 sensor:

### Configuring your Raspberry Pi for I2C(optional)

This section can be skipped. However, if you want to understand the basic python code to retrieve data from the BME680 sensor, you can proceed with it.

Make sure you have the latest version of the Raspbian operating system You can download the latest version from the official Raspberry [website.](https://www.raspberrypi.org/downloads/) I’ve already published a quick start [guide](https://www.hackster.io/shahizat005/getting-started-with-the-raspberry-pi-4-desktop-kit-834625) recently. Check it out.

After installing the OS, we will check whether the latest drivers are installed with the following commands.

sudo apt-get update

This command updates the list of available packages and their versions.

sudo apt-get upgrade

I2C Tools is a package with useful console tools for I2C. After installation, several useful utilities will be available to us: **i2cdetect**, **i2cdump**, **i2cget** and **i2cset**.

sudo apt-get install i2c-tools

Python code:

import bme680  
import time  
import datetime  
from datetime import datetime  
  
try:  
 sensor = bme680.BME680(bme680.I2C\_ADDR\_PRIMARY)  
except IOError:  
 sensor = bme680.BME680(bme680.I2C\_ADDR\_SECONDARY)  
  
sensor.set\_humidity\_oversample(bme680.OS\_2X)  
sensor.set\_pressure\_oversample(bme680.OS\_4X)  
sensor.set\_temperature\_oversample(bme680.OS\_8X)  
sensor.set\_filter(bme680.FILTER\_SIZE\_3)  
  
try:  
 while True:  
 if sensor.get\_sensor\_data():  
 output = datetime.now().strftime('%Y-%m-%d,%H:%M:%S,')+'{0:.2f} C,{1:.2f} hPa,{2:.3f} %RH'.format(  
 sensor.data.temperature,  
 sensor.data.pressure,  
 sensor.data.humidity)  
 print(output)  
 time.sleep(1)  
except KeyboardInterrupt:  
 pass

Type the following command in the terminal

sudo python3 bme680\_sensor.py