Question – 1: Imagine you are a cybersecurity analyst working for a large multinational corporation. One morning, your team receives an urgent report about a potential security breach in the company's network. The IT department has noticed unusual network activity originating from a particular IP address. Your team has been tasked with investigating this incident to determine if it poses a threat to the organization's network security.

Assignment Question: 1. Using the Python library Scapy, analyze the network packets associated with the suspicious IP address provided.

Expected Procedure:

1. A detailed explanation of how Scapy can be utilized to capture and dissect network packets.

2. A step-by-step breakdown of the process you followed to capture and analyze the network traffic.

3. Identification and interpretation of any suspicious or anomalous network behavior observed in the captured packets.

4. Recommendations for mitigating the identified security risks and securing the network against similar threats in the future.

Expected Code: 1. Write a python code to Network Packet Analysis with Scapy

Answer: To perform network packet analysis using the Python library Scapy, follow these steps:

- **Install Scapy:** If you haven't already installed Scapy, you can install it using

  pip install scapy

- **Capturing and analysing Network packets:** Write Python code to capture and analyze network packets to identify any suspicious behaviour. Code:

```
from scapy.all import *

def analyze_packet(packet):
    if IP in packet:
        print("Source IP:", packet[IP].src)
        print("Destination IP:", packet[IP].dst)
        print("Protocol:", packet[IP].proto)
    if packet[IP].src == suspicious_ip:
        print("Suspicious packet detected:")
        print(packet.summary())
sniff(prn=analyze_packet, filter="ip", count=10)
```

- **Interpretation:** Interpret the captured packets based on their content and structure. Pay attention to source and destination IP addresses, protocols etc.
- **Mitigation Recommendations:** Blocking traffic from the suspicious IP address, updating firewall rules, implementing intrusion detection/prevention systems, and enhancing network monitoring capabilities can be followed to mitigate the identified security risks and securing the network against similar threats in the future.

Expected Code:

```
pip install scapy

from scapy.all import *

def analyze_packet(packet):
    if IP in packet:
        print("Source IP:", packet[IP].src)
        print("Destination IP:", packet[IP].dst)
        print("Protocol:", packet[IP].proto)
    if packet[IP].src == suspicious_ip:
        print("Suspicious packet detected:")
        print(packet.summary())
sniff(prn=analyze_packet, filter="ip", count=10)
```

Question - 2: Imagine you are working as a cybersecurity analyst at a prestigious firm. Recently, your company has been experiencing a surge in cyber-attacks, particularly through phishing emails and websites. These attacks have not only compromised sensitive information but also tarnished the reputation of the company. In light of these events, your team has been tasked with developing a robust solution to detect and mitigate phishing websites effectively. Leveraging your expertise in Python programming and cybersecurity, your goal is to create a program that can accurately identify phishing websites based on various features and indicators.

Assignment Task:

Using the Python programming language, develop a phishing website detection system that analyzes website characteristics and determines the likelihood of it being a phishing site.

Expected Procedure:

1. Accept 2 web URL. One real and another one phishing.

2. Analyze the data from both the websites.

3. Identify the phishing site.

Expected Code:

1. Phishing Website Detection with Python

Answer: To create a phishing website detection system in Python, we use machine learning algorithms to analyse and classify them as phishing or legitimate.

Expected code:

```
import requests

from bs4 import BeautifulSoup

from sklearn.feature_extraction.text import TfidfVectorizer
```

```
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score

def extract_text(url):
    try:
        response = requests.get(url)
        soup = BeautifulSoup(response.text, 'html.parser')
        text = ‘ ’.join([p.text for p in soup.find_all('p')])
        return text
    except:
        return None

phishing_url = “https://example-phishing.com”
legitimate_url = “https://example-legitimate.com”
phishing_text = extract_text(phishing_url)
legitimate_text = extract_text(legitimate_url)

if phishing_text and legitimate_text:
    texts = [phishing_text, legitimate_text]
    labels = [‘phishing’, ‘legitimate’]

    vectorizer = TfidfVectorizer()
    X = vectorizer.fit_transform(texts)

    X_train, X_test, y_train, y_test = train_test_split(X, labels, test_size=0.2, random_state=42)

    classifier = RandomForestClassifier(n_estimators=100)
    classifier.fit(X_train, y_train)
    y_pred = classifier.predict(X_test)

    accuracy = accuracy_score(y_test, y_pred)
    print(“Accuracy:”, accuracy)
```

```
    test_url = “https://new-test-website.com”
    test_text = extract_text(test_url)
    if test_text:
        test_features = vectorizer.transform([test_text])
        prediction = classifier.predict(test_features)[0]
        print(“Predicted label for the test website:”, prediction)
    else:
        print(“Unable to extract text from the test website.”)
else:
    print(“Failed to extract text from one of the websites.”)
```

This code extracts text content from provided URLs, vectorizes the text using TF-IDF, trains the Random Forest classifier, and then identifies if a website is phishing or legitimate based on its text features.