University of Mumbai

PRACTICAL JOURNAL – PAPER I

PSIT4P1a

Blockchain

SUBMITTED BY

Vijay Hanumant Patil

SEAT NO 40434

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR QUALIFYING M.Sc. (I.T.) PART-II (SEMESTER – IV) EXAMINATION

2022-2023

Department of Information Technology

3RD FLOOR, DR. SHANKAR DAYAL SHARMA BHAVAN, IDOL BUILDING, VIDYANAGRI,

SANTACRUZ (E), MUMBAI – 400098.

University of Mumbai

Department of Information Technology

CERTIFICATE

This is to certify that **Mr. Vijay Hanumant Patil, Seat No. 40434 studying** in **Master of Science in Information Technology Part II Semester IV** has satisfactorily completed the Practical of **PSIT4P1a Blockchain** as prescribed by University of Mumbai, during the academic year **2022-23**.

Signature
Subject-In-Charge

Signature
Head of the Department

Signature External
Examiner

College Seal

Date:

INDEX

Practical No.	Date	Name of Practical	Page No.	Signature
1	18/04/2023	Write the following programs for Blockchain in Python: A] A simple client class that generates the private and public keys by using the built in Python RSA algorithm and test it. B] A transaction class to send and receive money and test it. C] Create multiple transactions and display them. D] Create a blockchain, a genesis block and execute it. E] Create a mining function and test it. F] Add blocks to the miner and dump the blockchain.	4	
2	18/04/2023	Install and configure Go Ethereum and the Mist browser. Develop and test a sample application.	19	
3	18/04/2023	Implement and demonstrate the use of the following in Solidity: A] Variable, Operators, Loops, Decision Making, Strings, Arrays, Enums, Structs, Mappings, Conversions, Ether Units, Special Variables. B] Functions, Function Modifiers, View functions, Pure Functions, Fallback Function, Function Overloading, Mathematical functions, Cryptographic functions.	25	
4	18/04/2023	Implement and demonstrate the use of the following in Solidity: A] Withdrawal Pattern, Restricted Access. B] Contracts, Inheritance, Constructors, Abstract Contracts, Interfaces. C] Libraries, Assembly, Events, Error handling.	61	

5	23/05/2023	Install hyperledger fabric and composer. Deploy and execute the application.	85	
6	22/06/2023	Write a program to demonstrate mining of Ether.	90	
7	22/06/2023	Demonstrate the running of the blockchain node.	92	
8	22/06/2023	Demonstrate the use of Bitcoin Core API.	94	
9	23/06/2023	Create your own blockchain and demonstrate its use.	96	

PRACTICAL 1

AIM: Write the following programs for Blockchain in Python:

[A] A simple client class that generates the private and public keys by using the builtin Python RSA algorithm and test it.

CODE:

```
import binascii
from Crypto.PublicKey import RSA
from Crypto.Signature import PKCS1_v1_5
from Crypto import Random


class Client:
    def __init__(self):
        random = Random.new().read
        self._private_key = RSA.generate(1024, random)
        self._public_key = self._private_key.publickey()
        self._signer = PKCS1_v1_5.new(self._private_key)

    @property
    def identity(self):
        return
binascii.hexlify(self._public_key.exportKey(format="DER")).decode("a
scii")

# Create an instance of the Client class
UDIT = Client()

# Print the public key (identity) of the clientq
print("\nPublic Key:", UDIT.identity)
```

OUTPUT:

```
Public Key:
 30819f300d06092a864886f70d010101050003818d0030818902818100c9b694d2aa1855dbbad947
 5aac327ab784b7ae7a14b81fbe416e5886f492dff2fb71a80cdb194d212a11523f42271c4a962410
 ce6bd9dcf28cbd43aa9fe8a6da2a54345ebbcf3facd1213fcc800a34ded20309db83389299c8bb83
 2532b28a281fef07971421d56740a95ddfd789cdaafad97aea839584e3d92aee5664ec42d7020301
 0001
```

[B] A transaction class to send and receive money and test it.

CODE:

```
import binascii
import collections
import datetime
from Crypto.Hash import SHA
from Crypto.PublicKey import RSA
from Crypto.Signature import PKCS1_v1_5
from Crypto import Random

class Client:
    def __init__(self):
        random = Random.new().read
        self._private_key = RSA.generate(1024, random)
        self._public_key = self._private_key.publickey()
        self._signer = PKCS1_v1_5.new(self._private_key)

    @property
    def identity(self):
        return
binascii.hexlify(self._public_key.exportKey(format="DER")).decode("a
scii")

class Transaction:
    def __init__(self, sender, recipient, value):
        self.sender = sender
        self.recipient = recipient
        self.value = value
        self.time = datetime.datetime.now()

    def to_dict(self):
        identity = "Genesis" if self.sender == "Genesis" else
self.sender.identity
        return collections.OrderedDict(
            {
                "sender": identity,
                "recipient": self.recipient,
```

```
                "value": self.value,
                "time": self.time,
            }
        )

    def sign_transaction(self):
        private_key = self.sender._private_key
        signer = PKCS1_v1_5.new(private_key)
        h = SHA.new(str(self.to_dict()).encode("utf8"))
        return binascii.hexlify(signer.sign(h)).decode("ascii")

UDIT = Client()
UGC = Client()

t = Transaction(UDIT, UGC.identity, 5.0)

print("\nTransaction     Recipient:\n",     t.recipient)          #
print("\nTransaction   Sender:\n",   t.sender)   print("\nTransaction
Value:\n", t.value)
signature = t.sign_transaction()
print("\nSignature:\n", signature)
```

OUTPUT:

```
Transaction Recipient:
 30819f300d06092a864886f70d010101050003818d0030818902818100be0978593e24a777060c9b
  95c383a53f3e3213905fc67538a37f15e9c2c4bd6e0eb667999074079928c3e3d7f0636f0d7e9f8
  cc0bd7a38da76342c612c103fe43a603e97c602d1f2a0dbbe794ab983aa1d1062d70485c0e0c734
  3c265e90ae4094c6ede9037b0d38af530b27914df1a08c339ec4bc76de28b2dbe5cd6a0e3ebf020
  3010001

Signature:
 a3d6c2635b55cd891bc43dc63b137bfb10eaa68dd7ce276f8e098367cc83195f5a508fa2e4e2c74a
  341e143d38432b36e4dc240f5f4b17d07d5be3ebf67a59f42583c215e9d78059091580629eee052
  985b2bf355ab58eae226b41dbfaa9690bfce61745392e9f172173ce5a2f12040b753610468ca4a3
  fccb1437968a11fc00
```

[C] Create multiple transactions and display them.

CODE:

```
import binascii
import collections
import datetime
from Crypto.Hash import SHA
from Crypto.PublicKey import RSA
from Crypto.Signature import PKCS1_v1_5
from Crypto import Random


class Client:
    def __init__(self):
        random = Random.new().read
        self._private_key = RSA.generate(1024, random)
        self._public_key = self._private_key.publickey()
        self._signer = PKCS1_v1_5.new(self._private_key)

    @property
    def identity(self):
        return
binascii.hexlify(self._public_key.exportKey(format="DER")).decode("a
scii")


class Transaction:
    def __init__(self, sender, recipient, value):
        self.sender = sender
        self.recipient = recipient
        self.value = value
        self.time = datetime.datetime.now()

    def to_dict(self):
        identity = "Genesis" if self.sender == "Genesis" else
self.sender.identity
        return collections.OrderedDict(
```

```
            {
                "sender": identity,
                "recipient": self.recipient,
                "value": self.value,
                "time": self.time,
            }
        )

    def sign_transaction(self):
        private_key = self.sender._private_key
        signer = PKCS1_v1_5.new(private_key)
        h = SHA.new(str(self.to_dict()).encode("utf8"))
        return binascii.hexlify(signer.sign(h)).decode("ascii")

    def display_transaction(transaction):
        # for transaction in transactions:
        dict = transaction.to_dict()
        print("sender: " + dict['sender'])
        print('-----')
        print("recipient: " + dict['recipient'])
        print('-----')
        print("value: " + str(dict['value']))
        print('-----')
        print("time: " + str(dict['time']))
        print('-----')

UDIT = Client()
UGC = Client()
AICTE = Client()
MU = Client()

t1 = Transaction(UDIT, UGC.identity, 15.0)
t1.sign_transaction()
transactions = [t1]
t2 = Transaction(UDIT, AICTE.identity, 6.0)
t2.sign_transaction()
transactions.append(t2)
```

```
t3 = Transaction(UGC, MU.identity, 2.0)
t3.sign_transaction()
transactions.append(t3)
t4 = Transaction(AICTE, UGC.identity, 4.0)
t4.sign_transaction()
transactions.append(t4)
for transaction in transactions:
    Transaction.display_transaction(transaction)
    print("      ")
```

OUTPUT:

```
sender:
 30819f300d06092a864886f70d010101050003818d00308189028181009c383a3fc7
 248a85bb871be70bdad857db28da890de9de76f3df85950fa0452ab1615bf6b7967f
 28c642e433a04335befdb4232c0735d271a0949a28e51e9154f327c7edd6be6e5588
 73c12afbd66a48c6fc726515789d1109438f75446829b3832be14e894a40f27e1331
 cc48e536a30cc47a1039b4d87364a3ac2c3f7553110203010001
-----
recipient:
 30819f300d06092a864886f70d010101050003818d00308189028181009a597e8496e
 f09e903eac0b9f97d8bfde76565f58599206bbd47fb020b0d0daef0568449ef7cb44
 68abe58a30e7877276404a5264840f4e0ddda570cbefec408c459d58429573427694
 382caa972f3878f7ae04ff27bbea057bf9360dd65e193eaf8301c5168840e6a55812
 867b746de1da1695c5130d49cbee519c6cb23698710203010001
-----
value: 15.0
-----
time: 2023-06-25 18:38:53.536480
-----
```

```
sender:
 30819f300d06092a864886f70d010101050003818d00308189028181009c383a3fc7
 248a85bb871be70bdad857db28da890de9de76f3df85950fa0452ab1615bf6b7967f
 28c642e433a04335befdb4232c0735d271a0949a28e51e9154f327c7edd6be6e5588
 73c12afbd66a48c6fc726515789d1109438f75446829b3832be14e894a40f27e1331
 cc48e536a30cc47a1039b4d87364a3ac2c3f7553110203010001
-----
recipient:
 30819f300d06092a864886f70d010101050003818d00308189028181009cfc0b7e441
 5f62ea994a001e1d30f8d23d89ec17062f162c5f2ac56c4c5883ace946a59854562d
 116efa15269b50a0d180e8b46db923ec942c347826037007c3b638bd666a4949e773
 bc98ba3ff5e0f4fa06e97f0d31986ee2090e02a4ba9dd60951d48e8253763a08730b
 0c4e4f3aae04ac568b9c9708406a5c9564d03ace250203010001
-----
value: 6.0
-----
time: 2023-06-25 18:38:53.538479
-----
```

```
sender:
 30819f300d06092a864886f70d010101050003818d00308189028181009a597e8496e
 f09e903eac0b9f97d8bfde76565f58599206bbd47fb020b0d0daef0568449ef7cb44
 68abe58a30e7877276404a5264840f4e0ddda570cbefec408c459d58429573427694
 382caa972f3878f7ae04ff27bbea057bf9360dd65e193eaf8301c5168840e6a55812
 867b746de1da1695c5130d49cbee519c6cb23698710203010001
-----
recipient:
 30819f300d06092a864886f70d010101050003818d00308189028181009a51bb5b3ba
 ef116ce6aaf25937457f3b789787be44b0512795ee48ab373b93af9d103ebd77ec1e
 de442c493867c4a8fad6b50dc5ee1cc7daec59e0615d855f04b45c4f35796194185d
 e848b59ab6ae4ca9946ceab649192b998708db6d9c62927ff4516e1d330f67fa5e4a
 6c32148c6b0206686ef15e234c85a8c70366c33e910203010001
-----
value: 2.0
-----
time: 2023-06-25 18:38:53.539479
-----
```

```
sender:
 30819f300d06092a864886f70d010101050003818d0030818902818100cfc0b7e441
 5f62ea994a001e1d30f8d23d89ec17062f162c5f2ac56c4c5883ace946a59854562d
 116efa15269b50a0d180e8b46db923ec942c347826037007c3b638bd666a4949e773
 bc98ba3ff5e0f4fa06e97f0d31986ee2090e02a4ba9dd60951d48e8253763a08730b
 0c4e4f3aae04ac568b9c9708406a5c9564d03ace250203010001
-----
recipient:
 30819f300d06092a864886f70d010101050003818d0030818902818100a597e8496e
 f09e903eac0b9f97d8bfde76565f58599206bbd47fb020b0d0daef0568449ef7cb44
 68abe58a30e7877276404a5264840f4e0ddda570cbefec408c459d58429573427694
 382caa972f3878f7ae04ff27bbea057bf9360dd65e193eaf8301c5168840e6a55812
 867b746de1da1695c5130d49cbee519c6cb2369871020301000 1
-----
value: 4.0
-----
time: 2023-06-25 18:38:53.540479
-----
```

[D] Create a blockchain, a genesis block and execute it.

CODE:

```
import binascii
import collections
import datetime
from Crypto.Hash import SHA
from Crypto.PublicKey import RSA
from Crypto.Signature import PKCS1_v1_5
from Crypto import Random

class Client:
    def __init__(self):
        random = Random.new().read
        self._private_key = RSA.generate(1024, random)
        self._public_key = self._private_key.publickey()
        self._signer = PKCS1_v1_5.new(self._private_key)

    @property
    def identity(self):
        return
binascii.hexlify(self._public_key.exportKey(format="DER")).decode("a
scii")

class Transaction:
    def __init__(self, sender, recipient, value):
        self.sender = sender
        self.recipient = recipient
        self.value = value
        self.time = datetime.datetime.now()

    def to_dict(self):
        identity = "Genesis" if self.sender == "Genesis" else
self.sender.identity
        return collections.OrderedDict(
            {
                "sender": identity,
```

```
                "recipient": self.recipient,
                "value": self.value,
                "time": self.time,
            }
        )

    def sign_transaction(self):
        private_key = self.sender._private_key
        signer = PKCS1_v1_5.new(private_key)
        h = SHA.new(str(self.to_dict()).encode("utf8"))
        return binascii.hexlify(signer.sign(h)).decode("ascii")

    def display_transaction(transaction):
        # for transaction in transactions:
        dict = transaction.to_dict()
        print("sender: " + dict['sender'])
        print('-----')
        print("recipient: " + dict['recipient'])
        print('-----')
        print("value: " + str(dict['value']))
        print('-----')
        print("time: " + str(dict['time']))
        print('-----')

class Block:
    def __init__(self, client):
        self.verified_transactions = []
        self.previous_block_hash = ""
        self.Nonce = ""
        self.client = client

def dump_blockchain(blocks):
    print(f"\nNumber of blocks in the chain: {len(blocks)}")
    for i, block in enumerate(blocks):
        print(f"block # {i}")
        for transaction in block.verified_transactions:
            Transaction.display_transaction(transaction)
```

```
            print("   ")
    print("      ")

UDIT = Client()
t0 = Transaction("Genesis", UDIT.identity, 500.0)
block0 = Block(UDIT)
block0.previous_block_hash = ""
NONCE = None
block0.verified_transactions.append(t0)
digest = hash(block0)
last_block_hash = digest
TPCoins = [block0]
dump_blockchain(TPCoins)
```

OUTPUT:

```
Number of blocks in the chain: 1
block # 0
sender: Genesis
-----
recipient:
 30819f300d06092a864886f70d010101050003818d0030818902818100eca6bb5563
 9066584a301764d24f95860798f3cc060ee433ea8aa72458506e8c279812b5415f5d
 a25d966065b3eadd12dfa6bd819411253bf9883a11947b7a093a52c47afe7fff165a
 454859d297d02a9d0baadff621af0ceefb302183762305aa53c66c6681940f8fb9c3
 05dfdd132cfb49e6d24ee01578043f591e6b28b7910203010001
-----
value: 500.0
-----
time: 2023-06-25 18:44:33.262135
-----
```

[E] Create a mining function and test it.

CODE:

```
import hashlib


def sha256(message):
    return hashlib.sha256(message.encode("ascii")).hexdigest()


def mine(message, difficulty=1):
    assert difficulty >= 1
    prefix = "1" * difficulty
    for i in range(1000):
        digest = sha256(str(hash(message)) + str(i))
        if digest.startswith(prefix):
            print(f"After {str(i)} iterations found nonce: {digest}")
            return digest

print(mine("test message", 2))
```

OUTPUT:

```
After 247 iterations found nonce:
 114ad9945d4a3b191d654352691375a6ce71606ba0e08ea737f9427b7a81d9f0
114ad9945d4a3b191d654352691375a6ce71606ba0e08ea737f9427b7a81d9f0
```

[F] Add blocks to the miner and dump the blockchain.

CODE:

```
import datetime
import hashlib
# Create a class with two functions
class Block:
  def __init__(self, data, previous_hash):
    self.timestamp = datetime.datetime.now(datetime.timezone.utc)
    self.data = data
    self.previous_hash = previous_hash
    self.hash = self.calc_hash()
  def calc_hash(self):
    sha = hashlib.sha256()
    hash_str = self.data.encode("utf-8")
    sha.update(hash_str)
    return sha.hexdigest()
# Instantiate the class
blockchain = [Block("First block", "0")]
blockchain.append(Block("Second block", blockchain[0].hash))
blockchain.append(Block("Third block", blockchain[1].hash))
# Dumping the blockchain
for block in blockchain:
  print(
f"Timestamp:  {block.timestamp}\nData:  {block.data}\nPrevious  Hash:
{block.previous_hash}\nHash: {block.hash}\n"
  )
```

OUTPUT:

```
Timestamp: 2023-06-25 13:18:33.950453+00:00
Data: First block
Previous Hash: 0
Hash: 876fb923a443ba6afe5fb32dd79961e85be2b582cf74c233842b630ae16fe4d9

Timestamp: 2023-06-25 13:18:33.950453+00:00
Data: Second block
Previous Hash:
 876fb923a443ba6afe5fb32dd79961e85be2b582cf74c233842b630ae16fe4d9
Hash: 8e2fb9e02898feb024dff05ee0b27fd5ea0a448e252d975e6ec5f7b0a252a6cd

Timestamp: 2023-06-25 13:18:33.950453+00:00
Data: Third block
Previous Hash:
 8e2fb9e02898feb024dff05ee0b27fd5ea0a448e252d975e6ec5f7b0a252a6cd
Hash: 06e369fbfbe5362a8115a5c6f3e2d3ec7292cc4272052dcc3280898e3206208d
```

PRACTICAL 2

AIM: Install and configure Go Ethereum and the Mist browser. Develop and test a sample application.

Step 1: Go to Chrome Web Store Extensions Section.

Step 2: Search MetaMask.

Step 3: Check the number of downloads to make sure that the legitimate MetaMask is being installed, as hackers might try to make clones of it.

Step 4: Click the Add to Chrome button.

Step 5: Once installation is complete this page will be displayed. Click on the Get Started button.

Step 6: This is the first time creating a wallet, so click the Create a Wallet button. If there is already a wallet then import the already created using the Import Wallet button.

Step 7: Click I Agree button to allow data to be collected to help improve MetaMask or else click the No Thanks button. The wallet can still be created even if the user will click on the No Thanks button.

Step 8: Create a password for your wallet. This password is to be entered every time the browser is launched and wants to use MetaMask. A new password needs to be created if chrome is uninstalled or if there is a switching of browsers. In that case, go through the Import Wallet button. This is because MetaMask stores the keys in the browser. Agree to Terms of Use.

Step 9: Click n the dark area which says Click here to reveal secret words to get your secr

Step 10: This is the most important step. Back up your secret phrase properly. Do not store your secret phrase on your computer. Please read everything on this screen until you understand it completely before proceeding. The secret phrase is the only way to access your wallet if you forget your password. Once done click the Next button.

Step 11: Click the buttons respective to the order of the words in your seed phrase. In other words, type the seed phrase using the button on the screen. If done correctly the Confirm button should turn blue.

Step 12: Click the Confirm button. Please follow the tips mentioned.

Step 13: One can see the balance and copy the address of the account by clicking on the Account 1 area.

Step 14: One can access MetaMask in the browser by clicking the Foxface icon on the top right. If the Foxface icon is not visible, then click on the puzzle piece icon right next to it.

PRACTICAL 3

AIM: Implement and demonstrate the use of the following in Solidity:

[A] Variable, Operators, Loops, Decision Making, Strings, Arrays, Enums, Structs, Mappings, Conversions, Ether Units, Special Variables.

[I] Variable

[i] State Variable

CODE:

```
//Solidity program to demonstrate state variables
pragma solidity ^0.5.0;
// Creating a contract
contract var_Test
{
    // Declaring a state variable
    uint8 public state_var;

    // Defining a constructor
    constructor() public {
        state_var = 16;
    }
}
```

OUTPUT:

[ii] Local Variable

CODE:

```
//Solidity program to demonstrate Local variables
pragma solidity ^0.5.0;
// Creating a contract
contract local_var_Test
{
    // Defining function to show the declaration and
    // scope of Local variables
    function acsess_local_variable() public pure returns(uint) {
        // Initializing Local variables
        uint a = 10;
        uint b = 40;
        uint sum = a + b;
        // Access the Local variable
        return sum;
    }
}
```

OUTPUT:

[iii] Global Variable

CODE:

```
//Solidity program to show Global variables
pragma solidity ^0.5.0;
// Creating a contract
contract globalTest
{
    // Defining a variable
    address public admin;
    // Creating a constructor to
    // use Global variable
    constructor() public
    {
        admin = msg.sender;
    }
}
```

OUTPUT:

[II] Operators

CODE:

```
pragma solidity ^0.4.0;
contract Operators {
        uint256 result = 0;
    function addition(uint256 a, uint256 b) public pure returns (uint256) {
        return a + b;
    }
    function subtraction(uint256 a, uint256 b) public pure returns (uint256)
{
        return a - b;
    }
    function division(uint256 a, uint256 b) public pure returns (uint256) {
        return a / b;
    }
    function multiply(uint256 a, uint256 b) public pure returns (uint256) {
        return a * b;
    }
}
```

OUTPUT:

division
a: 15
b: 3
Calldata
Parameters
call
0: uint256: 5

multiply
a: 25
b: 4
Calldata
Parameters
call
0: uint256: 100

subtraction
a: 23
b: 12
Calldata
Parameters
call
0: uint256: 11

[III] Loops

CODE:

```
pragma solidity ^0.5.0;

contract LoopingTest {
    uint256 storedData;

    constructor() public {
        storedData = 10;
    }

    function getResult() public pure returns (string memory) {
        uint256 a = 10;
        uint256 b = 2;
        uint256 result = a + b;
        return integerToString(result);
    }

    function integerToString(uint256 _i) internal pure returns (string
memory) {
        if (_i == 0) {
            return "0";
        }
        uint256 j = 0;
        uint256 len;
        for (j = _i; j != 0; j /= 10) {
            //for loop example
            len++;
        }
        bytes memory bstr = new bytes(len);
        uint256 k = len - 1;
        while (_i != 0) {
            bstr[k--] = bytes1(uint8(48 + (_i % 10)));
            _i /= 10;
        }
        return string(bstr); //access local variable
    }
}
```

OUTPUT:

LOOPINGTEST AT 0XD7A...F771B (MEI
Balance: 0 ETH
getResult
0: string: 12

[IV] Decision Making

CODE:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.1;
// Creating a contract
contract Types {
        // Declaring state variables
    uint256 i = 10;
    string result;

    function decision_making() public payable returns (string memory) {
        if (i < 10) {
            result = "less than 10";
        }
        else if (i == 10) {
            result = "equal to 10";
        }
        else {
            result = "greater than 10";
        }
        return result;
    }
}
```

OUTPUT:

[V] Strings

[i] Double Quotes

CODE:

```
pragma solidity ^0.5.0;

contract testString
{
    function test_string() public pure returns (string memory)
    {
        string memory a = "hello";
        return a;
    }
}
```

OUTPUT:

[ii] Single Quotes

CODE:

```
pragma solidity ^0.5.2;
contract stringTest
{
    function getResult() public pure returns(string memory) {
        uint a = 25;
        uint b = 25;
        uint result = a + b;
        return integerToString(result);
    }
    function integerToString(uint _i) internal pure returns (string memory){
        if (_i == 0)
        {
            return "0";
        }
        uint j = _i;
        uint len;
        while (j != 0)
        {
            len++;
            j/= 10;
        }
        bytes memory bstr = new bytes(len);
        uint k = len - 1;
        while (_i != 0)
        {
            bstr[k-- ] = byte(uint8(48 + _i % 10));
            _i /= 10;
        }
    return string(bstr);
    }
}
```

OUTPUT:

STRINGTEST AT 0X5FD...9D88D (MEM
Balance: 0 ETH
getResult
0: string: 50

[VI] Arrays

CODE:

```
// SPDX-License-Identifier: MIT
pragma solidity 0.7.0;

contract Arrays {

    function initArray() public pure returns (uint256) {
        uint128[3] memory array = [1, 2, uint128(3)];
        return array[0];
    }

    function dynamicArray(uint256 a, uint256 b) public pure returns (uint256)
{

        uint128[] memory array = new uint128[](a);
        uint128 val = 5;

        for (uint128 j = 0; j < a; j++) {
            array[j] = j * val;
        }

        return array[b];
    }
}
```

OUTPUT:

ARRAYS AT 0X7B9...B6ACE (MEMORY
Balance: 0 ETH
dynamicArray
a: 30
b: 10
Calldata
Parameters
call
0: uint256: 50
initArray
0: uint256: 1

[VII] Enums

CODE:

```
pragma solidity ^0.5.0;

contract enumTest {
    enum FreshJuiceSize {
        SMALL,
        MEDIUM,
        LARGE
    }
    FreshJuiceSize choice;
    FreshJuiceSize constant defaultChoice = FreshJuiceSize.MEDIUM;

    function setLarge() public {
        choice = FreshJuiceSize.LARGE;
    }

    function getChoice() public view returns (FreshJuiceSize) {
        return choice;
    }

    function getDefaultChoice() public pure returns (uint256) {
        return uint256(defaultChoice);
    }
}
```

OUTPUT:

ENUMTEST AT 0X332...D4B6D (MEM
Balance: 0 ETH
setLarge
getChoice
0: uint8: 2
getDefaultCho
0: uint256: 1

[VIII] Structs

STEPS:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.1;
contract test {
    struct Book {
        string title;
        string author;
        uint book_id;
    }
    Book book;
    function setBook() public {
        book = Book('Learn Java', 'TP', 1);
    }
    function getBookId() public view returns (uint) {
        return book.book_id;
    }
}
```

OUTPUT:

[IX] Mappings

CODE:

```
pragma solidity ^0.5.0;

contract LedgerBalance
{
    mapping(address => uint) public balances;

    function updateBalance(uint newBalance) public
    {
    balances[msg.sender] = newBalance;
    }
}
contract Updater
{
    function updateBalance() public returns (uint)
    {
        LedgerBalance ledgerBalance = new LedgerBalance();
        ledgerBalance.updateBalance(10);
        return ledgerBalance.balances(address(this));
    }
}
```

OUTPUT:

With Original Balance: 10

With Updated Balance: 100

LEDGERBALANCE AT 0X93F...C96CC (
Balance: 0 ETH
updateBalance
newBalance:
100
Calldata
Parameters
transact
balances
:
0x5B38Da6a701c568545dCfcB03FcB
Calldata
Parameters
call
0: uint256: 100

[X] Conversions

CODE:

```
pragma solidity ^0.4.0;
contract Conversions {
    function intToUint(int8 a) public pure returns (uint256) {
        uint256 b = uint256(a);
        return b;
    }
    function uint32ToUint16(uint32 a) public pure returns (uint16) {
        uint16 b = uint16(a);
        return b;
    }
}
```

OUTPUT:

[XI] Ether Units

CODE:

```
pragma solidity ^0.4.6;

contract SimpleStorage {
    uint256 storedData = 2;

    function set(uint256 x) public {
        storedData = x;
        /*
        Ether Units
        Wei
        Finney
        Szabo
        Ether
        */
        if (2000000000000000000 == 2 ether) {
            storedData = 2;
        }
        else {
            storedData = 3;
        }
        /*
        Time Units
        seconds
        minutes
        hours
        days
        weeks
        month
        years
        */
        if (120 seconds == 2 minutes) {
            storedData = 6;
        }
        else {
            storedData = 9;
        }
    }
```

```
    function get() constant public returns (uint256 retVal)
    {
        return storedData;
    }
}
```

OUTPUT:

[XII] Special Variables

CODE:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.1;

contract LedgerBalance {
    mapping(address => uint256) public balances;

    function updateBalance(uint256 newBalance) public {
        balances[msg.sender] = newBalance;
    }
}

contract Updater {
    function updateBalance() public returns (uint256) {
        LedgerBalance ledgerBalance = new LedgerBalance();
        ledgerBalance.updateBalance(10);
        return ledgerBalance.balances(address(this));
    }
}
```

OUTPUT:

Function Stack
0: updateBalance(newBalance) - 2313 gas
Solidity State
balances: mapping(address => uint256)
Solidity Locals
newBalance: 1000 uint256
Step details
vm trace step: 114
execution step: 114

[B] Functions, Function Modifiers, View Functions, Pure Functions, Fallback Function, Function Overloading, Mathematical functions, Cryptographic functions.

[I] Functions

CODE:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.5.0;

contract Test {
  function return_example()
    public
    pure
    returns (
      uint256,
      uint256,
      uint256,
      string memory
    )
  {
    uint256 num1 = 10;
    uint256 num2 = 16;
    uint256 sum = num1 + num2;
    uint256 prod = num1 * num2;
    uint256 diff = num2 - num1;
    string memory message = "Multiple return values";
    return (sum, prod, diff, message);
  }
}
```

OUTPUT:

TEST AT 0XF8E...9FBE8 (MEMORY)
Balance: 0 ETH
return_exampl
0: uint256: 26
1: uint256: 160
2: uint256: 6
3: string: Multiple return values

[II] Function Modifiers

CODE:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.5.0;

contract ExampleContract {
  address public owner = 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4;
  uint256 public counter;

  modifier onlyowner() {
    require(msg.sender == owner, "Only the contract owner can call");
    _;
  }

  function incrementcounter() public onlyowner {
    counter++;
  }
}
```

OUTPUT:

[III] View Functions

CODE:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.5.0;

contract view_demo {
  uint256 num1 = 2;
  uint256 num2 = 4;

  function getResult() public view returns (uint256 product, uint256 sum) {
    product = num1 * num2;
    sum = num1 + num2;
  }
}
```

OUTPUT:

[IV] Pure Functions

CODE:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.5.0;

contract pure_demo {
  function getResult() public pure returns (uint256 product, uint256 sum) {
    uint256 num1 = 2;
    uint256 num2 = 4;
    product = num1 * num2;
    sum = num1 + num2;
  }
}
```

OUTPUT:

[V] Fallback Function

CODE:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.5.0;

contract A {
  uint256 n;

  function set(uint256 value) external {
    n = value;
  }

  function() external payable {
    n = 0;
  }
}

contract example {
  function callA(A a) public returns (bool) {
    (bool success, ) = address(a).call(abi.encodeWithSignature("setter()"));
    require(success);
    address payable payableA = address(uint160(address(a)));
    return (payableA.send(2 ether));
  }
}
```

OUTPUT:

A AT 0XD91...39138 (MEMORY)
Balance: 0 ETH
set
value:
"7000"
Calldata
Parameters
transact

callA
a:
"0xd9145CCE52D386f254917e481eB
Calldata
Parameters
transact

[VI] Function Overloading

CODE:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.5.0;

contract OverloadingExample {
  function add(uint256 a, uint256 b) public pure returns (uint256) {
    return a + b;
  }

  function add(string memory a, string memory b)
    public
    pure
    returns (string memory)
  {
    return string(abi.encodePacked(a, b));
  }
}
```

OUTPUT:

add
a: 7
b: 2
Calldata
Parameters
call
0: uint256: 9
add
a: Hello
b: UDIT
Calldata
Parameters
call
0: string: HelloUDIT

[VII] Mathematical Functions

CODE:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.5.0;

contract Test{ function CallAddMod() public pure returns(uint){

    return addmod(7,3,3);

}

function CallMulMod() public pure returns(uint){

    return mulmod(7,3,3);

    }
}
```

OUTPUT:

[VIII] Cryptographic Functions

CODE:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.5.0;

contract Test{ function callKeccak256() public pure returns(bytes32 result){

  return keccak256("BLOCKCHAIN");



}

function callsha256() public pure returns(bytes32 result){

  return sha256("BLOCKCHAIN");



}

function callripemd() public pure returns (bytes20 result){

  return ripemd160("BLOCKCHAIN");

  }
}
```

OUTPUT:

TEST AT 0X0FC...9A836 (MEMORY)

Balance: 0 ETH

callKeccak256

0: bytes32: result 0xedb146af3dfbb7f995ec65b249dd88d2d54ca0707a84f08c25e4cc08f5168aea

callripemd

0: bytes20: result 0x638cf4481022e8be8fab43fa5f76ccffc62f2a09

callsha256

0: bytes32: result 0xdffdca1f7dd5c94afea2936253a2463a26aad06fa9b5f36b5affc8851e8c8d42

PRACTICAL 4

AIM: Implement and demonstrate the use of the following in Solidity:

[A] Withdrawal Pattern, Restricted Access.

[I] Withdrawal Pattern

CODE:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;

contract WithdrawalPattern {
  address public owner;
  uint256 public lockedbalance;
  uint256 public withdrawablebalance;

  constructor() {
    owner = msg.sender;
  }

  modifier onlyowner() {
    require(msg.sender  ==  owner,  "Only  the  owner  can  call  this
function");
    _;
  }

  function deposit(uint256 amount) public payable {
    require(amount > 0, "Amount must be greater than zero");
    lockedbalance += amount;
  }

  function withdraw(uint256 amount) public payable onlyowner {
    require(
      amount <= withdrawablebalance,
      "Insufficient withdrawable balance"
    );
    withdrawablebalance -= amount;
```

```
    payable(msg.sender).transfer(amount);
  }

  function unlock(uint256 amount) public onlyowner {
    require(amount <= lockedbalance, "Insufficient locked balance");
    lockedbalance -= amount;
    withdrawablebalance += amount;
  }
}
```

OUTPUT:

Step 1: Click on owner to create the owner object.

Step 2: Enter an amount and click on deposit.

Step 3: Click on locked balance button to display the locked amount in the account.

Step 4: Click on withdrawable balance button.

Step 5: Click on unlock button and enter any amount to transfer amount to withdrawable balance. Check locked balance and withdrawable balance.

Step 6: Enter any amount you want to withdraw and click the withdraw button. You should get an error and the transaction should be reverted.

[II] Restricted Access

CODE:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.5.0;
contract AccessRestriction {
    address public owner = msg.sender;
    uint public lastOwnerChange = now;

    modifier onlyBy(address _account) {
        require(msg.sender == _account);
        _;
    }

    modifier onlyAfter(uint _time) {
        require(now >= _time);
        _;
    }

    modifier costs(uint _amount) {
        require(msg.value >= _amount);
        _;
        if (msg.value > _amount) {
            msg.sender.transfer(msg.value - _amount);
        }
    }

    function changeOwner(address _newOwner) public onlyBy(owner) {
        owner = _newOwner;
    }

    function buyContract() public payable onlyAfter(lastOwnerChange +
4 weeks) costs(1 ether) {
        owner = msg.sender;
        lastOwnerChange = now;
    }
}
```

OUTPUT:

Step 1: Click on owner to create the owner object.

Step 2: Click on lastOwnerChange button.

Step 3: Change the address of the account from Account dropdown in Deploy tab of Remix IDE.

Step 4: Copy and paste the address in changeOwner input and click on changeOwner.

Step 5: You should get an error as following.

Step 6: Now, change back to the actual address of the account and click on changeOwner.

[B] Contracts, Inheritance, Constructors, Abstract Contracts, Interfaces.

[I] Contracts

CODE:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.5.0;

contract Contract_demo {
  string message = "Hello";

  function dispMsg() public view returns (string memory) {
    return message;
  }
}
```

OUTPUT:

[II] Inheritance

CODE:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.5.0;

contract Parent {
  uint256 internal sum;

  function setValue() external {
    uint256 a = 10;
    uint256 b = 20;
    sum = a + b;
  }
}

contract child is Parent {
  function getValue() external view returns (uint256) {
    return sum;
  }
}

contract caller {
  child cc = new child();

  function testInheritance() public returns (uint256) {
    cc.setValue();
    return cc.getValue();
  }

  function show_value() public view returns (uint256) {
    return cc.getValue();
  }
}
```

OUTPUT:

Step 1: Select caller contract and deploy.

Step 2: Click test Inheritance and then click on show_value to view value.

[III] Constructors

CODE:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.5.0;

// Creating a contract
contract constructorExample {
  string str;

  constructor() public {
    str = "UDIT";
  }

  function getValue() public view returns (string memory) {
    return str;
  }
}
```

OUTPUT:

[IV] Abstract Contracts

CODE:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.5.0;

contract Calculator {
  function getResult() external view returns (uint256);
}

contract Test is Calculator {
  constructor() public {}

  function getResult() external view returns (uint256) {
    uint256 a = 1;
    uint256 b = 2;
    uint256 result = a + b;
    return result;
  }
}
```

OUTPUT:

Step 1: Select Test contract and deploy.

Step 2: Click on getResult to get sum of a+b.

[V] Interfaces

CODE:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.5.0;

interface Calculator {
  function getResult() external view returns(uint);
}
contract Test is Calculator {
  constructor() public {}
  function getResult() external view returns(uint){
   uint a = 1;
   uint b = 2;
   uint result = a + b;
   return result;
  }
}
```

OUTPUT:

Step 1: Select Test interface contract and deploy.

Step 2: Click on getResult to get sum of a+b.

[C] Libraries, Assembly, Events, Error handling.

[I] Libraries

CODE:

Create library myLib.sol

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;

library myMathLib {
  function sum(uint256 a, uint256 b) public pure returns (uint256) {
    return a + b;
  }

  function exponent(uint256 a, uint256 b) public pure returns
(uint256) {
    return a**b;
  }
}
```

Using the library myLib.sol

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;

import "myLib.sol";

contract UseLib {
  function getsum(uint256 x, uint256 y) public pure returns (uint256)
{
    return myMathLib.sum(x, y);
  }

  function getexponent(uint256 x, uint256 y) public pure returns
(uint256) {
    return myMathLib.exponent(x, y);
  }
}
```

OUTPUT:

Step 1: Change contract to UseLib and deploy.

Step 2: Input values to both getexponent and getsum functions and get their values as below.

MYMATHLIB AT 0X820...7EDDE (MEM
Balance: 0 ETH
exponent
a: 2
b: 3
Calldata
Parameters
call
0: uint256: 8
sum
a: 2
b: 3
Calldata
Parameters
call
0: uint256: 5

[II] Assembly

CODE:

```
// SPDX-License-Identifier: MIT
pragma solidity >=0.4.16 <0.9.0;
contract InlineAssembly {
  // Defining function
  function add(uint256 a) public view returns (uint256 b) {
    assembly {
      let c := add(a, 16)
      mstore(0x80, c)
      {
        let d := add(sload(c), 12)
        b := d
      }
      b := add(b, c)
    }
  }
}
```

OUTPUT:

[III] Events

CODE:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.5.0;

// Creating a contract
contract eventExample {
  // Declaring state variables
  uint256 public value = 0;

  // Declaring an event
  event Increment(address owner);

  // Defining a function for logging event
  function getValue(uint256 _a, uint256 _b) public {
    emit Increment(msg.sender);
    value = _a + _b;
  }
}
```

OUTPUT:

Step 1: Input values to getValue function and get the result by clicking on the value button.

Step 2: In the terminal, check for logs.


```
logs                    [
                            {
                                "from": "0xd8b934580fcE35a11B58C6D73aDeE468a2833fa8",
                                "topic": "0xfc3a67c9f0b5967ae4041ed898b05ec1fa49d2a3c22336247201d71be6f97120",
                                "event": "Increment",
                                "args": {
                                    "0": "0x5B38Da6a701c568545dCfcB03FcB875f56beddC4",
                                    "owner": "0x5B38Da6a701c568545dCfcB03FcB875f56beddC4"
                                }
                            }
                        ]
```

[IV] Error handling

CODE:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.5.0;

contract ErrorDemo {
  function getSum(uint256 a, uint256 b) public pure returns (uint256)
{
    uint256 sum = a + b;
    require(sum < 255, "Invalid");
    assert(sum<255);
    return sum;
  }
}
```

OUTPUT:

Step 1: Input values to the getSum function.

Step 2: Check terminal panel.

```
CALL    [call] from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4 to: ErrorDemo.getSum(uint256,uint256) data: 0x8e8...0000a

call to ErrorDemo.getSum errored: VM error: revert.

revert
        The transaction has been reverted to the initial state.
Reason provided by the contract: "Invalid".
Debug the transaction to get more information.
```

PRACTICAL 5

AIM: Install hyperledger fabric and composer. Deploy and execute the application.

St 1: Install Docker on Ubuntu

sudo apt-get install curl

```
udit@udit-ubuntu: ~
udit@udit-ubuntu:~$ sudo apt-get install curl
[sudo] password for udit:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libcurl4
The following NEW packages will be installed:
  curl
The following packages will be upgraded:
```

curl -fSSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add –

```
udit@udit-ubuntu:~$ curl -fSSL https://download.docker.com/linux/ubuntu/gpg | s
udo apt-key add -
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0W
arning: apt-key is deprecated. Manage keyring files in trusted.gpg.d instead (s
ee apt-key(8)).
100  3817  100  3817    0     0  43852      0 --:--:-- --:--:-- --:--:-- 44383
OK
udit@udit-ubuntu:~$
```

sudo add-apt-repository \

"deb [arch=amd64] https://download.docker.com/linux/ubuntu \

$(lsb_release -cs) \ stable"

```
udit@udit-ubuntu:~$ sudo add-apt-repository \
   "deb [arch=amd64] https://download.docker.com/linux/ubuntu \
   $(lsb_release -cs) \
   stable"
Repository: 'deb [arch=amd64] https://download.docker.com/linux/ubuntu jammy st
able'
Description:
Archive for codename: jammy components: stable
More info: https://download.docker.com/linux/ubuntu
Adding repository.
Press [ENTER] to continue or Ctrl-c to cancel.
Adding deb entry to /etc/apt/sources.list.d/archive_uri-https_download_docker_c
om_linux_ubuntu-jammy.list
Adding disabled deb-src entry to /etc/apt/sources.list.d/archive_uri-https_down
load_docker_com_linux_ubuntu-jammy.list
Get:1 https://download.docker.com/linux/ubuntu jammy InRelease [48.9 kB]
```

sudo apt-get update

```
udit@udit-ubuntu:~$ sudo apt-get update
Hit:1 https://download.docker.com/linux/ubuntu jammy InRelease
Hit:2 http://in.archive.ubuntu.com/ubuntu jammy InRelease
Hit:3 http://security.ubuntu.com/ubuntu jammy-security InRelease
Hit:4 http://in.archive.ubuntu.com/ubuntu jammy-backports InRelease
Get:5 http://in.archive.ubuntu.com/ubuntu jammy-updates InRelease [119 kB]
Get:6 http://in.archive.ubuntu.com/ubuntu jammy-updates/main i386 Packages [430
 kB]
Get:7 http://in.archive.ubuntu.com/ubuntu jammy-updates/main amd64 Packages [72
4 kB]
Get:8 http://in.archive.ubuntu.com/ubuntu jammy-updates/main Translation-en [19
0 kB]
```

apt-cache policy docker-ce

```
udit@udit-ubuntu:~$ apt-cache policy docker-ce
docker-ce:
  Installed: (none)
  Candidate: 5:24.0.2-1~ubuntu.22.04~jammy
  Version table:
     5:24.0.2-1~ubuntu.22.04~jammy 500
        500 https://download.docker.com/linux/ubuntu jammy/stable amd64 Package
s
     5:24.0.1-1~ubuntu.22.04~jammy 500
        500 https://download.docker.com/linux/ubuntu jammy/stable amd64 Package
s
     5:24.0.0-1~ubuntu.22.04~jammy 500
        500 https://download.docker.com/linux/ubuntu jammy/stable amd64 Package
```

sudo apt-get install -y docker-ce

```
udit@udit-ubuntu:~$ sudo apt-get install -y docker-ce
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  containerd.io docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras
  docker-compose-plugin git git-man liberror-perl libslirp0 pigz slirp4netns
Suggested packages:
  aufs-tools cgroupfs-mount | cgroup-lite git-daemon-run
  | git-daemon-sysvinit git-doc git-email git-gui gitk gitweb git-cvs
  git-mediawiki git-svn
The following NEW packages will be installed:
  containerd.io docker-buildx-plugin docker-ce docker-ce-cli
  docker-ce-rootless-extras docker-compose-plugin git git-man liberror-perl
```

Step 2: Create a docker network

sudo docker network create udit-iroha-network

```
udit@udit-ubuntu:~$ sudo docker network create udit-iroha-network
508883d5de27b2b369c31cd428acc669783ca7eb6536eeaebcc77f324a7e4dec
```

Step 3: Add PostgreSQL to the created docker network.

sudo docker run --name some-postgres \

-e POSTGRES_USER=postgres \

-e POSTGRES_PASSWORD=mysecretpassword \

-p 5432:5432 \

--network=udit-iroha-network \

-d postgres:9.5

```
udit@udit-ubuntu:~$ sudo docker run --name some-postgres \
-e POSTGRES_USER=postgres \
-e POSTGRES_PASSWORD=mysecretpassword \
-p 5432:5432 \
--network=udit-iroha-network \
-d postgres:9.5
Unable to find image 'postgres:9.5' locally
9.5: Pulling from library/postgres
fa1690ae9228: Pull complete
a73f6e07b158: Pull complete
973a0c44ddba: Pull complete
07e5342b01d4: Pull complete
578aad0862c9: Pull complete
a0b157088f7a: Pull complete
6c9046f06fc5: Pull complete
ae19407bdc48: Pull complete
e53b7c20aa96: Pull complete
a135edcc0831: Pull complete
fed07b1b1b94: Pull complete
18d9026fcfbd: Pull complete
4d2d5fae97d9: Pull complete
d419466e642d: Pull complete
Digest: sha256:75ebf479151a8fd77bf2fed46ef76ce8d518c23264734c48f2d1de42b4eb40ae
Status: Downloaded newer image for postgres:9.5
b7e9907af8e45279b3ee8ba8cf96ed73945fec76c5fb32c3d6464f8196c84b2f
```

Step 4: Create a volume of persistant storage named "blockstore" to store the blocks on blockchain.

sudo docker volume create blockstore

```
udit@udit-ubuntu:~$ sudo docker volume create blockstore
blockstore
```

Step 5: Configure Iroha on the network. Download the Iroha code from github and install git.

sudo apt-get install git

```
udit@udit-ubuntu:~$ sudo apt-get install git
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
git is already the newest version (1:2.34.1-1ubuntu1.9).
git set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 245 not upgraded.
```

git clone -b develop https://github.com/hyperledger/iroha --depth=1

```
udit@udit-ubuntu:~$ git clone -b develop https://github.com/hyperledger/iroha -
-depth=1
Cloning into 'iroha'...
remote: Enumerating objects: 2037, done.
remote: Counting objects: 100% (2037/2037), done.
remote: Compressing objects: 100% (1772/1772), done.
remote: Total 2037 (delta 433), reused 784 (delta 215), pack-reused 0
Receiving objects: 100% (2037/2037), 15.84 MiB | 9.03 MiB/s, done.
Resolving deltas: 100% (433/433), done.
```

Step 6: Run the Iroha docker container.

sudo docker run -it --name iroha \

-p 50051:50051 \

-v $(pwd)/iroha/example:/opt/iroha_data \

-v blockstore:/tmp/block_store \

--network=udit-iroha-network \

--entrypoint=/bin/bash \

hyperledger/iroha:latest

```
udit@udit-ubuntu:~$ sudo docker run -it --name iroha \
-p 50051:50051 \
-v $(pwd)/iroha/example:/opt/iroha_data \
-v blockstore:/tmp/block_store \
--network=udit-iroha-network \
--entrypoint=/bin/bash \
hyperledger/iroha:latest
Unable to find image 'hyperledger/iroha:latest' locally
latest: Pulling from hyperledger/iroha
e0b25ef51634: Pull complete
e1300e501129: Pull complete
a3124d67b376: Pull complete
657a8a434a5c: Pull complete
ffcf37f233dc: Pull complete
cd4d37888857: Pull complete
4f4fb700ef54: Pull complete
Digest: sha256:d3382063b858070786e13811e8cdcd8d7f0215b84a4444e0ea9c134ed662adc8
Status: Downloaded newer image for hyperledger/iroha:latest
root@951ab987e810:/opt/iroha_data#
```

Step 7: Run Iroha.

irohad --config config.docker \

--genesis_block genesis.block \

--keypair_name node0

```
root@951ab987e810:/opt/iroha_data# irohad --config config.docker \
> --genesis_block genesis.block \
> --keypair_name node0
[2023-06-23 19:32:50.446019430][I][Init]: Irohad version: 1.5.0
[2023-06-23 19:32:50.446059998][I][Init]: config initialized
[2023-06-23 19:32:50.446477847][I][Irohad]: created
[2023-06-23 19:32:50.446503056][I][Irohad]: [Init] => pending transactions stor
age
WARNING:  hash indexes are not WAL-logged and their use is discouraged
WARNING:  hash indexes are not WAL-logged and their use is discouraged
[2023-06-23 19:32:50.782608193][I][Irohad]: [Init] => storage
[2023-06-23 19:32:50.783609295][I][Irohad/Storage/Storage]: drop block storage
[2023-06-23 19:32:50.784025818][I][Irohad]: Recreating schema.
WARNING:  hash indexes are not WAL-logged and their use is discouraged
WARNING:  hash indexes are not WAL-logged and their use is discouraged
[2023-06-23 19:32:51.218452015][I][Irohad]: [Init] => storage
[2023-06-23 19:32:51.218482842][I][Irohad/Storage/Storage]: create mutable stor
age
[2023-06-23 19:32:51.240896513][I][Irohad/Storage/Storage/MutableStorageImpl]:
Applying block: height 1, hash 9debdb1a70db2cede2222427b849f6bf7ab20845da7c3db1
837c0df25ec1c61a
[2023-06-23 19:32:51.251196301][I][Init]: Genesis block inserted, number of tra
nsactions: 1
[2023-06-23 19:32:51.251845525][I][Irohad/Storage/Storage/PostgresSettingQuery]
: Kept value for MaxDescriptionSize: 64
[2023-06-23 19:32:51.251855733][I][Irohad]: [Init] => settings
[2023-06-23 19:32:51.251860007][I][Irohad]: [Init] => validators configs
[2023-06-23 19:32:51.251862104][I][Irohad]: [Init] => transaction batch parser
[2023-06-23 19:32:51.251879528][I][Irohad]: [Init] => validators
```

Step 8: Open new terminal and attach the docker container to our terminal.

sudo docker exec -it iroha /bin/bash

```
root@951ab987e810: /opt/iroha_data
root@951ab987e810: /opt/iroha_data    root@951ab987e810: /opt/iroha_data
udit@udit-ubuntu:~$ sudo docker exec -it iroha /bin/bash
[sudo] password for udit:
root@951ab987e810:/opt/iroha_data#
```

Step 9: Launch the iroha-cli tool and login as admin@test.

iroha-cli -account_name admin@test

```
udit@udit-ubuntu:~$ sudo docker exec -it iroha /bin/bash
[sudo] password for udit:
root@951ab987e810:/opt/iroha_data# iroha-cli -account_name admin@test
Welcome to Iroha-Cli.
Choose what to do:
1. New transaction (tx)
2. New query (qry)
3. New transaction status request (st)
> :
```

PRACTICAL 6

AIM: Write a program to demonstrate mining of Ether.

CODE:

```
from hashlib import sha256
import time

MAX_NONCE = 100000000000

def hashGenerator(text):
    return sha256(text.encode("ascii")).hexdigest()

def mine(block_number, transactions, previous_hash, prefix_zeros):
    prefix_str = '0'*prefix_zeros
    for nonce in range(MAX_NONCE):
        text = str(block_number) + transactions + previous_hash +
str(nonce)
        new_hash = hashGenerator(text)
        if new_hash.startswith(prefix_str):
            print(f"Successfully mined Ethers with nonce value :
{nonce}")
            return new_hash

    raise BaseException(f"Couldn't find correct hash after trying
{MAX_NONCE} times")

if __name__ == '__main__':
    transactions = '''
    Prateek->Hajra->77,
    Kunal->Soham->18
    '''
    difficulty = 4
    start = time.time()
    print("Ether mining started.")
```

```
    new_hash                                                    =
mine(5,transactions,'0000000xa036944e29568d0cff17edbe038f81208fecf9a
66be9a2b8321c6ec7', difficulty)
    total_time = str((time.time() - start))
    print(f"Ether mining finished.")
    print(f"Ether Mining took : {total_time} seconds")
    print(f"Calculated Hash = {new_hash}")
```

OUTPUT:

```
Ether mining started.
Successfully mined Ethers with nonce value : 35167
Ether mining finished.
Ether Mining took : 0.06797456741333008 seconds
Calculated Hash =
 0000fc864cda3e3c5ff5be050d0d74068e1367c510e162bb27af608d0d1edbea
```

PRACTICAL 7

AIM: Demonstrate the running of the blockchain node.

CODE:

```
from hashlib import sha256

def hashGenerator(text):
    return sha256(text.encode("ascii")).hexdigest()

class Block:
    def __init__(self, data, hash, prev_hash):
        self.data = data
        self.hash = hash
        self.prev_hash = prev_hash

class Blockchain:
    def __init__(self):
        hashLast = hashGenerator('gen_last')
        hashStart = hashGenerator('gen_hash')

        genesis = Block('Genesis Block', hashStart, hashLast)
        self.chain = [genesis]

    def add_block(self, new_block):
        new_block.prev_hash = self.chain[-1].hash
        new_block.hash    =    hashGenerator(str(new_block.data)    +
new_block.prev_hash)
        self.chain.append(new_block)

my_coin = Blockchain()
my_coin.add_block(Block({"amount": 72}, "", ""))
my_coin.add_block(Block({"amount": 24}, "", ""))

for block in my_coin.chain:
    print(block.__dict__)
```

OUTPUT:

```
{'data': 'Genesis Block', 'hash':
 '0a87388e67f16d830a9a3323dad0fdfa4c4044a6a6389cab1a0a37b651a5717b',
 'prev_hash':
 'bd6fecc16d509c74d23b04f00f936705e3eaa907b04b78872044607665018477'}
{'data': {'amount': 72}, 'hash':
 '0e97f2864c0431a30ae70be804f7e396136debab3836a0998c232efd0a9551e7',
 'prev_hash':
 '0a87388e67f16d830a9a3323dad0fdfa4c4044a6a6389cab1a0a37b651a5717b'}
{'data': {'amount': 24}, 'hash':
 'd12245a64c2b169c7877e35d6b7a8de1e514ab8e4df6f016235044438fa10f55',
 'prev_hash':
 '0e97f2864c0431a30ae70be804f7e396136debab3836a0998c232efd0a9551e7'}
```

PRACTICAL 8

AIM: Demonstrate the use of Bitcoin Core API.

CODE:

```
from bitcoinlib.wallets import Wallet
w = Wallet.create("UDITWallet")
key1 = w.get_key()
print(key1.address)
w.scan()
print(w.info())
```

OUTPUT:

```
18WPeXwhbMh4fTosr4bTmHzkq6h3Rbh3Nk
=== WALLET ===
 ID                              2
 Name                            UDITWallet
 Owner
 Scheme                          bip32
 Multisig                        False
 Witness type                    legacy
 Main network                    bitcoin
 Latest update                   2023-06-25 18:56:51.183095

= Wallet Master Key =
 ID                              17
 Private                         True
 Depth                           0
```

```
- NETWORK: bitcoin -
- - Keys
   22 m/44'/0'/0'/0/0              18WPeXwhbMh4fTosr4bTmHzkq6h3Rbh3Nk
                 address index 0                        0.00000000 ฿
   23 m/44'/0'/0'/0/1              1KmMGyiGK7QzUEv3i3yWGra8dcwuJy7vw7
                 address index 1                        0.00000000 ฿
   24 m/44'/0'/0'/0/2              1DPHYJiULScrqc8cfhV9D71m9R77rr1ei1
                 address index 2                        0.00000000 ฿
   25 m/44'/0'/0'/0/3              1CRBZCJUBKJEf27TMdAJgPPTbiiFaJf1sD
                 address index 3                        0.00000000 ฿
   26 m/44'/0'/0'/0/4              1J6U3rXAmHq9ys8G5i2VX2yJPJCCtDNehu
                 address index 4                        0.00000000 ฿
   28 m/44'/0'/0'/1/0              1FyAbZiJXkEbvpAfG6jWpQ9iFc63wd2zkK
                 address index 0                        0.00000000 ฿
   29 m/44'/0'/0'/1/1              187Dgp9RAUxRtNGuZurzeCxGqBPwfLkB1i
                 address index 1                        0.00000000 ฿
   30 m/44'/0'/0'/1/2              1A2fszNSRK9av23jPs5bMYREv2Vyf6xShq
                 address index 2                        0.00000000 ฿
   31 m/44'/0'/0'/1/3              1AameeN7QCpPzXc76tgwegATXCSHfhMutc
                 address index 3                        0.00000000 ฿
   32 m/44'/0'/0'/1/4              1DNwVEQE4heKmP5moERQdGn4gGzSkZp8cY
                 address index 4                        0.00000000 ฿
```

```
- - Transactions Account 0 (0)


= Balance Totals (includes unconfirmed) =



None
```

PRACTICAL 9

AIM: Create your own blockchain and demonstrate its use.

CODE:

```
from hashlib import sha256

def hashGenerator(text):
    return sha256(text.encode("ascii")).hexdigest()

class Block:
    def __init__(self,data,hash,prev_hash):
        self.data=data
        self.hash=hash
        self.prev_hash=prev_hash

class Blockchain:
    def __init__(self):
      hashLast=hashGenerator('gen_last')
      hashStart=hashGenerator('gen_hash')

      genesis=Block('gen-data',hashStart,hashLast)
      self.chain=[genesis]

    def add_block(self,data):
        prev_hash=self.chain[-1].hash
        hash=hashGenerator(data+prev_hash)
        block=Block(data,hash,prev_hash)
        self.chain.append(block)

bc=Blockchain()
bc.add_block('1')
bc.add_block('2')
bc.add_block('3')

for block in bc.chain:
```

```
    print(block.__dict__)
```

OUTPUT:

```
{'data': 'gen-data', 'hash':
 '0a87388e67f16d830a9a3323dad0fdfa4c4044a6a6389cab1a0a37b651a5717b',
 'prev_hash':
 'bd6fecc16d509c74d23b04f00f936705e3eaa907b04b7887204460766501847 7'}
{'data': '1', 'hash':
 'e3e6c97161f3deaf01599fda60ba85593b07f70328bf228473d1d408f7400241',
 'prev_hash':
 '0a87388e67f16d830a9a3323dad0fdfa4c4044a6a6389cab1a0a37b651a5717b'}
{'data': '2', 'hash':
 '47e8645e3c14bd4034a498aa88ea630bc0793375207bf90ca469792a5d9484e1',
 'prev_hash':
 'e3e6c97161f3deaf01599fda60ba85593b07f70328bf228473d1d408f7400241'}
{'data': '3', 'hash':
 '82084603decb1a14a8819dacaa86197659f1e150c4a50186e68043004b5a3c06',
 'prev_hash':
 '47e8645e3c14bd4034a498aa88ea630bc0793375207bf90ca469792a5d9484e1'}
```

University of Mumbai

PRACTICAL JOURNAL – ELECTIVE I

PSIT4P2d

Cyber Forensics

SUBMITTED BY

Vijay Hanumant Patil

SEAT NO 40434

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR QUALIFYING M.Sc. (I.T.) PART-II (SEMESTER – IV) EXAMINATION

2022-2023

DEPARTMENT OF INFORMATION TECHNOLOGY
3RD FLOOR, DR. SHANKAR DAYALSHARMA BHAVAN,
VIDYANAGRI, SANTACRUZ(EAST),
MUMBAI – 400098.

University of Mumbai

Department of Information Technology

Certificate

This is to certify that **Mr. Vijay Hanumant Patil, Seat No. 40434** studying in **Master of Science** in **Information Technology Part II Semester IV** has satisfactorily completed the Practical of **PSIT4P2d Cyber Forensics** as prescribed by University of Mumbai, during the academic year **2022-23**.

Signature	**Signature**
Subject-In-Charge	**Head of the Department**

Signature

External Examiner

College Seal: _____ **Date:** _____

INDEX

Practical No.	Date	Name of Practical	Page No.	Signature
1	3/3/2023	File System Analysis using The SleuthKit (Autospy, fsstat, istat, fls and img_stat)	03	
2	13/3/2023	Explore Windows forensic tools (OSForensics)	12	
3	21/3/2023	Using Forensic Toolkit(FTK) &Writing report usingFTK (AccessData FTK)	25	
4	4/4/2023	a. Using File Recovery Tools [FTK Imager] Creating Image b. Recover Deleted files using Recuva, PC Inspector File Recovery, Recover My Files	32	
5	10/4/2023	a. Using Web attack detectiontools [Wireshark] b. Using Log & Traffic Capturing & Analysis Tools [Wireshark]	49	
6	25/4/2023	Dump Memory contents using PMdump	69	
7	11/5/2023	Using Data Acquisition Tools [ProDiscover Pro]	71	
8	23/5/2023	a. Using Steganography Tools [S-Tools] b. Using Whitespace Stegnography tool SNOW	75	
9	16/6/2023	Performing Password Cracking [Cain & Abel]	83	
10	28/6/2023	a. Scan Registry using RegScanner	87	

PRACTICAL NO .01

File System Analysis using The SleuthKit (Autospy, fsstat, istat, fls and img_stat)

Aim: Exploring Autopsy.

To install Autopsy, navigate to C:\CHFI-Tools\CHFIv10 Module 03UnderstandingHardDisksandFileSystems\FileSystemAnalysisTools\Autopsy, double-click autopsy-4.14.0-64bit.msi installer and follow the wizard driven installation steps to complete the installation process.

Autopsy Welcome window will appear along with Autopsy main window in the background. In the Welcome window, click New Case.

A New Case Information window opens asking you to input the Case Name and the Base Directory. The base directory is the location where the case data will get stored. The case name may be entered according to your identification purpose. In this lab, we are assigning the case name as Linux_Analysis.7.

Before specifying the base directory, we will be creating a folder on the Desktop with the name Image File Analysis and setting the path of the Base directory to this folder. Upon setting the base directory, click Next

The New Case Information window now shows the Optional Information section where you can specify details such as name of the examiner and case number. For this lab, let us enter the name of the examiner as Jonathan and the case number as 1001-125. You may also fill out the other optional fields. Click Finish after entering the details for optional fields.

The Add Data Source window now appears displaying the section Select Type of Data Source to Add. Here,you need to select the type of data source to be provided as an input. In this lab, we will be analyzing a disk image; therefore, select the option Disk Image or VM File and click Next.

A window (named Open) will appear where you need to specify the forensic image. Navigate to C:\CHFI-Tools\Evidence Files\Forensic Images, select Linux_Evidence_001.img and click Open.

The Add Data Source window now displays the Configure Ingest Modules section, which contains lists of options that are checked. Select the options according to your requirement and click Next.

The application now displays the result in the Autopsy main window. Expand the Data Sources node in the left pane and click on the image file i.e.,Linux_Evidence_001.img. This will show the contents of the image file, as shown in the following screenshot:

You may examine all the required files stored in the image as a part of filesystem analysis. In this lab, we are going to view the passwd file that is stored in \etc location. Therefore, select the etc folder from the left pane.

Upon selecting the folder, all the files and folders present in etc are displayed in the right pane of the window.

Scroll down the window, select the passwd file and click the Text tab.

Autopsy displays all the text (user account information) present in the passwd file, under the Strings tab,

as shown in the following screenshot:

The SeatPlan.xls file appears in the right pane of the Autopsy window. Click the file.

Click on File Metadata and scroll down the section to find the MD5 value for the SeatPlan.xls file.

To study fsstat

To view partition tables associated with Windows_Evidence_002.dd, type mmls "C:\CHFI-Tools\EvidenceFiles\ForensicImages\Windows_Evidence_002.dd"and press Enter. This displays the partition layout of a volume system (partition tables) associated with the image file, as shown in the following screenshot:

Similarly, to view the type of file system and the OS related to the image, type fsstat "C:\CHFI-Tools\Evidence Files\Forensic Images\Windows_Evidence_001.dd"and then press Enter.

From the above screenshot, it can be observed that the file system is NTFS and the source OS is Windows XP

To Study img_stat

Use the img_stat command to view the details of the selected image. Type img_stat"C:\CHFI-Tools\Evidence Files\Forensic Images\Windows_Evidence_001.dd"and press Enter to view the details.

To Study istat

Use the istat tool in The Sleuth Kit to view the details of metadata structure. To display an overview of the MFT file, type istat -f ntfs "C:\CHFI-Tools\Evidence Files\Forensic Images\Windows_Evidence_001.dd" 0 and press Enter to view the details.

To display the MFTMirr File Overview, type istat -f ntfs "C:\CHFI-Tools\Evidence Files\Forensic Images\Windows_Evidence_001.dd" 1 and press Enter.

```
Administrator: C:\Windows\system32\cmd.exe

C:\CHFI-Tools\CHFIv10 Module 03 Understanding Hard Disks and File Systems\File System Analysis Tools\The Sleuth Kit (TSK)\bin>istat -f ntfs "C:\CHFI-Tools\Evidence Files\Forensic Images\Windows_Evidence_001.dd" 1
MFT Entry Header Values:
 Entry: 1        Sequence: 1
  $LogFile Sequence Number: 2118363
Allocated File
 Links: 1

$STANDARD_INFORMATION Attribute Values:
Flags: Hidden, System
 Owner ID: 0
Security ID: 256  (S-1-5-18)
Created:        2019-12-19 01:55:24.565826900 (Pacific Standard Time)
File Modified:  2019-12-19 01:55:24.565826900 (Pacific Standard Time)
MFT Modified:   2019-12-19 01:55:24.565826900 (Pacific Standard Time)
Accessed:       2019-12-19 01:55:24.565826900 (Pacific Standard Time)

$FILE_NAME Attribute Values:
Flags: Hidden, System
Name: $MFTMirr
Parent MFT Entry: 5     Sequence: 5
Allocated Size: 4096    Actual Size: 4096
Created:        2019-12-19 01:55:24.565826900 (Pacific Standard Time)
File Modified:  2019-12-19 01:55:24.565826900 (Pacific Standard Time)
MFT Modified:   2019-12-19 01:55:24.565826900 (Pacific Standard Time)
Accessed:       2019-12-19 01:55:24.565826900 (Pacific Standard Time)

Attributes:
```

The layout of the MFT is determined by processing entry 0 in the MFT.MFT entry 1 is for the MFTMirr file, which has a non-residentattribute that contains a backup copy of the first MFT entry. The Boot file system metadata file is in MFT entry 7 and contains the boot sector of the file system. The MFT entry for the AttrDef filesystem metadata file is 4. It defines the names and type identifiers for each type of attribute. NTFS keeps track of the damaged clusters by allocating them to a$DATA attribute of the BadClus file system metadata file. The MFT entry is 8.

To Study fls

Use the fls command-line tool of TSK to list the files and directory names. Type fls -f ntfs "C:\CHFI-Tools\EvidenceFiles\ForensicImages\Windows_Evidence_001.dd"and then press Enter.

```
Administrator: C:\Windows\system32\cmd.exe

C:\CHFI-Tools\CHFIv10 Module 03 Understanding Hard Disks and File Systems\File System Analysis Tools\The Sleuth Kit (TSK)\bin>fls -f ntfs "C:\CHFI-Tools\Evidence Files\Forensic Images\Windows_Evidence_001.dd"
r/r 4-128-1:    $AttrDef
r/r 8-128-2:    $BadClus
r/r 8-128-1:    $BadClus:$Bad
r/r 6-128-1:    $Bitmap
r/r 7-128-1:    $Boot
d/d 11-144-4:   $Extend
r/r 2-128-1:    $LogFile
r/r 0-128-6:    $MFT
r/r 1-128-1:    $MFTMirr
d/d 1228-144-1: $RECYCLE.BIN
r/r 9-128-8:    $Secure:$SDS
r/r 9-144-11:   $Secure:$SDH
r/r 9-144-14:   $Secure:$SII
r/r 10-128-1:   $UpCase
r/r 10-128-4:   $UpCase:$Info
r/r 3-128-3:    $Volume
d/d 39-144-5:   Audio Files
d/d 48-144-5:   images
d/d 85-144-5:   Other Files
d/d 1169-144-1: Outlook Files
d/d 1180-144-5: Songs
d/d 36-144-1:   System Volume Information
```

PRACTICAL NO.02

a. Explore Windows forensic tools (OSForensics)

Navigate to C:\CHFI-Tools\CHFIv10Module 06 Windows Forensics\Windows Forensics Tools\OS Forensics, double-clickosf.exe to launch the steup. In the final step of installation, check the Launch OSForensics option and click Finish. OSForensics GUI appears, along with PassMark OSForensics pop-up. In the pop-up, click Continue Using Trial Version. Our first task is to create a case using this tool. Click the Create Case icon in the tool's main window to create a new case.

A New Case window appears; fill out the required fields in the window. Ensure that you select the radio button for the Live Acquisition of Current Machine option in the Acquisition Type category. You may choose to save the new case folder either in Default or Custom locations. Click OK

The OSForensics tool can help investigators in searching and locating files on a system. To start searching for files, click File Name Search in the left pane of the window.

In the Start Folder field, specify the path to search for image files by clicking the ellipsis button and choosing the location (here, we are specifying the location C:\CHFI-Tools\EvidenceFiles\Image Files to search for images in it). Then, click the Search button.

Alternatively, if you wish to search only for a single image on the entire system, select Images from the Pre-sets drop-down menu, enter the name of the desired image file in the Search String field, and then click Search, as shown in the screenshot below. Here, we are searching for the JPEG image file named Kitty.

The Create Index section appears in the right pane. In this section, select the Use Pre-defined File Types option for creating the index and check the required options listed under it for selecting the file types that you wish to index (here, we have selected the Images option). Then, click Next.

OSForensics - Case 75
Workflow
Mismatch File Search
Program Artifacts
File System Browser
File Viewer
Raw Disk Viewer
Registry Viewer
Web Browser
Create Index
Search Index
Create Signature
Compare Signature
SQLite DB Browser
Hash Sets
Create Index
Help
Step 1 of 5
What types of files would you like to index?
Use Pre-defined File Types
Check All
Uncheck All
Emails
Attachments
Plain text files
All other supported file types
Office + PDF documents
Web files + XML
Unknown files
ZIP and compressed archives
Video, audio and other media
System hibernation and paging files
Images
Executables and binary files
Use OCR (Win10 only)
Use previously saved configuration:
Configuration
File types
Next

OSForensics - Case 75
Workflow
Mismatch File Search
Program Artifacts
File System Browser
File Viewer
Raw Disk Viewer
Registry Viewer
Web Browser
Create Index
Search Index
Create Index
Help
Step 2 of 5
Which drive(s) or folder(s) would you like to index?
Start Folder
Type
Add...
Remove
Advanced settings (optional)
File extensions
Edit
Precognitive search
Edit
Add Start Location
Whole Drive
C:\
Drive indexing options:
Index files only
Specific Folder
C:\CHFI-Tools\Evidence Files\Image Files
OK
Cancel

OSForensics - Case 75
Workflow
Mismatch File Search
Program Artifacts
File System Browser
File Viewer
Raw Disk Viewer
Registry Viewer
Web Browser
Create Index
Search Index
Create Signature
Compare Signature
SQLite DB Browser
Hash Sets
Create Index
Help
Step 2 of 5
Which drive(s) or folder(s) would you like to index?
Start Folder
Type
C:\CHFI-Tools\Evidence Files\Image Files
Folder
Add...
Remove
Advanced settings (optional)
File extensions
Edit
Precognitive search
Edit
Skip files/folders
Edit
Binary string extraction
Edit
Languages & Stemming
Edit
Back
Next

OSForensics - Case 75
Workflow
Mismatch File Search
Program Artifacts
File System Browser
File Viewer
Raw Disk Viewer
Registry Viewer
Web Browser
Create Index
Search Index
Create Signature
Compare Signature
SQLite DB Browser
Create Index
Help
Step 3 of 5
Memory optimization / Indexing limits
Estimate the number of files (and size) being indexed. This will help optimize memory usage and index more efficiently.
Small
Medium
Large
Extreme
Don't know (Pre-scan required)
Custom
Edit
Pre-scan will determine these settings for you.
This may take a while (from 5 minutes to 30 minutes) depending on the size of your data.
*Max file size does not apply to some file formats
Select number of threads:
4
Use RAM drive for temporary files to speed up indexing
Back
Next

We will now search the indexed files. To search for the indexed files, select Search Index from the left pane of the tool window. The Search Index section now appears in the tool window. The index field at the top displays the name of the case we created (i.e., Case 75). Click Search. The tool will load the image files that you have indexed under the Images tab.

Click User Activity in the left pane to scan for evidence such as browsed websites, USB drives, recent downloads and wireless networks.

If you wish to save any of the above user-activity information on your system, such as Browser History, then scroll down the items list in the left pane below the Scan Drive option and expand the Browser History node. Select the browser for which you wish to retrieve the history (Here, we are selecting Chrome). The browser history pertaining to this browser will be displayed in the right pane of the tool window.

An Export List to. Window will appear. Navigate to C:\Users\Administrator\Documents\PassMark\OSForensics\Cases\Case 75, name the file (here, we are naming it as Browsing History), and click Save to export the browsing history.

To recover deleted files from the file system, click Deleted File Search in the left pane, select a disk on which you wish to perform the deleted file search from the Disk drop-down menu, and click the search button.

To locate files having contents that do not match the file extensions, click Mismatch File Search in the left pane of the tool window. The tool will display the Mismatch File Search section in the right pane, as shown in the screenshot:

To view the processes running on the system, click Memory Viewer

To retrieve detailed information about the core components of the system, click System Information. The tool will display the System Information section.

In this manner, you can perform investigation on a system or the folders and partitions within the system to gather data of interest to the forensic investigation.

PRACTICAL NO .03

Using Forensic Toolkit (FTK) &Writing report usingFTK (AccessData FTK)

Aim: Using Windows Forensics Tools.

Step 1: Start Forensic Toolkit.

Step 2: Here, prompted with a warning dialog box, click on OK to continue.

Step 3: click on OK button.

Step 4: Now select Start New Case option and click on ok.

Step 5: Enter the detail for a New case.

Step 6: Fill the information in Forensic Examiner Information dialog box.

Step 7: leave the default settings and click on next.

Step 8: Now again leave the default settings and click on next.

Step 9: In the Refine Case-Default, click the Include All items button and then click Next.

Step 10: In Refine Index-Default, accept the default settings and click Next.

Step 11: Now here Click on add Evidence button.

Step 12: Enter Evidence Information and click on OK button.

Step 13: Now click on Next.

Step 14: Click on Finish to initiate the analysis.

Step 15: Now Processing Will Start.....

Step 16: when FTK finishes the processing part, the FTK window opens to the Overview tab.

Step 17: Select Deleted Files option to explore the evidence items.

Step 18: Select Encrypted Files to view.

From the menu, select **Report**, and then **Generate Report** or click the button on the toolbar.

The Case Information dialog appears, enter the Case information and The Bookmarks-A & B dialog appears select what you want to include in report click next.

List File Properties dialog appear, include the list you want in Report and click next and Finish.

PRACTICAL NO:04

Using File Recovery Tools [FTK Imager] Creating Image

Aim: Understanding & working with the process of the process of taking a drive image using AccessData's FTK Imager tool.

Step 1) Run **FTK Imager.exe** to start the tool.

Step 2) To create a forensic image:

Click File > Create Disk Image

Step 3) In the Select Source dialog box, select the source you want to make an image of. Click Next.

If you select Logical Drive and need to select a floppy or CD as a source, you can check the Automate multiple removable media box to create groups of images. Imager will automatically increment the case numbers with each image, and if something interrupts the process, you may assign case number manually.

Step 4) Select the drive or browse to the source of the image you want, and then click Finish.

Step 5) In the Create Image dialog, click Add**....** to add the image destination.

- Compare the stored hashes of your image content by checking the Verify images after they are created box. If a file doesn't have a hash, this option will generate one.

List the entire contents of your images with path, creation dates, whether files were deleted, and other metadata. The list is saved in tab-separated value (.TSV) format.

Step 6) Select the type of image you want to create.

The type you choose will usually depend on what tools you plan to use on the image. The dd format will work with more open-source tools, but you might want SMART or E01 if you will primarily be working with ASR Expert Witness or EnCase, respectively.

Note: If you are creating an image of a CD or DVD, this step is skipped because all CD/DVD images are created in the IsoBuster CUE format. Hashes are not generated for CD and DVD images so they will not be verified, as well.

Important: The raw image type is not compressed. If you select the Raw (dd) type, be sure to have adequate available drive space for the resulting image.

Step 7) If you are creating an AFF image type, choose AFF. Click Next.
The Image Destination Folder dialog box you see will be different than that seen when selecting any other image type

Step 8) If your version of FTK requests evidence information, you can provide it. Specify Evidence Item Information. All Evidence Item Information is optional, but it is helpful to have the information easily accessible in case it is called into question at any time after creation

Complete the fields in the Evidence Item Information dialog. Click Next.

Step 9) Select the Image Destination folder and file name. You can also set the maximum fragment size of image split files. Click Finish to complete the wizard.

In the Image Destination Folder field, do one of the following:

- Type the location path where you want to save the image file.
- Click Browse to find and select the desired location.

In the Image Filename field, specify a name for the image file but do not specify a file extension.

Step 10) Specify the Image fragment Size:

- Default Image Fragment Size = 1500 MB
- To save images segments that can be burned to a CD, specify 650 MB.
- To save image segments that can be burned to a DVD, specify 4000 MB.
- The .S01 format is limited by design to sizes between 1 MB and 2047 MB (2 GB). Compressed block pointers are 31-bit numbers (the high bit is a compressed flag), which limits the size of any one segment to two gigabytes.

Step 10 a) Select the compression level to use.

- 0=No Compression
- 1=Fastest, Least Compression (faster, and also slightly smaller than a 0-compression file)
- 9=Slowest, Most Compression (smallest file, slowest to create).

Numbers between 1 and 9 produce an image with varying levels of compression to speed ratio.

Step 11) To encrypt the image, choose the correct encryption box as explained below:
a. To encrypt the new image with AD Encryption, mark the Use AD Encryption box.
b. To encrypt the new image with AFF Encryption, mark the Use AFF Encryption box.

Step 12) Click Finish.

Step 13) When AFF Encryption is selected, type the password, and retype the password to confirm. Click Show Password to see that you have typed it correctly the first time.

Step 14) When encryption selections are made, click **OK** to save selections and return to the **Create Image dialog.** Click **Start** to begin the imaging process.

After the images are successfully created, the Drive/Image Verify Results box shows detailed image information, including MD5 and SHA1 check sums, and bad sectors.

Now is a good time to refill that coffee cup! Once the acquisition is complete, you can view an image summary and the drive will appear in the evidence list in the left-hand side of the main FTK Imager window. You can right-click on the drive name to Verify the Image:

A progress dialog appears that shows the following:

- The source that is being imaged
- The location where the image is being saved
- The status of the imaging process
- A graphical progress bar
- The amount of data in MB that has been copied and the total amount to be copied
- Elapsed time since the imaging process began
- Estimated time remaining until the process is complete
- Image Summary button. Click it to open the Image Summary window as shown below:

b. Recover Deleted files using Recuva, PC Inspector File Recovery, Recover My Files

Recuva

Download and install **Recuva**

Select File Type you need to recover

Select File location, which you want to recover files

Click next (If want enable Deep Scan)

Switch to advanced mode and select all recover file

Select the location were to recover the file

All deleted Files are recovered at specified location

PC Inspector File Recovery

Download and install **PC Inspector File Recovery**

Choose what to recover delete files, lost data or lost drive

Recover data is saved to specified location

Recover My Files

Download and install Recover My Files

Select the option you wish to of Recover Files and Recover a Drive

Select the drive to search and recover files & select one of the File Recovery options

What file headers would like to search from recover files

Scan and save the recovered file to desired location

PRACTICAL NO :05

a. Using Web attack detection tools [Wireshark]

1. Download and install wireshark

2. After completing the installation, navigate to C:\CHFI-Tools\EvidenceFiles\WiresharkSample Capture Files\Files for Network Forensics and double-click HTTP Traffic.pcapng.

3. The above operation will launch the Wireshark GUI window and display the packets captured in HTTP Traffic.pcapng, as shown in the following screenshot

4. We will now apply the http filter so that the application displays results related to traffic generated through http. Type http in the Filter field and press Enter to filter the http traffic, as indicated in the screenshot below:

5. After examining the entries displayed by the application in the screenshot above, it is evident that the http traffic is associated with a WordPress website, and it is being transmitted in a plain text format. Generally, user credentials are stored in the POST requests. Therefore, examining packet(s) containing POST request can help an investigator find the user credentials.
6. We will now filter the traffic to obtain results specific only to POST request(s). Now, type http.request.method== POST in the Filter field and press Enter. Wireshark filters the traffic containing POST request(s) and displays them, as shown in the screenshot below. The user credentials stored in this request can be found under the Packet Details pane in the middle of the application window, under the HTML Form URL Encoded node.

7. Now, navigate to C:\CHFI-Tools\Evidence Files\Wireshark Sample Capture Files\Files for Network Forensics and double-click DNS RemoteShell.pcapng.

8. The network traffic entries captured in the DNS Remote Shell.pcapng file will now be displayed in the Wireshark GUI window, as shown in the following screenshot

9. Since DNS uses port 53 for communicating with clients, we will be filtering the traffic flowing to and from port 53. To filter traffic flowing on port 53, type the command tcp.port == 53 in the Filter field and press Enter. Wireshark filters the traffic flowing on port 53 and displays it, as shown in the following screenshot

When we examine the first packet from the listed packets, i.e., Packet 14,we can see that Port 1396 on Source IP address (192.168.1.3) is trying to establish a remote connection with Port 53 on the Destination IP address(192.168.1.2) through a [SYN] request. As Port 53 is seen on 192.168.1.2,this IP address represents the DNS server here. When we examine Packet17, we see that the IP address 192.168.1.2, which is the DNS server, is responding with a [SYN, ACK] acknowledgment to IP address 192.168.1.3.Upon examining Packet 18, we see that the IP address 192.168.1.3 is sending an [ACK] acknowledgment to 192.168.1.2, thus establishing a remote connection with the target/victim (i.e., 192.168.1.2). From these observations, we can infer that 192.168.1.3 is the attacker's IP address, which has succeeded in establishing a connection from its Port 1396 with Port 53 on the DNS server, as indicated through Packets 14, 17, and 18 in the screenshot below:

10. Right-click on any one of the packets from 14 to 38 (here, we have right-clicked on packet 14), select Follow from the context menu, and then click TCP Stream from the resultant drop-down list, as shown in the screenshot below:

11. Now, we will look for FTP brute-force attempts in the network. Close the current packet capture file in Wireshark.

12. Apply the ftp.response.code == 530 filter to monitor all unsuccessful login attempts over FTP. Upon applying the filter, the application will fetch results pertaining to unsuccessful login attempts, as shown in the screenshot below:

13. The screenshot above shows multiple unsuccessful login attempts madefrom the source IP 10.0.0.16 to the target IP 10.0.0.8, which stronglyindicates a brute-force attack.27. Apply the ftp.response.code == 230 filter to see successful logins on the FTP server. The application will fetch results that show successful logins from the source IP 10.0.0.16 to the target IP 10.0.0.8, as shown in the screenshot below. This indicates the attacker has successfully gained the victim's login credentials

14. Now, we will look for ARP Poisoning attempt in the network.

15. when ARP Poisoning has been attempted, Wireshark will detect duplicate IP addresses on the ARP protocol with the warning message Duplicate IP address detected for <IP address>. Therefore, we need to check if the use of a duplicate IP address has been detected.45. To locate a duplicate IP address in the traffic, apply the arp. duplicate-address-detected filter. Wireshark detects duplicate IP address on the ARP protocol with the warning message Duplicate IP address detected for 10.0.0.1, as shown in the screenshot below. The duplicated IP address can be seen both in the Packet Details pane in the middle of the application window and in its upper pane.

In this manner, you can analyze packet capture files that have recorded the traffic on a network as part of a forensic investigation.

b. Using Log & Traffic Capturing & Analysis Tools [Wireshark]

Using Log Capturing and Analysis tools

Aim: Exploring Wireshark

Wireshark is a network packet analyzer that intercepts, captures and logs information about packets passing through a network interface. This is useful for analyzing network problems, detecting network intrusions, network misuse, and other security problems, monitor usage and gather statistics, and many other applications.

Filtering Packets

If you're trying to inspect something specific, such as the traffic a program sends when phoning home, it helps to close down all other applications using the network so you can narrow down the traffic. Still, you'll likely have a large amount of packets to sift through. That's where Wireshark's filters come in.

The most basic way to apply a filter is by typing it into the filter box at the top of the window and clicking Apply (or pressing Enter). For example, type "dns" and you'll see only DNS packets. When you start typing, Wireshark will help you autocomplete your filter.

You can also click the Analyze menu and select Display Filters to create a new filter.

Another interesting thing you can do is right-click a packet and select Follow TCP

You'll see the full conversation between the client and the server.

Close the window and you'll find a filter has been applied automatically — Wireshark is showing you the packets that make up the conversation.

Inspecting Packets

Click a packet to select it and you can dig down to view its details.

You can also create filters from here — just right-click one of the details and use the Apply as Filter submenu to create a filter based on it.

Using Traffic Capturing and Analysis tools

Aim: Exploring Wireshark

Step 2: On menu bar select Capture. Select interfaces.

Step 3: Select Once you click on start, then Wireshark starts to capture the packets on that interface.

Step 4: Filter packets with HTTP protocol.

Step 5: A file with only text: http://www.mu.ac.in/ppe/LIBRARY%20SCIENCE-(SEM.I)-SH-2014.pdf

Step 6: Applying different filters using expressions.

1) Filtering HTTP POST request

2) Filtering 404 not found error

3) Filtering using HTTP Content Length

PRACTICAL NO:06

Dump Memory contents using PMdump

1. Here we will use program name mempass, and we will see how by using combination of pmdump and strings, we are able to dump the password in memory.

Let start the mempass

2. Now open new terminal and run PMdump, pmdump –list will list the running program currently.

```
C:\Users\IEUser\Desktop>pmdump

pmdump 1.2 - (c) 2002, Arne Vidstrom (arne.vidstrom@ntsecurity.nu)
             - http://ntsecurity.nu/toolbox/pmdump/

Usage: pmdump (pid) (filename)

         - dumps the process memory contents to a file

       pmdump -list

         - lists all running processes and their PID's
```

```
C:\Users\IEUser\Desktop>pmdump -list

pmdump 1.2 - (c) 2002, Arne Vidstrom (arne.vidstrom@ntsecurity.nu)
             - http://ntsecurity.nu/toolbox/pmdump/

     0 - System idle process
     4 - System
   252 - smss.exe
   328 - csrss.exe
   364 - csrss.exe
   372 - wininit.exe
   400 - winlogon.exe
   460 - services.exe
   468 - lsass.exe
   476 - lsm.exe
   568 - svchost.exe
   624 - VBoxService.exe
   676 - svchost.exe
   780 - svchost.exe
   820 - svchost.exe
   848 - svchost.exe
   872 - svchost.exe
   996 - svchost.exe
  1148 - svchost.exe
  1256 - spoolsv.exe
  1292 - svchost.exe
  1384 - svchost.exe
  1424 - svchost.exe
  1584 - cygrunsrv.exe
  1660 - wlms.exe
  1696 - conhost.exe
  1720 - sshd.exe
  1932 - sppsvc.exe
   296 - svchost.exe
  1520 - taskhost.exe
  2280 - GoogleCrashHandler.exe
  2596 - svchost.exe
  2708 - SearchIndexer.exe
  2928 - dwm.exe
  2956 - explorer.exe
  3096 - VBoxTray.exe
  3560 - wuauclt.exe
  3104 - cmd.exe
  2700 - conhost.exe
  3808 - cmd.exe
  1992 - conhost.exe
```

```
 2076 - taskeng.exe
 3456 - mempass.exe
  976 - cmd.exe
 2120 - pmdump.exe

C:\Users\IEUser\Desktop>mempass 3456 paswd.txt
```

3. Here we can see the pid of mempass i.e., 3456. We will dump the entire memory of mempass program in paswd.txt file

```
C:\Users\IEUser\Desktop>mempass 3456 paswd.txt
Please enter your password:hello_
```

4. File name paswd is created having direct memory of mempass program without tampering the running mempass program

In this way, we can retrieve passwords from applications which store them in memory without any form of encryption.

PRACTICAL NO:07

Using Data Acquisition Tools [ProDiscover Pro]

Step 1) Start ProDiscover. ProDiscover presents the launch dialog.

Step 2) Enter a project number, project name, and description of the project in the new project tab
option, and then click the Open button.

ProDiscover will then create a project and generate a template report in the work area.

Step 3) Open Image:

1] Select save project option from the file menu, or button bar and Open Image option.

Step 4) ProDiscover displays the contents of the selected file at the bottom of the main window. **Right click** on a file.

Step 5) In ProDiscover a pop-up dialog with the choice to View or Recover the selected file. Select **View**

Step 6)For further study, we can also copy the file at desired location

PRACTICAL NO:08

a. Using Steganography Tools [S-Tools]

Following steps Show how to use freeware S-Tools utility to hide and reveal files inside pictures

Step 1) Select the S-Tools.exe file and open the steganography software tool.

Step 2) With both the working directory and the S-Tools program open minimize both windows and place side-by-side.

The S-Tools program is a drag and drop software. The files used to create the steganography file can be dragged from the directory into the S-Tools program.

Step 3) Select the file from the directory and drag it over the S-Tools main window and release the
file.

A dialogue box appears indicating that the file type is unknown. Supported file types for audio and image files are shown below:

- Audio - *.wav
- Image - *.bmp and *.gif

If your image is in .jpg format, convert it to .bmp format.

Step 4) Select a file to hide within the base file. If it's not there, create a txt file and save the file. Here we have created file name Hidden file
Step 5) The *.txt text file is selected and dragged on top of the base image. Release the file while the cursor is still on top of the base file.

Step 6) A dialogue box will appear asking the user to enter and verify a passphrase. Additionally, the user will have to select an encryption algorithm.

Step 7) Select the 'OK' button after entering a valid passphrase.

Step 8) The S-Tools main window will appear and a new file will be visible. The name of the file will be called hidden data by default.

Step 9) Place the cursor on top of the hidden data image and select the right mouse button. The user will
have four options available to them:

- Save
- Save As
- Properties
- Reveal

Select the 'Save As' button.

Step 10) A 'Save As' dialogue box will appear. Enter a valid file name, select the working directory and select the 'Save' button.

Step 11) Selecting the 'Reveal' button will display a passphrase dialogue box. A passphrase must be entered twice in the dialogue box and the correct encryption algorithm must be selected.

Step 12) Enter a passphrase twice, select the encryption algorithm, and select the 'OK' button.

Step 13) A 'Revealed Archive' dialogue box will display which contains the file name and size of the hidden file.

Step 14) Select the 'Save As' button.

Step 15) The Revealed data is saved in revealfile.txt

b. Using Whitespace Stegnography tool SNOW

Step 1) Create a text file with some data in the same directory where Snow Tool is installed.

Step 2) Go to Command Prompt
Change the directory to run Snow tool

Step 3) Type the command
Snow –C –m "text to be hide" –p "password" <Sourcefile> <Destinationfile>

The source file is a Hello.txt file as shown above. Destination file will be the exact copy of source file containing hidden information.

Go to the directory; you will a new file **HelloWorld.txt**. Open the File

Step 4) New File has the same text as an original file without any hidden information. This file can be sent to the target.

Recovering Hidden Information

On destination, Receiver can reveal information by using the command

Snow –C –p "password123" HelloWorld.txt

As shown in the above figure, File decrypted, showing hidden information encrypted in the previous section.

PRACTICAL NO: 09

a. Performing Password Cracking [Cain & Abel]

Step 1: Install and open cain and abel.

Step 2: Select sniffer on the top.

Step 3: Next to folder icon click on icon name start/stop sniffer. Select device and click on ok.

Step 4: Click on "+" icon on the top. Click on ok.

Step 5: Shows the Connected host.

Step 6: Select Arp at bottom.

Step 7: Click on "+" icon at the top.

Step 8: Click on start/stop ARP icon on top.

Step 9: Poisoning the source.

Step 10: Go to any website on source ip address.

Step 11: Go to password option in the cain & abel and see the visited site password

PRACTICAL NO:10

a. Scan Registry using RegScanner

Step 1) Download and Install RegScanner

Step 2) Quickly scan your Registry and Search for the specific values with RegScanner

You can select to scan the base keys and the string types. Enter the search string and start the scan

Step 3) Check the Scan Result

Step 4) You can save, copy or export the selected items, Export Reg values as REG or HTML files, and more

University of Mumbai

PRACTICAL JOURNAL – Elective II

PSIT4P3d
Security Operations Center

SUBMITTED BY

Vijay Hanumant Patil

SEAT NO 40434

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR QUALIFYING M.Sc. (I.T.) PART-II (SEMESTER – IV) EXAMINATION

2022-2023

Department of Information Technology

3RD FLOOR, DR. SHANKAR DAYAL SHARMA BHAVAN, IDOL BUILDING, VIDYANAGRI, SANTACRUZ (E), MUMBAI – 400098.

University of Mumbai

Department of Information Technology

Certificate

This is to certify that **Mr. Vijay Hanumant Patil, Seat No. 40434** studying in **Master of Science in Information Technology Part II Semester IV** has satisfactorily completed the Practical of **PSIT4P3d Security Operations Center** as prescribed by University of Mumbai, during the academic year 2022 - 23.

Signature
Subject-In-Charge

Signature
Head of the Department

Signature External
Examiner

College Seal:_____

Date: _____

INDEX

Sr. No	Topic	Page No.	Date	Sign
1	Encrypting and Decrypting Data Using OpenSSL	**02**	2/3/2023	
2	Demonstrate the use of Snort and Firewall Rules	**04**	14/3/2023	
3	Demonstrate Extract an Executable from a PCAP	**09**	23/3/2023	
4	Demonstrate Analysis of DNS Traffic	**12**	6/4/2023	
5	Create your own syslog Server	**18**	12/4/2023	
6	Configure your Linux system to send syslog messages to a syslog server and Read them	**20**	28/4/2023	
7	Install and Run Splunk on Linux	**22**	9/5/2023	
8	Install and Configure ELK on Linux	**25**	19/5/2023	
9	Install and Configure GrayLog on Linux	**28**	31/5/2023	
10	Demonstrate Conversion of Data into a Universal Format	**33**	19/6/2023	

Practical No.1

Aim: Encrypting and Decrypting Data Using OpenSSL.

Encrypting Messages with OpenSSL
OpenSSL can be used as a standalone tool for encryption. While many encryption algorithms can be used, this lab focuses on AES. To use AES to encrypt a text file directly from the command line using OpenSSL, follow the steps below:

Step 1: Encrypting a Text File

a. Log into CyberOPS Workstation VM.

b. Open a terminal window.

c. Because the text file to be encrypted is in the /home/analyst/lab.support.files/ directory, change to that directory:

[analyst@secOps ~]$ cd ./lab.support.files/

[analyst@secOps lab.support.files]$

d. Type the command below to list the contents of the encrypted letter_to_grandma.txt text file on the screen:

[analyst@secOps lab.support.files]$ cat letter_to_grandma.txt

```
[analyst@secOps lab.support.files]$ cat letter_to_grandma.txt
Hi Grandma,
I am writing this letter to thank you for the chocolate chip cookies you sent me. I
got them this morning and I have already eaten half of the box! They are absolutely
delicious!

I wish you all the best. Love,
Your cookie-eater grandchild.
```

e. From the same terminal window, issue the command below to encrypt the text file. The command will use AES-256 to encrypt the text file and save the encrypted version as message.enc. OpenSSL will ask for a password and for password confirmation. Provide the password as requested and be sure to remember the password.

[analyst@secOps lab.support.files]$ openssl aes-256-cbc -in letter_to_grandma.txt -out message.enc

```
[analyst@secOps lab.support.files]$ openssl aes-256-cbc -in letter_to_grandma.txt -o
ut message.enc
enter aes-256-cbc encryption password:
Verifying - enter aes-256-cbc encryption password:
[analyst@secOps lab.support.files]$
```

f. **cat message.enc**

```
[analyst@secOps lab.support.files]$  cat message.enc
Salted__
files]$
```

g. To make the file readable, run the OpenSSL command again, but this time add the -a option. The -a option tells OpenSSL to encode the encrypted message using a different encoding method of Base64 before storing the results in a file.

[analyst@secOps lab.support.files]$ openssl aes-256-cbc -a -in letter_to_grandma.txt -out message.enc

```
[analyst@secOps lab.support.files]$ openssl aes-256-cbc -a -in letter_to_grandma.txt
 -out message.enc
enter aes-256-cbc encryption password:
Verifying - enter aes-256-cbc encryption password:
[analyst@secOps lab.support.files]$
```

h. **cat message.enc**

```
[analyst@secOps lab.support.files]$ cat message.enc
U2FsdGVkX1+UKTA1/M7OnWohih3Gl7aCGLufs2Fn8R2IP2wHLSrU85rB2qH318/d
/T13jkC2wRzmXNRI2vt77nCw1vxZ/ntUl4zNDLgfPQvPm8OTv9LTt7RrHEMI/15D
zsNbL9yPIODuqdAlkvmxqEqX8WDV6gfAKTgqYnvNWrwB3PvIGXBc89nOU1jSCOea
nTTchJMY7IdaPSqmZVo+qzdgSPMFJKcmbOI/Gs+DLs5TbJSIbLURvT8eH3R6AHu6
uxAilyukHH//32itz5R48UrY923CQ9bdFVaBc7yOZPHluLSLThlFrCLuv9nCv66S
MCKdSuilqTxN+lneAarcbiz+SJPznyURKDTxAWt1G2U=
```

Step2: Decrypting Messages with OpenSSL

With a similar OpenSSL command, it is possible to decrypt message.enc.

a. Use the command below to decrypt message.enc:

[analyst@secOps lab.support.files]$ openssl aes-256-cbc –a -d -in message.enc -out decrypted_letter.txt

```
[analyst@secOps lab.support.files]$ openssl aes-256-cbc -a -d -in message.enc -out d
ecrypted_letter.txt
enter aes-256-cbc decryption password:
```

b. OpenSSL will ask for the password used to encrypt the file. Enter the same password again.

c. When OpenSSL finishes decrypting the message.enc file, it saves the decrypted message in a text file called decrypted_letter.txt. Use the cat display the contents of decrypted_letter.txt:

[analyst@secOps lab.support.files]$ cat decrypted_letter.txt

```
[analyst@secOps lab.support.files]$ cat decrypted_letter.txt
Hi Grandma,
I am writing this letter to thank you for the chocolate chip cookies you sent me. I
got them this morning and I have already eaten half of the box! They are absolutely
delicious!

I wish you all the best. Love,
Your cookie-eater grandchild.
```

Practical No.2

Aim: Demonstrate the use of Snort and Firewall Rules.

Part 1: Preparing the Virtual Environment

a. Launch Oracle VirtualBox and change the CyberOps Workstation for Bridged mode

b.Launch the CyberOps Workstation VM, open a terminal and configure its network by executing the sh script.

[analyst@secOps ~]$ sudo ./lab.support.files/scripts/configure_as_dhcp.sh

```
[analyst@secOps ~]$ sudo ./lab.support.files/scripts/configure_as_dhcp.sh
[sudo] password for analyst:
Configuring the NIC to request IP info via DHCP...
Requesting IP information...
IP Configuration successful.

[analyst@secOps ~]$
```

c. Use the **ifconfig** command to verify CyberOps Workstation VM now has an IP address on your local network. You can also test connectivity to a public webserver by pinging www.cisco.com. Use Ctrl+C to stop the pings.

```
[analyst@secOps ~]$ ping www.cisco.com
PING e2867.dsca.akamaiedge.net (96.6.39.125) 56(84) bytes of data.
64 bytes from a96-6-39-125.deploy.static.akamaitechnologies.com (96.6.39.125):
cmp_seq=1 ttl=54 time=74.4 ms
64 bytes from a96-6-39-125.deploy.static.akamaitechnologies.com (96.6.39.125):
cmp_seq=2 ttl=54 time=75.2 ms
64 bytes from a96-6-39-125.deploy.static.akamaitechnologies.com (96.6.39.125):
cmp_seq=3 ttl=54 time=70.3 ms
64 bytes from a96-6-39-125.deploy.static.akamaitechnologies.com (96.6.39.125):
cmp_seq=4 ttl=54 time=110 ms
64 bytes from a96-6-39-125.deploy.static.akamaitechnologies.com (96.6.39.125):
cmp_seq=5 ttl=54 time=114 ms
```

Part 2: Firewall and IDS Logs

Step 1: Real-Time IDS Log Monitoring

a. From the CyberOps Workstation VM, run the script to start mininet.

[analyst@secOps ~]$ sudo ./lab.support.files/scripts/cyberops_extended_topo_no_fw.py

```
[analyst@secOps ~]$ sudo ./lab.support.files/scripts/cyberops_extended_topo_no_f
w.py
[sudo] password for analyst:
Sorry, try again.
[sudo] password for analyst:
*** Adding controller
*** Add switches
*** Add hosts
*** Add links
*** Starting network
*** Configuring hosts
R1 R4 H1 H2 H3 H4 H5 H6 H7 H8 H9 H10 H11
*** Starting controllers
*** Starting switches
*** Add routes
*** Post configure switches and hosts
*** Starting CLI:
mininet>
```

b. From the mininet prompt, open a shell on R1 using the command below:

mininet> xterm R1

```
mininet> xterm R1
mininet>
```

```
"Node: R1"
[root@secOps analyst]#
```

c. From R1's shell, start the Linux-based IDS, Snort.

[root@secOps analyst]# ./lab.support.files/scripts/start_snort.sh

```
ved.
           Copyright (C) 1998-2013 Sourcefire, Inc., et al.
           Using libpcap version 1.8.1
           Using PCRE version: 8.42 2018-03-20
           Using ZLIB version: 1.2.11

           Rules Engine: SF_SNORT_DETECTION_ENGINE  Version 3.0  <Build 1>
           Preprocessor Object: SF_SSLPP  Version 1.1  <Build 4>
           Preprocessor Object: SF_FTPTELNET  Version 1.2  <Build 13>
           Preprocessor Object: SF_GTP  Version 1.1  <Build 1>
           Preprocessor Object: SF_MODBUS  Version 1.1  <Build 1>
           Preprocessor Object: SF_IMAP  Version 1.0  <Build 1>
           Preprocessor Object: SF_SSH  Version 1.1  <Build 3>
           Preprocessor Object: SF_DNP3  Version 1.1  <Build 1>
           Preprocessor Object: SF_SIP  Version 1.1  <Build 1>
           Preprocessor Object: SF_SMTP  Version 1.1  <Build 9>
           Preprocessor Object: SF_SDF  Version 1.1  <Build 1>
           Preprocessor Object: SF_DCERPC2  Version 1.0  <Build 3>
           Preprocessor Object: SF_DNS  Version 1.1  <Build 4>
           Preprocessor Object: SF_REPUTATION  Version 1.1  <Build 1>
           Preprocessor Object: SF_POP  Version 1.0  <Build 1>
Commencing packet processing (pid=1448)
```

d. From the CyberOps Workstation VM mininet prompt, open shells for hosts H5 and H10.

mininet> xterm H5

```
[root@secOps analyst]#  exit
```

mininet> xterm H10

e. H10 will simulate a server on the Internet that is hosting malware. On H10, run the mal_server_start.sh script to start the server.

[root@secOps analyst]# ./lab.support.files/scripts/mal_server_start.sh

```
[root@secOps analyst]# ./lab.support.files/scripts/mal_server_start.sh
[root@secOps analyst]#
```

e. H10 will simulate a server on the Internet that is hosting malware. On H10, run the mal_server_start.sh script to start the server.

f. On H10, use netstat with the -tunpa options to verify that the web server is running. When used as shown below, netstat lists all ports currently assigned to services:

[root@secOps analyst]# netstat -tunpa

```
[root@secOps analyst]# netstat -tunpa
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
PID/Program name
tcp        0      0 0.0.0.0:6666            0.0.0.0:*               LISTEN
1517/nginx: master
[root@secOps analyst]#
```

g. In the R1 terminal window, an instance of Snort is running. To enter more commands on R1, open another R1 terminal by entering the xterm R1 again in the CyberOps Workstation VM terminal window. You may also want to arrange the terminal windows so that you can see and interact with each device.

h. In the new R1 terminal tab, run the tail command with the -f option to monitor the /var/log/snort/alert file in real-time. This file is where snort is configured to record alerts.

[root@sec0ps analyst]# tail -f /var/log/snort/alert

```
"Node: R1"
[root@secOps analyst]# tail -f /var/log/snort/alert
```

i. From H5, use the wget command to download a file named Nimda.Amm.exe. Designed to download content via HTTP, wget is a great tool for downloading files from web servers directly from the command line.

[root@secOps analyst]# wget 209.165.202.133:6666/W32.Nimda.Amm.exe
Or
curl -O 209.165.202.133:6666/W32.Nimda.Amm.exe

```
[root@secOps analyst]# wget 209.165.202.133:6666/W32.Nimda.Amm.exe
--2023-06-25 01:49:35--  http://209.165.202.133:6666/W32.Nimda.Amm.exe
Connecting to 209.165.202.133:6666... connected.
HTTP request sent, awaiting response... 200 OK
Length: 345088 (337K) [application/octet-stream]
Saving to: W32.Nimda.Amm.exe

W32.Nimda.Amm.exe   100%[===================>] 337.00K  --.-KB/s   in 0.008s

2023-06-25 01:49:35 (40.0 MB/s) - W32.Nimda.Amm.exe saved [345088/345088]

[root@secOps analyst]# curl -O 209.165.202.133:6666/W32.Nimda.Amm.exe
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  337k  100  337k    0     0   9.9M      0 --:--:-- --:--:-- --:--:-- 10.2M
[root@secOps analyst]#
```

j. As the malicious file was transiting R1, the IDS, Snort, was able to inspect its payload. The payload matched at least one of the signatures configured in Snort and triggered an alert on the second R1 terminal window (the tab where tail -f is running). The alert entry is show below. Your timestamp will be different:

```
0] {TCP} 209.165.200.235:43732 -> 209.165.202.133:6666
06/25-01:49:55.078139  [**] [1:1000003:0] Malicious Server Hit! [**] [Priority:
0] {TCP} 209.165.200.235:43734 -> 209.165.202.133:6666
```

On H5, use the tcpdump command to capture the event anddownload the malware file again so you can capture the transaction.type command.

"tcpdump –i H5-eth0 –w nimda.download.pcap&"

```
[root@secOps analyst]# tcpdump -i H5-eth0 -w nimda.download.pcap&
[1] 1595
[root@secOps analyst]# tcpdump: listening on H5-eth0, link-type EN10MB (Ethernet
), capture size 262144 bytes
```

Press ENTER a few times to regain control of the shell while tcpdump runs in background.

Now that tcpdump is capturing packets, download the malware again. On H5, re-run the command.

"curl -O 209.165.202.133:6666/W32.Nimda.Amm.exe"

```
[root@secOps analyst]# curl -O 209.165.202.133:6666/W32.Nimda.Amm.exe
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  337k  100  337k    0     0  8219k      0 --:--:-- --:--:-- --:--:-- 8425k
```

m. Stop the capture by bringing tcpdump to foreground with the fg Because tcpdump was the only process sent to background, there is no need to specify the PID. Stop the tcpdump process with Ctrl+C. The tcpdump process stops and displays a summary of the capture. The number of packets may be different for your capture.

[root@secOps analyst]# fg tcpdump -i h5-eth0 -w nimda.download.pcap

```
[root@secOps analyst]# fg tcpdump -i H5-eth0 -w nimda.download.pcap
tcpdump -i H5-eth0 -w nimda.download.pcap
^C43 packets captured
43 packets received by filter
0 packets dropped by kernel
```

n. On H5, Use the ls command to verify the pcap file was in fact saved to disk and has size greater than zero:

[root@secOps analyst]# ls -l

total 1400

```
[root@secOps analyst]# ls -l
total 700
drwxr-xr-x 2 analyst analyst   4096 Mar 22  2018 Desktop
drwxr-xr-x 3 analyst analyst   4096 Mar 22  2018 Downloads
drwxr-xr-x 9 analyst analyst   4096 Jun 25 00:34 lab.support.files
-rw-r--r-- 1 root    root    349393 Jun 25 02:01 nimda.download.pcap
drwxr-xr-x 2 analyst analyst   4096 Mar 21  2018 second_drive
-rw-r--r-- 1 root    root    345088 Jun 25 01:58 W32.Nimda.Amm.exe
```

Step 2: Tuning Firewall Rules Based on IDS Alerts

a. In the CyberOps Workstation VM, start a third R1 terminal window.

mininet > xterm R1

b. In the new R1 terminal window, use the iptables command to list the chains and their rules currently in use:

[root@secOps ~]# iptables -L -v

```
[root@secOps analyst]# iptables -L -v
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination


Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination


Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

[root@secOps analyst]#
```

c. Connections to the malicious server generate packets that must transverse the iptables firewall on R1. Packets traversing the firewall are handled by the FORWARD rule and therefore, that is the chain that will receive the blocking rule. To keep user computers from connecting to the malicious server identified in Step 1, add the following rule to the FORWARD chain on R1:

d. Use the iptables command again to ensure the rule was added to the FORWARD chain. The CyberOps Workstation VM may take a few seconds to generate the output:

[root@secOps ~]# iptables -I FORWARD -p tcp -d 209.165.202.133 --dport 6666 -j DROP

```
[root@secOps analyst]# iptables -I FORWARD -p tcp -d 209.165.202.133 --dport 66
66 -j DROP
[root@secOps analyst]# iptables -L -v
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination


Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

    0     0 DROP       tcp  --  any    any     anywhere             209.165.202.
133      tcp dpt:6666

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
```

e. On H5, try to download the file again:

```
[root@secOps analyst]# wget 209.165.202.133:6666/W32.Nimda.Amm.exe
--2023-06-25 02:15:23--  http://209.165.202.133:6666/W32.Nimda.Amm.exe
Connecting to 209.165.202.133:6666... failed: Connection timed out.
Retrying.

--2023-06-25 02:17:33--  (try: 2)  http://209.165.202.133:6666/W32.Nimda.Amm.exe
Connecting to 209.165.202.133:6666... ^C
```

Part 3: Terminate and Clear Mininet Process

a. Navigate to the terminal used to start Mininet. Terminate the Mininet by entering quit in the main CyberOps VM terminal window.

```
mininet> quit
*** Stopping 0 controllers

*** Stopping 7 terms
*** Stopping 15 links
...............
*** Stopping 3 switches
S5 S9 S10
*** Stopping 13 hosts
R1 R4 H1 H2 H3 H4 H5 H6 H7 H8 H9 H10 H11
*** Done
```

b. After quitting Mininet, clean up the processes started by Mininet. Enter the password cyberops when prompted.

[analyst@secOps scripts]$ sudo mn –c

```
[analyst@secOps ~]$ sudo mn -c
[sudo] password for analyst:
*** Removing excess controllers/ofprotocols/ofdatapaths/pings/noxes
killall controller ofprotocol ofdatapath ping nox_core lt-nox_core ovs-openflowd
 ovs-controller udpbwtest mnexec ivs 2> /dev/null
killall -9 controller ofprotocol ofdatapath ping nox_core lt-nox_core ovs-openfl
owd ovs-controller udpbwtest mnexec ivs 2> /dev/null
pkill -9 -f "sudo mnexec"
*** Removing junk from /tmp
rm -f /tmp/vconn* /tmp/vlogs* /tmp/*.out /tmp/*.log
*** Removing old X11 tunnels
*** Removing excess kernel datapaths
ps ax | egrep -o 'dp[0-9]+' | sed 's/dp/nl:/'
***  Removing OVS datapaths
ovs-vsctl --timeout=1 list-br
ovs-vsctl --timeout=1 list-br
*** Removing all links of the pattern foo-ethX
ip link show | egrep -o '([-_.[:alnum:]]+-eth[[:digit:]]+)'
ip link show
*** Killing stale mininet node processes
pkill -9 -f mininet:
*** Shutting down stale tunnels
pkill -9 -f Tunnel=Ethernet
pkill -9 -f .ssh/mn
```

Practical No.3

Aim: Demonstrate Extract an Executable from a PCAP.

Step 1: open terminal and write this command **"cd ./lab.support.files/pcaps/ "**

Step 2: write **"ls -l"** list command.

```
[analyst@secOps ~]$ cd lab.support.files/pcaps
[analyst@secOps pcaps]$ ls -l
total 4028
-rw-r--r-- 1 analyst analyst  371462 Mar 21  2018 nimda.download.pcap
-rw-r--r-- 1 analyst analyst 3750153 Mar 21  2018 wannacry_download_pcap.pcap
```

Step 3: On command promt **"wireshark-gtk nimda.download.pcap"** (This will open the wireshark UI)

```
[analyst@secOps pcaps]$ wireshark-gtk nimda.download.pcap
```

Step 4: Check HTTP and check host and full URL to download the malware file.

Step 5: right click on TCP which shows top on the list. Then click on Follow TCP Stream.

Step 6: Check the original file name in the Follow TCP Stream window.

Step 7: Now we need to download and check that file by uploading to an online virustotal website.

Find exe file from HTTP>click file> select export obj > Select exe File >Save as > Select Folder> Save

In the HTTP object list window, select the W32.Nimda.Amm.exe file and click Save As at the bottom of the screen.

Step 8: In command prompt "**ls -l** " to check if the file is saved or not.

```
[analyst@secOps pcaps]$ ls -l
total 4368
-rw-r--r-- 1 analyst analyst  371462 Mar 21  2018 nimda.download.pcap
-rw-r--r-- 1 analyst analyst  345088 Jun 25 03:01 W32.Nimda.Amm.exe
-rw-r--r-- 1 analyst analyst 3750153 Mar 21  2018 wannacry_download_pcap.pcap
```

Step 9: to check the file information put this command **"file W32.Nimda.Amm.exe"**

```
[analyst@secOps pcaps]$ file W32.Nimda.Amm.exe
W32.Nimda.Amm.exe: PE32+ executable (console) x86-64, for MS Windows
```

Practical No.4

Aim: Demonstrate Analysis of DNS Traffic.

Part 1: Capture DNS Traffic
Part 2: Explore DNS Query Traffic
Part 3: Explore DNS Response Traffic

Solution:
Part 1: Capture DNS Traffic

Step 1: Open **Wireshark** and start a Wireshark capture by double clicking a network interface with traffic.

Step 2: At the Command Prompt, enter **ipconfig /flushdns** clear the DNS cache.

Step 3: Enter **nslookup** at the prompt to enter the nslookup interactive mode.

Step 4: Enter the domain name of a website. The domain name www.cisco.com

```
> www.cisco.com
Server:  UnKnown
Address:  192.168.0.1

Non-authoritative answer:
Name:    e2867.dsca.akamaiedge.net
Addresses:  2600:1417:75:d9f::b33
          2600:1417:75:d8a::b33
          23.10.37.140
Aliases:  www.cisco.com
          www.cisco.com.akadns.net
          wwwds.cisco.com.edgekey.net
          wwwds.cisco.com.edgekey.net.globalredir.akadns.net
```

Step 5: type **exit in prompt it will exit the nslookup**

```
          2600:1417:75:d8a::b33
          23.10.37.140
Aliases:  www.cisco.com
          www.cisco.com.akadns.net
          wwwds.cisco.com.edgekey.net
          wwwds.cisco.com.edgekey.net.globalredir.akadns.net

> exit
```

Part 2: Explore DNS Query Traffic.

Step 1: Observe the traffic captured in the Wireshark Packet List pane. Enter **udp.port == 53** in the filter box and click the arrow (or press enter) todisplay only DNS packets.

Select the DNS packet labeled **Standard query 0x0002 A www.cisco.com**.In thePacket Details pane, notice this packet has Ethernet II, Internet Protocol Version 4, User Datagram Protocol and Domain Name System (query).

Expand **Ethernet II** to view the details. Observe the source and destinationfields.

What are the source and destination MAC addresses? Which network interfaces are these MAC addresses associated with?

In this example, the source MAC address is associated with the NIC on the PC andthe destination MAC address is associated with the default gateway. If there is a local DNS server, the destination MAC address would be the MAC address of the local DNS server.

Expand **Internet Protocol Version 4**. Observe the source and destination IPv4 addresses.

What are the source and destination IP addresses? Which network interfaces are theseIP addresses associated with?

In this example, the source IP address is associated with the NIC on the PC and the destination IP address is associated with the DNS server.

Expand the **User Datagram Protocol**. Observe the source and destination ports.

What are the source and destination ports? What is the default DNS port number?

The source port number is 58461 and the destination port is 53, which is the defaultDNS port number.

Open a Command Prompt and enter **arp –a** and **ipconfig /all** to record the MAC andIP addresses of the PC

```
C:\Users\singh>arp -a

Interface: 192.168.56.1 --- 0xd
  Internet Address      Physical Address      Type
  192.168.56.255        ff-ff-ff-ff-ff-ff     static
  224.0.0.2             01-00-5e-00-00-02     static
  224.0.0.22            01-00-5e-00-00-16     static
  224.0.0.251           01-00-5e-00-00-fb     static
  224.0.0.252           01-00-5e-00-00-fc     static
  239.255.255.250       01-00-5e-7f-ff-fa     static

Interface: 192.168.0.109 --- 0x11
  Internet Address      Physical Address      Type
  192.168.0.1           18-a6-f7-ab-04-dc     dynamic
  192.168.0.255         ff-ff-ff-ff-ff-ff     static
  224.0.0.2             01-00-5e-00-00-02     static
  224.0.0.22            01-00-5e-00-00-16     static
  224.0.0.251           01-00-5e-00-00-fb     static
  224.0.0.252           01-00-5e-00-00-fc     static
  239.255.255.250       01-00-5e-7f-ff-fa     static
  255.255.255.255       ff-ff-ff-ff-ff-ff     static
```

```
thernet adapter VirtualBox Host-Only Network:

   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : VirtualBox Host-Only Ethernet Adapter
   Physical Address. . . . . . . . . : 0A-00-27-00-00-0D
   DHCP Enabled. . . . . . . . . . . : No
   Autoconfiguration Enabled . . . . : Yes
   Link-local IPv6 Address . . . . . : fe80::bc65:b322:40e8:7df5%13(Preferred)
   IPv4 Address. . . . . . . . . . . : 192.168.56.1(Preferred)
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . :
   DHCPv6 IAID . . . . . . . . . . . : 671744039
   DHCPv6 Client DUID. . . . . . . . : 00-01-00-01-29-F4-0B-77-1C-6F-65-93-DA-5F
   NetBIOS over Tcpip. . . . . . . . : Enabled

ireless LAN adapter Local Area Connection* 1:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : Microsoft Wi-Fi Direct Virtual Adapter
   Physical Address. . . . . . . . . : D2-37-45-D5-F0-BE
   DHCP Enabled. . . . . . . . . . . : Yes
   Autoconfiguration Enabled . . . . : Yes
```

Compare the MAC and IP addresses in the Wireshark results to the results fromthe **ipconfig /all** results. What is your observation?

The IP and MAC addresses captured in the Wireshark results are the same as the addresses listed in arp – a and ipconfig /all command.

Expand **Domain Name System (query)** in the Packet Details pane. Then expandthe **Flags** and **Queries**.

Observe the results. The flag is set to do the query recursively to query for the IP address to www.cisco.com.

```
> Ethernet II, Src: Tp-Link1_d5:f0:be (d0:37:45:d5:f0:be), Dst: Tp-Link1_ab:04:dc
> Internet Protocol Version 4, Src: 192.168.0.109, Dst: 192.168.0.1
> User Datagram Protocol, Src Port: 49750, Dst Port: 53
v Domain Name System (query)
    Transaction ID: 0x0002
  v Flags: 0x0100 Standard query
      0... .... .... .... = Response: Message is a query
      .000 0... .... .... = Opcode: Standard query (0)
      .... ..0. .... .... = Truncated: Message is not truncated
      .... ...1 .... .... = Recursion desired: Do query recursively
      .... .... .0.. .... = Z: reserved (0)
      .... .... ...0 .... = Non-authenticated data: Unacceptable
    Questions: 1
    Answer RRs: 0
    Authority RRs: 0
    Additional RRs: 0
  > Queries
    [Response In: 101126]
```

Part 3: Explore DNS Response Traffic

Step 1: Select the corresponding response DNS packet labeled **Standard query response 0x0002 A www.cisco.com**.

*Wi-Fi

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

udp.port==53

No.	Time	Source	Destination	Protocol	Length	Info
1011…	402.079863	192.168.0.1	192.168.0.109	DNS	133	Standard query response 0x0001 No such name PTR 1.0.168.192
1011…	407.078114	192.168.0.109	192.168.0.1	DNS	69	Standard query 0x0002 A cisco.com
1011…	407.083314	192.168.0.1	192.168.0.109	DNS	85	Standard query response 0x0002 A cisco.com A 72.163.4.185
1011…	407.087650	192.168.0.109	192.168.0.1	DNS	69	Standard query 0x0003 AAAA cisco.com
1011…	407.305034	192.168.0.1	192.168.0.109	DNS	97	Standard query response 0x0003 AAAA cisco.com AAAA 2001:420
1094…	446.745262	192.168.0.109	192.168.0.1	DNS	74	Standard query 0xd5a7 A www.google.com
1094…	446.753348	192.168.0.1	192.168.0.109	DNS	90	Standard query response 0xd5a7 A www.google.com A 142.250.6
1098…	544.429530	192.168.0.109	192.168.0.1	DNS	95	Standard query 0x2845 A optimizationguide-pa.googleapis.com
1098…	544.440411	192.168.0.1	192.168.0.109	DNS	111	Standard query response 0x2845 A optimizationguide-pa.googl
1098…	544.510819	192.168.0.109	192.168.0.1	DNS	71	Standard query 0xe965 A i.ytimg.com

Step 2: Expand **Domain Name System (response)**. Then expand the **Flags**, **Queries**,and **Answers**. Observe the results.

1011…	407.078114	192.168.0.109	192.168.0.1	DNS	69	Standard query 0x0002 A cisco.com
1011…	407.083314	192.168.0.1	192.168.0.109	DNS	85	Standard query response 0x0002 A cisco
1011…	407.087650	192.168.0.109	192.168.0.1	DNS	69	Standard query 0x0003 AAAA cisco.com
1011…	407.305034	192.168.0.1	192.168.0.109	DNS	97	Standard query response 0x0003 AAAA ci
1094…	446.745262	192.168.0.109	192.168.0.1	DNS	74	Standard query 0xd5a7 A www.google.com
1094…	446.753348	192.168.0.1	192.168.0.109	DNS	90	Standard query response 0xd5a7 A www.g
1098…	544.429530	192.168.0.109	192.168.0.1	DNS	95	Standard query 0x2845 A optimizationgu
1098…	544.440411	192.168.0.1	192.168.0.109	DNS	111	Standard query response 0x2845 A optim
1098…	544.510819	192.168.0.109	192.168.0.1	DNS	71	Standard query 0xe965 A i.ytimg.com

```
> Frame 101125: 69 bytes on wire (552 bits), 69 bytes captured (552 bits) on interface \Device\NPF_{B0FFBDC5-FBAB-48
> Ethernet II, Src: Tp-LinkT_d5:f0:be (d0:37:45:d5:f0:be), Dst: Tp-LinkT_ab:04:dc (18:a6:f7:ab:04:dc)
> Internet Protocol Version 4, Src: 192.168.0.109, Dst: 192.168.0.1
> User Datagram Protocol, Src Port: 49750, Dst Port: 53
v Domain Name System (query)
    Transaction ID: 0x0002
  v Flags: 0x0100 Standard query
      0... .... .... .... = Response: Message is a query
      .000 0... .... .... = Opcode: Standard query (0)
      .... ..0. .... .... = Truncated: Message is not truncated
      .... ...1 .... .... = Recursion desired: Do query recursively
      .... .... .0.. .... = Z: reserved (0)
      .... .... ...0 .... = Non-authenticated data: Unacceptable
    Questions: 1
    Answer RRs: 0
    Authority RRs: 0
    Additional RRs: 0
  > Queries
    [Response In: 101126]
```

Practical No 5

Aim: Create your own syslog Server

Step 1: To check whether rsyslog services already running or not use above command

"sudo systemctl status rsyslog"

```
ubuntu@ubuntu2004:~$ sudo systemctl status rsyslog
Unit rsyslog.service could not be found.
```

Step 2: In case not installed or running, install rsyslog using the following commands:

"sudo apt-get update"

"sudo apt-get install rsyslog"

```
ubuntu@ubuntu2004:~$ sudo apt-get install rsyslog
Reading package lists... Done
Building dependency tree
Reading state information... Done
Suggested packages:
  rsyslog-mysql | rsyslog-pgsql rsyslog-mongodb rsyslog-doc rsyslog-openssl
  | rsyslog-gnutls rsyslog-gssapi rsyslog-relp
The following NEW packages will be installed:
  rsyslog
0 upgraded, 1 newly installed, 0 to remove and 308 not upgraded.
Need to get 0 B/427 kB of archives.
After this operation, 1,695 kB of additional disk space will be used.
Selecting previously unselected package rsyslog.
(Reading database ... 148664 files and directories currently installed.)
Preparing to unpack .../rsyslog_8.2001.0-1ubuntu1.3_amd64.deb ...
Unpacking rsyslog (8.2001.0-1ubuntu1.3) ...
Setting up rsyslog (8.2001.0-1ubuntu1.3) ...
The user `syslog' is already a member of `adm'.
The user `syslog' is already a member of `tty'.
```

Step 3: Open rsyslog configuration file

"sudo nano /etc/rsyslog.conf"

```
ubuntu@ubuntu2004:~$ sudo nano /etc/rsyslog.conf
```

Step 4: Uncomment above four lines that enable udp and tcp port binding:

```
module(load="imuxsock") # provides support for local system logging
#module(load="immark")  # provides --MARK-- message capability

# provides UDP syslog reception
module(load="imudp")
input(type="imudp" port="514")

# provides TCP syslog reception
module(load="imtcp")
input(type="imtcp" port="514")
```

Step 5: Add template right before GLOBAL DIRECTIVES section.

$template remote-incoming-logs,"/var/log/%HOSTNAME%/%PROGRAMNAME%.log"

***.* ?remote-incoming-logs**

```
# provides TCP syslog reception
module(load="imtcp")
input(type="imtcp" port="514")

$template remote-incoming-logs,"/var/log/%HOSTNAME%/%PROGRAMNAME%.log"
*.* ?remote-incoming-logs
# provides kernel logging support and enable non-kernel klog messages
module(load="imklog" permitnonkernelfacility="on")

###########################
#### GLOBAL DIRECTIVES ####
###########################
```

Step 6: Save and restart rsyslog service:

"sudo systemctl restart rsyslog"

```
ubuntu@ubuntu2004:~$ sudo systemctl restart rsyslog
```

Step 7: Confirme that rsyslog service is listening on configured ports

"ss -tunelp | grep 514"

```
ubuntu@ubuntu2004:~$ ss -tunelp | grep 514
udp     UNCONN   0        0                0.0.0.0:514            0.0.0.0:*
 ino:86633 sk:4 <->
udp     UNCONN   0        0                   [::]:514               [::]:*
 ino:86634 sk:8 v6only:1 <->
tcp     LISTEN   0        25               0.0.0.0:514            0.0.0.0:*
 ino:86637 sk:9 <->
tcp     LISTEN   0        25                  [::]:514               [::]:*
 ino:86638 sk:d v6only:1 <->
```

Step 8: Allow rsyslog firewall port rules

"sudo ufw allow 514/tcp" "sudo

ufw allow 514/udp"

```
ubuntu@ubuntu2004:~$ sudo ufw allow 514/tcp
Rules updated
Rules updated (v6)
ubuntu@ubuntu2004:~$ sudo ufw allow 514/udp
Skipping adding existing rule
Skipping adding existing rule (v6)
```

Step 9: To verify configuration, run the following command:

"sudo rsyslogd -N1 -f /etc/rsyslog.con

```
ubuntu@ubuntu2004:~$ sudo rsyslogd -N1 -f /etc/rsyslog.conf
rsyslogd: version 8.2001.0, config validation run (level 1), master config /etc/
rsyslog.conf
rsyslogd: End of config validation run. Bye.
```

Practical No.6

Aim: Configure your Linux system to send syslog messages to a syslog server and Read them.

Step 1: Install and configure rsyslog server first for that please refer practical no 5.

Step 2: Open kali linux and install rsyslog using the following commands

"sudo apt-get update"

```
┌──(kali㉿kali)-[~]
└─$ sudo apt update
[sudo] password for kali:
Get:1 http://kali.download/kali kali-rolling InRelease [41.2 kB]
Get:2 http://kali.download/kali kali-rolling/main amd64 Packages [19.4 MB]
Get:3 http://kali.download/kali kali-rolling/main amd64 Contents (deb) [45.1 MB]
Get:4 http://kali.download/kali kali-rolling/contrib amd64 Packages [119 kB]
Get:5 http://kali.download/kali kali-rolling/contrib amd64 Contents (deb) [176 kB]
Get:6 http://kali.download/kali kali-rolling/non-free amd64 Packages [217 kB]
Get:7 http://kali.download/kali kali-rolling/non-free amd64 Contents (deb) [930 kB]
Fetched 66.0 MB in 5min 49s (189 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
781 packages can be upgraded. Run 'apt list --upgradable' to see them.
```

"sudo apt-get install rsyslog"

```
┌──(kali㉿kali)-[~]
└─$ sudo apt-get install rsyslog
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libestr0 libfastjson4 liblognorm5
Suggested packages:
  rsyslog-mysql | rsyslog-pgsql rsyslog-mongodb rsyslog-doc rsyslog-openssl | rsyslog-gnutls rsyslog-gssapi rs
The following NEW packages will be installed:
  libestr0 libfastjson4 liblognorm5 rsyslog
0 upgraded, 4 newly installed, 0 to remove and 781 not upgraded.
Need to get 829 kB of archives.
After this operation, 2,280 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://kali.download/kali kali-rolling/main amd64 libestr0 amd64 0.1.11-1 [9,204 B]
Get:2 http://kali.download/kali kali-rolling/main amd64 libfastjson4 amd64 1.2304.0-1 [28.9 kB]
Get:3 http://kali.download/kali kali-rolling/main amd64 liblognorm5 amd64 2.0.6-4 [67.2 kB]
Get:4 http://kali.download/kali kali-rolling/main amd64 rsyslog amd64 8.2302.0-1 [723 kB]
Fetched 829 kB in 23s (36.1 kB/s)
Selecting previously unselected package libestr0:amd64.
(Reading database ... 393892 files and directories currently installed.)
```

Step 3: Open rsyslog configuration file
"sudo nano /etc/rsyslog.conf"

```
#
# For more information install rsyslog-doc and see
# /usr/share/doc/rsyslog-doc/html/configuration/index.html


#################
#### MODULES ####
#################

module(load="imuxsock") # provides support for local system logging
module(load="imklog")   # provides kernel logging support
#module(load="immark")  # provides --MARK-- message capability

# provides UDP syslog reception
#module(load="imudp")
#input(type="imudp" port="514")

# provides TCP syslog reception
```

Step 4: Add above lines at the end of the file
@192.168.137.50:514
***.* @@192.168.137.50:514**

```
kern.*                          -/var/log/kern.log
mail.*                          -/var/log/mail.log
user.*                          -/var/log/user.log

#
# Emergencies are sent to everybody logged in.
#
*.emerg                         :omusrmsg:*
@192.168.137.50:514
*.*@@192.168.137.50:514
```

Note: You can enable to send logs over UDP. For TCP use @@ , instead of one

Step 5: For the end add these following variables in case when the rsyslog server goes down.

$ActionQueueFileName queue
$ActionQueueMaxDiskSpace 1g
$ActionQueueSaveOnShutdown on
$ActionQueueType LinkedList
$ActionResumeRetryCount -1

```
*.emerg                         :omusrmsg:*
@192.168.137.50:514
*.*@@192.168.137.50:514

@ActionQueueFileName queue
@ActionQueueMaxDiskSpace 1g
@ActionQueueSaveOnShutdown on
@ActionQueueType LinkedList
@ActionResumeRetryCount -1
```

Step 6: Then Save and exit the file

Step 7: restart the rsyslog service
"sudo systemctl restart rsyslog"

```
┌──(kali㉿kali)-[~]
└─$ sudo systemctl restart rsyslog
```

Verify the logs
After the configuration is completed on the client machine, we want to verify that everything went well.

Step 8: Go to your Rsyslog server to verify the logs from your client machine
"ls /var/log/"

In my case, the directory named kali is the name of my client machine which I am currently using. We will enter this directory and see something like this:

Step 9: To check logs use the following command: Let's for example inspect rsyslogd.log.
"sudo tail -f /var/log/kali/rsyslogd.log"

```
ubuntu@ubuntu2004:~$ sudo tail -f /var/log/kali/rsyslogd.log
2022-05-18T05:47:20-04:00 kali rsyslogd: [origin software="rsyslogd" swVersion="
8.2204.0" x-pid="8621" x-info="https://www.rsyslog.com"] start
2022-05-18T05:47:20-04:00 kali rsyslogd: [origin software="rsyslogd" swVersion="
8.2204.0" x-pid="4842" x-info="https://www.rsyslog.com"] exiting on signal 15.
2022-05-18T05:47:20-04:00 kali rsyslogd: imuxsock: Acquired UNIX socket '/run/sy
stemd/journal/syslog' (fd 3) from systemd.  [v8.2204.0]
2022-05-18T05:47:20-04:00 kali rsyslogd: [origin software="rsyslogd" swVersion="
8.2204.0" x-pid="8621" x-info="https://www.rsyslog.com"] start
```

Practical No.7

Aim: Install and Run Splunk on Linux.

Step1: Download Splunk Installer
"cd /tmp && wget https://download.splunk.com/products/splunk/releases/7.1.1/linux/splunk-7.1.1-8f0ead9ec3db-linux-2.6-amd64.deb"

```
ubuntu@ubuntu:~$ cd /tmp && wget https://download.splunk.com/products/splunk
/releases/7.1.1/linux/splunk-7.1.1-8f0ead9ec3db-linux-2.6-amd64.deb
--2023-06-25 08:21:23--  https://download.splunk.com/products/splunk/release
s/7.1.1/linux/splunk-7.1.1-8f0ead9ec3db-linux-2.6-amd64.deb
Resolving download.splunk.com (download.splunk.com)... 18.66.53.32, 18.66.53
.89, 18.66.53.94, ...
Connecting to download.splunk.com (download.splunk.com)|18.66.53.32|:443...
connected.
HTTP request sent, awaiting response... 200 OK
Length: 263297630 (251M) [binary/octet-stream]
Saving to: 'splunk-7.1.1-8f0ead9ec3db-linux-2.6-amd64.deb'

.deb                  5%[>                  ]  13.27M  1.24MB/s    eta eb
splunk-7.1.1-8f0ea 100%[================>] 251.10M  2.02MB/s    in 3m 10s

2023-06-25 08:24:34 (1.32 MB/s) - 'splunk-7.1.1-8f0ead9ec3db-linux-2.6-amd64
.deb' saved [263297630/263297630]
```

Step 2: Install Splunk
"sudo dpkg -i splunk-7.1.1-8f0ead9ec3db-linux-2.6-amd64.deb"

```
ubuntu@ubuntu:/tmp$ sudo dpkg -i splunk-7.1.1-8f0ead9ec3db-linux-2.6-amd64.d
eb
[sudo] password for ubuntu:
Selecting previously unselected package splunk.
(Reading database ... 175043 files and directories currently installed.)
Preparing to unpack splunk-7.1.1-8f0ead9ec3db-linux-2.6-amd64.deb ...
Unpacking splunk (7.1.1) ...
Setting up splunk (7.1.1) ...
complete
```

Step 3: Enable the Splunk to start at boot
Press enter key till you reach to the end of the agreement, then you have to accept the license agreement by typing "y".
Then you have to enter the initial admin password and use this password to access the web portal

```
ubuntu@ubuntu:/tmp$ sudo /opt/splunk/bin/splunk enable boot-start
SPLUNK SOFTWARE LICENSE AGREEMENT

THIS SPLUNK SOFTWARE LICENSE AGREEMENT ("AGREEMENT") GOVERNS THE LICENSING,
INSTALLATION AND USE OF SPLUNK SOFTWARE. BY DOWNLOADING AND/OR INSTALLING SP
LUNK
SOFTWARE: (A) YOU ARE INDICATING THAT YOU HAVE READ AND UNDERSTAND THIS
AGREEMENT, AND AGREE TO BE LEGALLY BOUND BY IT ON BEHALF OF THE COMPANY,
GOVERNMENT, OR OTHER ENTITY FOR WHICH YOU ARE ACTING (FOR EXAMPLE, AS AN
EMPLOYEE OR GOVERNMENT OFFICIAL) OR, IF THERE IS NO COMPANY, GOVERNMENT OR O
THER
ENTITY FOR WHICH YOU ARE ACTING, ON BEHALF OF YOURSELF AS AN INDIVIDUAL; AND
 (B)
YOU REPRESENT AND WARRANT THAT YOU HAVE THE AUTHORITY TO ACT ON BEHALF OF AN
Do you agree with this license? [y/n]: y
```

```
   * 8 total printable ASCII character(s).
Please enter a new password:
Please confirm new password:
Copying '/opt/splunk/etc/openldap/ldap.conf.default' to '/opt/splunk/etc/ope
nldap/ldap.conf'.
Generating RSA private key, 2048 bit long modulus
.................................................+++
..........+++
e is 65537 (0x10001)
writing RSA key

Generating RSA private key, 2048 bit long modulus
.....+++
...............................+++
e is 65537 (0x10001)
writing RSA key

Moving '/opt/splunk/share/splunk/search_mrsparkle/modules.new' to '/opt/splu
nk/share/splunk/search_mrsparkle/modules'.
Init script installed at /etc/init.d/splunk.
Init script is configured to run at boot.
```

Step 4: Start the Splunk service
"sudo service splunk start"

```
ubuntu@ubuntu:/tmp$ sudo service splunk start
ubuntu@ubuntu:/tmp$
```

Step 5: Check splunk service Status
"sudo service splunk status"

```
● splunk.service - LSB: Start splunk
     Loaded: loaded (/etc/init.d/splunk; generated)
     Active: active (running) since Sun 2023-06-25 08:46:44 PDT; 46s ago
       Docs: man:systemd-sysv-generator(8)
    Process: 4025 ExecStart=/etc/init.d/splunk start (code=exited, status=0>
      Tasks: 166 (limit: 2214)
     Memory: 487.4M
     CGroup: /system.slice/splunk.service
             ├─4088 splunkd -p 8089 start
             ├─4089 [splunkd pid=4088] splunkd -p 8089 start [process-runne>
             ├─4100 mongod --dbpath=/opt/splunk/var/lib/splunk/kvstore/mong>
             ├─4172 /opt/splunk/bin/python -O /opt/splunk/lib/python2.7/sit>
             ├─4174 /opt/splunk/bin/splunkd instrument-resource-usage -p 80>
             ├─4408 [splunkd pid=4088] [search-launcher]
             └─4409 [splunkd pid=4088] [search-launcher] [process-runner]

Jun 25 08:46:31 ubuntu splunk[4026]:         All installed files intact.
Jun 25 08:46:31 ubuntu splunk[4026]:         Done
Jun 25 08:46:31 ubuntu splunk[4026]: All preliminary checks passed.
Jun 25 08:46:31 ubuntu splunk[4026]: Starting splunk server daemon (splunkd>
Jun 25 08:46:31 ubuntu splunk[4026]: Done
Jun 25 08:46:44 ubuntu splunk[4026]: Waiting for web server at http://127.0>
Jun 25 08:46:44 ubuntu splunk[4026]: If you get stuck, we're here to help.
lines 1-23
```

Step 6: Splunk will be started at port 8000. You can access the application via URL "http://localhost:8000/". To logged in into the app enter username as "admin" then enter your password. In my case the password is "admin!123"

Step 7: After you logged in into the app you can see the above screen

Practical No.8

Aim: Install and Configure ELK on Linux.

Part 1: Installing java

Step 1: write the below command and update and install the jdk

"sudo apt update"

```
ubuntu@ubuntu:~/Desktop$ sudo apt update
Get:1 https://artifacts.elastic.co/packages/7.x/apt stable InRelease [13.7 kB]
Hit:2 http://security.ubuntu.com/ubuntu focal-security InRelease
Get:3 https://artifacts.elastic.co/packages/7.x/apt stable/main amd64 Packages [
111 kB]
Get:4 https://artifacts.elastic.co/packages/7.x/apt stable/main i386 Packages [8
1.3 kB]
Hit:5 http://us.archive.ubuntu.com/ubuntu focal InRelease
Hit:6 http://us.archive.ubuntu.com/ubuntu focal-updates InRelease
Hit:7 http://us.archive.ubuntu.com/ubuntu focal-backports InRelease
Fetched 206 kB in 3s (63.5 kB/s)
Reading package lists... Done
Building dependency tree
Reading state information... Done
144 packages can be upgraded. Run 'apt list --upgradable' to see them.
```

Install java
"sudo apt install default-jre"

```
ubuntu@ubuntu:~/Desktop$ sudo apt install default-jre
[sudo] password for ubuntu:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  ca-certificates-java default-jre-headless fonts-dejavu-extra java-common
  libatk-wrapper-java libatk-wrapper-java-jni openjdk-11-jre
  openjdk-11-jre-headless
Suggested packages:
  fonts-ipafont-gothic fonts-ipafont-mincho fonts-wqy-microhei
  | fonts-wqy-zenhei
The following NEW packages will be installed:
  ca-certificates-java default-jre default-jre-headless fonts-dejavu-extra
  java-common libatk-wrapper-java libatk-wrapper-java-jni openjdk-11-jre
  openjdk-11-jre-headless
0 upgraded, 9 newly installed, 0 to remove and 144 not upgraded.
```

Step 2: check the java version by this command "java -version"

```
ubuntu@ubuntu:~/Desktop$ java --version
openjdk 11.0.19 2023-04-18
OpenJDK Runtime Environment (build 11.0.19+7-post-Ubuntu-0ubuntu120.04.1)
OpenJDK 64-Bit Server VM (build 11.0.19+7-post-Ubuntu-0ubuntu120.04.1, mixed moc
e, sharing)
ubuntu@ubuntu:~/Desktop$
```

Part 2: Install and Configure the Elasticsearch

Elastic Search
Elasticsearch store logs coming from external sources and offers real-time distributed search and analytics with the RESTful web interface.

Step 1: Download and install the GPG signing key.

“curl -fsSL https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo apt-key add -”

```
ubuntu@ubuntu:~/Desktop$ curl -fsSL https://artifacts.elastic.co/GPG-KEY-elastic
search | sudo apt-key add -
OK
```

Step 2: Set up the Elasticsearch repository on your system by running the below command.

“echo "deb https://artifacts.elastic.co/packages/7.x/apt stable main" | sudo tee -a /etc/apt/sources.list.d/elastic-7.x.list”

```
ubuntu@ubuntu:~/Desktop$ echo "deb https://artifacts.elastic.co/packages/7.x/apt
 stable main" | sudo tee -a /etc/apt/sources.list.d/elastic-7.x.list
deb https://artifacts.elastic.co/packages/7.x/apt stable main
```

Step 3: Update the repository cache and then install the Elasticsearch package.

“sudo apt update”

```
ubuntu@ubuntu:~/Desktop$ sudo apt update
Get:1 https://artifacts.elastic.co/packages/7.x/apt stable InRelease [13.7 kB]
Hit:2 http://security.ubuntu.com/ubuntu focal-security InRelease
Get:3 https://artifacts.elastic.co/packages/7.x/apt stable/main amd64 Packages [
111 kB]
Get:4 https://artifacts.elastic.co/packages/7.x/apt stable/main i386 Packages [8
1.3 kB]
Hit:5 http://us.archive.ubuntu.com/ubuntu focal InRelease
Hit:6 http://us.archive.ubuntu.com/ubuntu focal-updates InRelease
Hit:7 http://us.archive.ubuntu.com/ubuntu focal-backports InRelease
Fetched 206 kB in 3s (63.5 kB/s)
Reading package lists... Done
Building dependency tree
Reading state information... Done
144 packages can be upgraded. Run 'apt list --upgradable' to see them.
```

“sudo apt install elasticsearch”

```
ubuntu@ubuntu:~/Desktop$ sudo apt install elasticsearch
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  elasticsearch
0 upgraded, 1 newly installed, 0 to remove and 144 not upgraded.
Need to get 317 MB of archives.
After this operation, 530 MB of additional disk space will be used.
Get:1 https://artifacts.elastic.co/packages/7.x/apt stable/main amd64 elasticsea
rch amd64 7.17.10 [317 MB]
Fetched 47.1 MB in 1min 20s (587 kB/s)
Selecting previously unselected package elasticsearch.
(Reading database ... 189994 files and directories currently installed.)
Preparing to unpack .../elasticsearch_7.17.10_amd64.deb ...
Creating elasticsearch group... OK
Creating elasticsearch user... OK
Unpacking elasticsearch (7.17.10) ...
Setting up elasticsearch (7.17.10) ...
### NOT starting on installation, please execute the following statements to con
figure elasticsearch service to start automatically using systemd
```

Step 4: Edit the Elasticsearch configuration file to set the cluster name for Graylog set up.

“sudo nano /etc/elasticsearch/elasticsearch.yml”

Uncomment

network.host:localhost

http.port:9200

```
# ---------------------------------- Network -----------------------------------
#
# By default Elasticsearch is only accessible on localhost. Set a different
# address here to expose this node on the network:
#
network.host: localhost
#
# By default Elasticsearch listens for HTTP traffic on the first free port it
# finds starting at 9200. Set a specific HTTP port here:
#
```

Step 5: Next, start the Elasticsearch service with the systemctl. Give Elasticsearch little time to start up otherwise, you can get errors about not being able to connect to it.

sudo systemctl start elasticsearch

```
ubuntu@ubuntu:~/Desktop$ sudo systemctl start elasticsearch
ubuntu@ubuntu:~/Desktop$
```

Step 6: Now, run the below command. It will enable Elasticsearch to start every time your server boots:

sudo systemctl enable elasticsearch

```
ubuntu@ubuntu:~/Desktop$ sudo systemctl enable elasticsearch
Synchronizing state of elasticsearch.service with SysV service script with /lib/
systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable elasticsearch
Created symlink /etc/systemd/system/multi-user.target.wants/elasticsearch.servic
e → /lib/systemd/system/elasticsearch.service.
ubuntu@ubuntu:~/Desktop$
```

Step 7: You will then test whether your Elasticsearch service is running. Do it by sending an HTTP request:

curl -X GET "localhost:9200"

```
ubuntu@ubuntu:~/Desktop$ curl -X GET "localhost:9200"
{
  "name" : "ubuntu",
  "cluster_name" : "elasticsearch",
  "cluster_uuid" : "dnVrXuFqQluC0sVRlKxZ3w",
  "version" : {
    "number" : "7.17.10",
    "build_flavor" : "default",
    "build_type" : "deb",
    "build_hash" : "fecd68e3150eda0c307ab9a9d7557f5d5fd71349",
    "build_date" : "2023-04-23T05:33:18.138275597Z",
    "build_snapshot" : false,
    "lucene_version" : "8.11.1",
    "minimum_wire_compatibility_version" : "6.8.0",
    "minimum_index_compatibility_version" : "6.0.0-beta1"
  },
  "tagline" : "You Know, for Search"
}
```

Practical No: 9

Aim: Install and Configure GrayLog on Linux

Part 1: Install Java and Els

Step 1:Install Java and Els (Practical 8)

Step 2: Edit the Elasticsearch configuration file to set the cluster name for Graylog set up.
"sudo nano /etc/elasticsearch/elasticsearch.yml"

```
#
# Use a descriptive name for your cluster:
#
cluster.name: graylog
#
# ---------------------------------- Node ----------------------------------
```

Step 3: Set the cluster name as graylog, as shown below. Then, uncomment the line and below add this line **"action.auto_create_index: false"** then save.

```
# ---------------------------------- Various -----------------------------------
#
# Require explicit names when deleting indices:
#
#action.destructive_requires_name: true
action.auto_create_index: false
# ---------------------------------- Security ----------------------------------
#
```

Step 4: Start the Elasticsearch service to read the new configurations.
"sudo systemctl daemon-reload"
"sudo systemctl start elasticsearch"
"sudo systemctl enable elasticsearch"

```
ubuntu@ubuntu:~/Desktop$ sudo nano /etc/elasticsearch/elasticsearch.yml
ubuntu@ubuntu:~/Desktop$ sudo systemctl daemon-reload
ubuntu@ubuntu:~/Desktop$ sudo systemctl start elasticsearch
ubuntu@ubuntu:~/Desktop$ sudo systemctl enable elasticsearch
Synchronizing state of elasticsearch.service with SysV service script with /lib/
systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable elasticsearch
ubuntu@ubuntu:~/Desktop$
```

Step 5: Elastic search should be now listening on port 9200. Use the curl command to check the Elasticsearch's response
"curl -X GET http://localhost:9200 "

```
ubuntu@ubuntu:~/Desktop$ curl -X GET http://localhost:9200
{
  "name" : "ubuntu",
  "cluster_name" : "graylog",
  "cluster_uuid" : "dnVrXuFqQluC0sVRlKxZ3w",
  "version" : {
    "number" : "7.17.10",
    "build_flavor" : "default",
    "build_type" : "deb",
    "build_hash" : "fecd68e3150eda0c307ab9a9d7557f5d5fd71349",
    "build_date" : "2023-04-23T05:33:18.138275597Z",
    "build_snapshot" : false,
    "lucene_version" : "8.11.1",
    "minimum_wire_compatibility_version" : "6.8.0",
    "minimum_index_compatibility_version" : "6.0.0-beta1"
  },
  "tagline" : "You Know, for Search"
}
```

Part 2: Install MongoDB

MongoDB acts as a database for storing Graylog's configuration. Graylog requires MongoDB v3.6, 4.0 or 4.2. Unfortunately, MongoDB's official repository doesn't have the required MongoDB versions for Ubuntu 20.04. So, we will install MongoDB v3.6 from the Ubuntu base repository.

Step 1: "sudo apt update"

```
ubuntu@ubuntu:~/Desktop$ sudo apt update
Hit:1 https://artifacts.elastic.co/packages/7.x/apt stable InRelease
Hit:2 http://us.archive.ubuntu.com/ubuntu focal InRelease
Hit:3 http://us.archive.ubuntu.com/ubuntu focal-updates InRelease
Get:4 http://security.ubuntu.com/ubuntu focal-security InRelease [114 kB]
Get:5 http://us.archive.ubuntu.com/ubuntu focal-backports InRelease [108 kB]
Fetched 222 kB in 4s (51.6 kB/s)
Reading package lists... Done
Building dependency tree
Reading state information... Done
144 packages can be upgraded. Run 'apt list --upgradable' to see them.
```

"sudo apt install -y mongodb-server"

```
ubuntu@ubuntu:~/Desktop$ sudo apt install -y mongodb-server
Reading package lists... Done
Building dependency tree
Reading state information... Done
mongodb-server is already the newest version (1:3.6.9+really3.6.8+90~g8e540c0b6d
-0ubuntu5.3).
0 upgraded, 0 newly installed, 0 to remove and 144 not upgraded.
ubuntu@ubuntu:~/Desktop$
```

Step 2: Start the MongoDB and enable it on the system start-up.

"sudo systemctl start mongodb"

"sudo systemctl enable mongodb"

```
ubuntu@ubuntu:~/Desktop$ sudo systemctl enable mongodb
Synchronizing state of mongodb.service with SysV service script with /lib/system
d/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable mongodb
ubuntu@ubuntu:~/Desktop$
```

Part 4: Install GrayLog Server

GrayLog Server reads data from Elasticsearch for
search queries comes from users and then displays it for them through the Graylog web interface.

Step 1: Download and install the Graylog 3.3 repository configuration package.

"wget https://packages.graylog2.org/repo/packages/graylog-4.2-repository_latest.deb"

```
ubuntu@ubuntu:~/Desktop$ wget https://packages.graylog2.org/repo/packages/graylo
g-4.2-repository_latest.deb
--2023-06-25 23:16:39--  https://packages.graylog2.org/repo/packages/graylog-4.2
-repository_latest.deb
Resolving packages.graylog2.org (packages.graylog2.org)... 104.21.88.209, 172.67
.153.95, 2606:4700:3035::ac43:995f, ...
Connecting to packages.graylog2.org (packages.graylog2.org)|104.21.88.209|:443..
. connected.
HTTP request sent, awaiting response... 302 Found
Location: https://graylog-package-repository.s3.eu-west-1.amazonaws.com/packages
/graylog-4.2-repository_latest.deb?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Date=2
0230626T061641Z&X-Amz-SignedHeaders=host&X-Amz-Expires=600&X-Amz-Credential=AKIA
IJSI6MCSPXFVDPIA%2F20230626%2Feu-west-1%2Fs3%2Faws4_request&X-Amz-Signature=6c21
9605c118e6563deae52860456cd63adfb6295aa4d156e78a968ac1d02ed0 [following]
--2023-06-25 23:16:41--  https://graylog-package-repository.s3.eu-west-1.amazona
```

"sudo dpkg -i graylog-3.3-repository_latest.deb"

```
ubuntu@ubuntu:~/Desktop$ sudo dpkg -i graylog-4.2-repository_latest.deb
Selecting previously unselected package graylog-4.2-repository.
(Reading database ... 191185 files and directories currently installed.)
Preparing to unpack graylog-4.2-repository_latest.deb ...
Unpacking graylog-4.2-repository (1-4) ...
Setting up graylog-4.2-repository (1-4) ...
```

Step 2: Update the repository cache. **"sudo apt update"**

```
ubuntu@ubuntu:~/Desktop$ sudo apt update
Hit:1 https://artifacts.elastic.co/packages/7.x/apt stable InRelease
Get:2 http://security.ubuntu.com/ubuntu focal-security InRelease [114 kB]
Hit:4 http://us.archive.ubuntu.com/ubuntu focal InRelease
Get:5 http://us.archive.ubuntu.com/ubuntu focal-updates InRelease [114 kB]
Get:6 http://security.ubuntu.com/ubuntu focal-security/main amd64 DEP-11 Metadat
a [59.9 kB]
Get:3 https://packages.graylog2.org/repo/debian stable InRelease [58.1 kB]
Get:7 http://security.ubuntu.com/ubuntu focal-security/universe amd64 DEP-11 Met
adata [95.5 kB]
Get:8 http://us.archive.ubuntu.com/ubuntu focal-backports InRelease [108 kB]
Get:9 http://security.ubuntu.com/ubuntu focal-security/multiverse amd64 DEP-11 M
etadata [940 B]
Get:10 http://us.archive.ubuntu.com/ubuntu focal-updates/main amd64 DEP-11 Metad
ata [275 kB]
```

Step 3: Install the Graylog server using the following command.
"sudo apt install -y graylog-server"

```
ubuntu@ubuntu:~/Desktop$ sudo apt install -y graylog-server
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  graylog-server
0 upgraded, 1 newly installed, 0 to remove and 144 not upgraded.
Need to get 197 MB of archives.
After this operation, 218 MB of additional disk space will be used.
Ign:1 https://packages.graylog2.org/repo/debian stable/4.2 amd64 graylog-server
all 4.2.13-1
Get:1 https://packages.graylog2.org/repo/debian stable/4.2 amd64 graylog-server
all 4.2.13-1 [197 MB]
Fetched 15.6 MB in 42s (373 kB/s)
Selecting previously unselected package graylog-server.
(Reading database ... 191189 files and directories currently installed.)
Preparing to unpack .../graylog-server_4.2.13-1_all.deb ...
Unpacking graylog-server (4.2.13-1) ...
Setting up graylog-server (4.2.13-1) ...
##############################################################################
```

Step 4: You must set a secret to secure the user passwords. Use the pwgen command to generate the secret.
"pwgen -N 1 -s 96"

```
ubuntu@ubuntu:~/Desktop$ sudo apt install pwgen
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  pwgen
0 upgraded, 1 newly installed, 0 to remove and 144 not upgraded.
Need to get 18.1 kB of archives.
After this operation, 52.2 kB of additional disk space will be used.
Get:1 http://us.archive.ubuntu.com/ubuntu focal/universe amd64 pwgen amd64 2.08-
2 [18.1 kB]
Fetched 18.1 kB in 1s (16.9 kB/s)
Selecting previously unselected package pwgen.
(Reading database ... 191212 files and directories currently installed.)
```

```
ubuntu@ubuntu:~/Desktop$ pwgen -N 1 -s 96
dHhrek7amsHYKJ4l0IKuJC6wOPbVZ0nCY7Ea4fPBTzQT5xWmrSpnvHY6Q1ePeBVFs8R2mNEH18RDRqDi
W4DxLvL4xb38DOe8
```

Step 5: **sudo gedit /etc/graylog/server/server.**conf edit the conf file and put
Then, place the secret like below.
sudo nano /etc/graylog/server/server.conf

```
# You MUST set a secret to secure/pepper the stored user passwords here. Use at>
# Generate one by using for example: pwgen -N 1 -s 96
# ATTENTION: This value must be the same on all Graylog nodes in the cluster.
# Changing this value after installation will render all user sessions and encr>
password_secret = dHhrek7amsHYKJ4l0IKuJC6wOPbVZ0nCY7Ea4fPBTzQT5xWmrSpnvHY6Q1ePe>
```

Step 6: Now, generate a hash (sha256) password for the root user (not to be confused with the system user, the root user of graylog is admin).

You will need this password to login to the Graylog web interface. Admin's password can't be changed using the web interface. So, you must edit this variable to set.

Replace password with the choice of your password. Put this command in terminal
"echo -n password | sha256sum"

```
ubuntu@ubuntu:~/Desktop$ echo -n yourpassword | sha256sum
e3c652f0ba0b4801205814f8b6bc49672c4c74e25b497770bb89b22cdeb4e951  -
```

Step 7: Edit the server.conf file again.in terminal
"sudo nano /etc/graylog/server/server.conf"

```
# You MUST set a secret to secure/pepper the stored user passwords here. Use at>
# Generate one by using for example: pwgen -N 1 -s 96
# ATTENTION: This value must be the same on all Graylog nodes in the cluster.
# Changing this value after installation will render all user sessions and encr>
password_secret = dHhrek7amsHYKJ4l0IKuJC6wOPbVZ0nCY7Ea4fPBTzQT5xWmrSpnvHY6Q1ePe>
```

Part 5: Setup Graylog web interface
From version Graylog 2.x, the web interface is being served directly by the Graylog server. Step 1: Enable the Graylog web interface by editing the server.conf file.
"sudo gedit /etc/graylog/server/server.conf"

Put http_bind_address = 192.168.0.10:9000
http_external_uri = http://public_ip:9000/

```
# Default: 127.0.0.1:9000
http_bind_address = 192.168.186.129:9000
#http_bind_address = [2001:db8::1]:9000

#### HTTP publish URI
#
# The HTTP URI of this Graylog node which is used to communicate with the othe
```

Step 2: Start and enable the Graylog service.
Place the below command
"sudo systemctl daemon-reload"
"sudo systemctl start graylog-server"
"sudo systemctl enable graylog-server"

```
ubuntu@ubuntu:~/Desktop$ sudo systemctl enable  graylog-server
Synchronizing state of graylog-server.service with SysV service script with /lib
/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable graylog-server
Created symlink /etc/systemd/system/multi-user.target.wants/graylog-server.servi
ce → /lib/systemd/system/graylog-server.service.
```

Step 3: Keep looking Graylog server startup logs. This log will be useful for you to troubleshoot Graylog in case of any issues.
"sudo tail -f /var/log/graylog-server/server.log"

Step 4: On the successful start of the Graylog server, you should get the following message in the log file.
You will able to see the log file**.**
2020-08-03T16:03:06.326-04:00 INFO [ServerBootstrap] Graylog server up and running.

Access Graylog
The Graylog web interface will now be listening on port 9000. Open your browser and point it to.
"http://ip.add.re.ss:9000" type in browser.

Practical No.10

Aim: Demonstrate Conversion of Data into a Universal Format.

Part 1: Normalize Timestamps in a Log Files.

Step 1: Launch the CyberOps Workstation VM.

Step 2: open terminal and type **"cd /home/analyst/lab.support.files/"**
Then type **"ls –l"**

```
[analyst@secOps lab.support.files]$ ls -l
total 588
-rw-r--r-- 1 analyst analyst    649 Mar 21  2018 apache_in_epoch.log
-rw-r--r-- 1 analyst analyst    126 Mar 21  2018 applicationX_in_epoch.log
drwxr-xr-x 4 analyst analyst   4096 Mar 21  2018 attack_scripts
-rw-r--r-- 1 analyst analyst    102 Mar 21  2018 confidential.txt
-rw-r--r-- 1 analyst analyst   2871 Mar 21  2018 cyops.mn
-rw-r--r-- 1 analyst analyst    255 Jun 25 00:34 decrypted_letter.txt
-rw-r--r-- 1 analyst analyst     75 Mar 21  2018 elk_services
-rw-r--r-- 1 analyst analyst    373 Mar 21  2018 h2_dropbear.banner
```

Step 3: Issue the following AWK command to convert and print the result on the terminal:
Write the command
"awk 'BEGIN {FS=OFS="|"} {$3=strftime("%c",$3)} {print}' applicationX_in_epoch.log"

```
[analyst@secOps lab.support.files]$ awk 'BEGIN {FS=OFS="|"} {$3=strftime("%c",$3
)} {print}' applicationX_in_epoch.log
2|Z|Mon 18 Aug 2008 11:00:00 AM EDT|AF|0
3|N|Tue 19 Aug 2008 11:00:00 AM EDT|AF|89
4|N|Sun 07 Sep 2008 11:00:00 AM EDT|AS|12
1|Z|Mon 08 Sep 2008 11:00:00 AM EDT|AS|67
5|N|Tue 09 Sep 2008 11:00:00 AM EDT|EU|23
6|R|Wed 10 Sep 2008 11:00:00 AM EDT|OC|89
||Wed 31 Dec 1969 07:00:00 PM EST
```

The command above is an AWK script. It may seem complicated. The main structure of the AWK script above is as follows:

- **awk –** This invokes the AWK interpreter.
- **„BEGIN –** This defines the beginning of the script.
- **{} –** This defines actions to be taken in each line of the input text file. An AWK script can have several actions.
- **FS = OFS = "|" –** This defines the field separator (i.e., delimiter) as the bar (|) symbol. Different text files may use different delimiting characters to separate fields.This operator allows the user to define what character is used as the field separator in the current text file.
- **$3** – This refers to the value in the third column of the current line. In the applicationX_in_epoch.log, the third column contains the timestamp in epoch to be converted.
- **strftime** – This is an AWK internal function designed to work with time. The %c and $3 in between parenthesis are the parameters passed to strftime.
- **applicationX**_in_epoch.log – This is the input text file to be loaded and used.

Because you are already in the lab.support.files directory, you do not need to add path information, /home/analyst/lab.support.files/applicationX_in_epoch.log.

Step 4: Use nano (or your favorite text editor) to remove the extra empty line at the end of the file
[analyst@secOps lab.support.files]$ nano applicationX_in_epoch.log

```
[analyst@secOps lab.support.files]$  nano applicationX_in_epoch.log
[analyst@secOps lab.support.files]$  cat applicationX_in_epoch.log
2|Z|1219071600|AF|0
3|N|1219158000|AF|89
4|N|1220799600|AS|12
1|Z|1220886000|AS|67
5|N|1220972400|EU|23
6|R|1221058800|OC|89
```

Step 5:While printing the result on the screen is useful for troubleshooting the script, analysts will likely need to save the output in a text file. Redirect the output of the script above to a file named applicationX_in_human.log to save it to a file:

[analyst@secOps lab.support.files]$ awk 'BEGIN {FS=OFS="|"} {$3=strftime("%c",$3)} {print}' applicationX_in_epoch.log > applicationX_in_human.log

Use cat to view the **applicationX_in_human.log.** Notice that the extra line is now removed and the timestamps for the log entries have been converted to human readable format.

[analyst@secOps lab.support.files]$ cat applicationX_in_human.log

```
[analyst@secOps lab.support.files]$ awk 'BEGIN {FS=OFS="|"} {$3=strftime("%c",$3
)} {print}' applicationX_in_epoch.log > applicationX_in_human.log
[analyst@secOps lab.support.files]$ cat applicationX_in_human.log
2|Z|Mon 18 Aug 2008 11:00:00 AM EDT|AF|0
3|N|Tue 19 Aug 2008 11:00:00 AM EDT|AF|89
4|N|Sun 07 Sep 2008 11:00:00 AM EDT|AS|12
1|Z|Mon 08 Sep 2008 11:00:00 AM EDT|AS|67
5|N|Tue 09 Sep 2008 11:00:00 AM EDT|EU|23
6|R|Wed 10 Sep 2008 11:00:00 AM EDT|OC|89
```

Part 2: Normalize Timestamps in an Apache Log File

Similar to what was done with the applicationX_in_epoch.log file, Apache web server log files can also be normalized.

Step 1: Open the terminal and type cat apache_in_epoch.log.

[analyst@secOps lab.support.files]$ cat apache_in_epoch.log

```
[analyst@secOps ~]$ cd /home/analyst/lab.support.files/
[analyst@secOps lab.support.files]$ ls -l
total 592
-rw-r--r-- 1 analyst analyst    649 Mar 21  2018 apache_in_epoch.log
-rw-r--r-- 1 analyst analyst    125 Jun 25 03:32 applicationX_in_epoch.log
-rw-r--r-- 1 analyst analyst    251 Jun 25 03:46 applicationX_in_human.log
drwxr-xr-x 4 analyst analyst   4096 Mar 21  2018 attack_scripts
-rw-r--r-- 1 analyst analyst    102 Mar 21  2018 confidential.txt
-rw-r--r-- 1 analyst analyst   2871 Mar 21  2018 cyops.mn
-rw-r--r-- 1 analyst analyst    255 Jun 25 00:34 decrypted_letter.txt
-rw-r--r-- 1 analyst analyst     75 Mar 21  2018 elk_services
-rw-r--r-- 1 analyst analyst    373 Mar 21  2018 h2_dropbear.banner
drwxr-xr-x 2 analyst analyst   4096 Apr  2  2018 instructor
```

```
[analyst@secOps lab.support.files]$ cat apache_in_epoch.log
198.51.100.213 - - [1219071600] "GET /twiki/bin/edit/Main/Double_bounce_sender?t
opicparent=Main.ConfigurationVariables HTTP/1.1" 401 12846
198.51.100.213 - - [1219158000] "GET /twiki/bin/rdiff/TWiki/NewUserTemplate?rev1
=1.3&rev2=1.2 HTTP/1.1" 200 4523
198.51.100.213 - - [1220799600] "GET /mailman/listinfo/hsdivision HTTP/1.1" 200
6291
198.51.100.213 - - [1220886000] "GET /twiki/bin/view/TWiki/WikiSyntax HTTP/1.1"
200 7352
198.51.100.213 - - [1220972400] "GET /twiki/bin/view/Main/DCCAndPostFix HTTP/1.1
" 200 5253
198.51.100.213 - - [1221058800] "GET /twiki/bin/oops/TWiki/AppendixFileSystem?te
mplate=oopsmore&m1=1.12&m2=1.12 HTTP/1.1" 200 11382
```

Step 2: In the CyberOps Workstation VM terminal, a copy of the Apache log file, apache_in_epoch.log, is stored in the /home/analyst/lab.support.files.

Step 3: type this command in the terminal to see the log in human readable.
[analyst@secOps lab.support.files]$ awk 'BEGIN {FS=OFS=" "} {$4=strftime("%c",$4)} {print}' apache_in_epoch.log

```
[analyst@secOps lab.support.files]$ awk 'BEGIN {FS=OFS=" "} {$4=strftime("%c",$4
)} {print}' apache_in_epoch.log
198.51.100.213 - - Wed 31 Dec 1969 07:00:00 PM EST "GET /twiki/bin/edit/Main/Dou
ble_bounce_sender?topicparent=Main.ConfigurationVariables HTTP/1.1" 401 12846
198.51.100.213 - - Wed 31 Dec 1969 07:00:00 PM EST "GET /twiki/bin/rdiff/TWiki/N
ewUserTemplate?rev1=1.3&rev2=1.2 HTTP/1.1" 200 4523
198.51.100.213 - - Wed 31 Dec 1969 07:00:00 PM EST "GET /mailman/listinfo/hsdivi
sion HTTP/1.1" 200 6291
198.51.100.213 - - Wed 31 Dec 1969 07:00:00 PM EST "GET /twiki/bin/view/TWiki/Wi
kiSyntax HTTP/1.1" 200 7352
198.51.100.213 - - Wed 31 Dec 1969 07:00:00 PM EST "GET /twiki/bin/view/Main/DCC
AndPostFix HTTP/1.1" 200 5253
198.51.100.213 - - Wed 31 Dec 1969 07:00:00 PM EST "GET /twiki/bin/oops/TWiki/Ap
pendixFileSystem?template=oopsmore&m1=1.12&m2=1.12 HTTP/1.1" 200 11382
```

Step 4: Before moving forward, think about the output of the script.
Can you guess what caused the incorrect output? Is the script incorrect? What are the relevant differences between the **applicationX_in_epoch.log and apache_in_epoch.log?**
The problem is the square brackets in the course file. The script expects the timestamp to be in the Unix Epoch format which does not include the square brackets. Because the script does not know what number represents the "[" character, it assumes zero and returns the Unix beginning of time in UTC -5.

Step 5: To fix the problem, the square brackets must be removed from the timestamp field before the conversion takes place. Adjust the script by adding two actions before the conversion. As shown,
[analyst@secOps lab.support.files]$ awk 'BEGIN {FS=OFS=" "} {gsub(/[|]/,"",$4)}{print}{$4=strftime("%c",$4)}{print}' apache_in_epoch.log

```
[analyst@secOps lab.support.files]$ awk 'BEGIN {FS=OFS=" "} {gsub(/[|]/,"",$4)}{
print}{$4=strftime("%c",$4)}{print}' apache_in_epoch.log
198.51.100.213 - - [1219071600] "GET /twiki/bin/edit/Main/Double_bounce_sender?t
opicparent=Main.ConfigurationVariables HTTP/1.1" 401 12846
198.51.100.213 - - Wed 31 Dec 1969 07:00:00 PM EST "GET /twiki/bin/edit/Main/Dou
ble_bounce_sender?topicparent=Main.ConfigurationVariables HTTP/1.1" 401 12846
198.51.100.213 - - [1219158000] "GET /twiki/bin/rdiff/TWiki/NewUserTemplate?rev1
=1.3&rev2=1.2 HTTP/1.1" 200 4523
198.51.100.213 - - Wed 31 Dec 1969 07:00:00 PM EST "GET /twiki/bin/rdiff/TWiki/N
ewUserTemplate?rev1=1.3&rev2=1.2 HTTP/1.1" 200 4523
198.51.100.213 - - [1220799600] "GET /mailman/listinfo/hsdivision HTTP/1.1" 200
6291
198.51.100.213 - - Wed 31 Dec 1969 07:00:00 PM EST "GET /mailman/listinfo/hsdivi
sion HTTP/1.1" 200 6291
198.51.100.213 - - [1220886000] "GET /twiki/bin/view/TWiki/WikiSyntax HTTP/1.1"
200 7352
198.51.100.213 - - Wed 31 Dec 1969 07:00:00 PM EST "GET /twiki/bin/view/TWiki/Wi
kiSyntax HTTP/1.1" 200 7352
198.51.100.213 - - [1220972400] "GET /twiki/bin/view/Main/DCCAndPostFix HTTP/1.1
" 200 5253
198.51.100.213 - - Wed 31 Dec 1969 07:00:00 PM EST "GET /twiki/bin/view/Main/DCC
AndPostFix HTTP/1.1" 200 5253
198.51.100.213 - - [1221058800] "GET /twiki/bin/oops/TWiki/AppendixFileSystem?te
mplate=oopsmore&m1=1.12&m2=1.12 HTTP/1.1" 200 11382
```