הדגמת תקשורת SSL

בשלב ראשון נרצה ליצור מפתח פרטי ותעודה חתומה עצמית (self-signed certificate) המכילה את המפתח הציבורי.

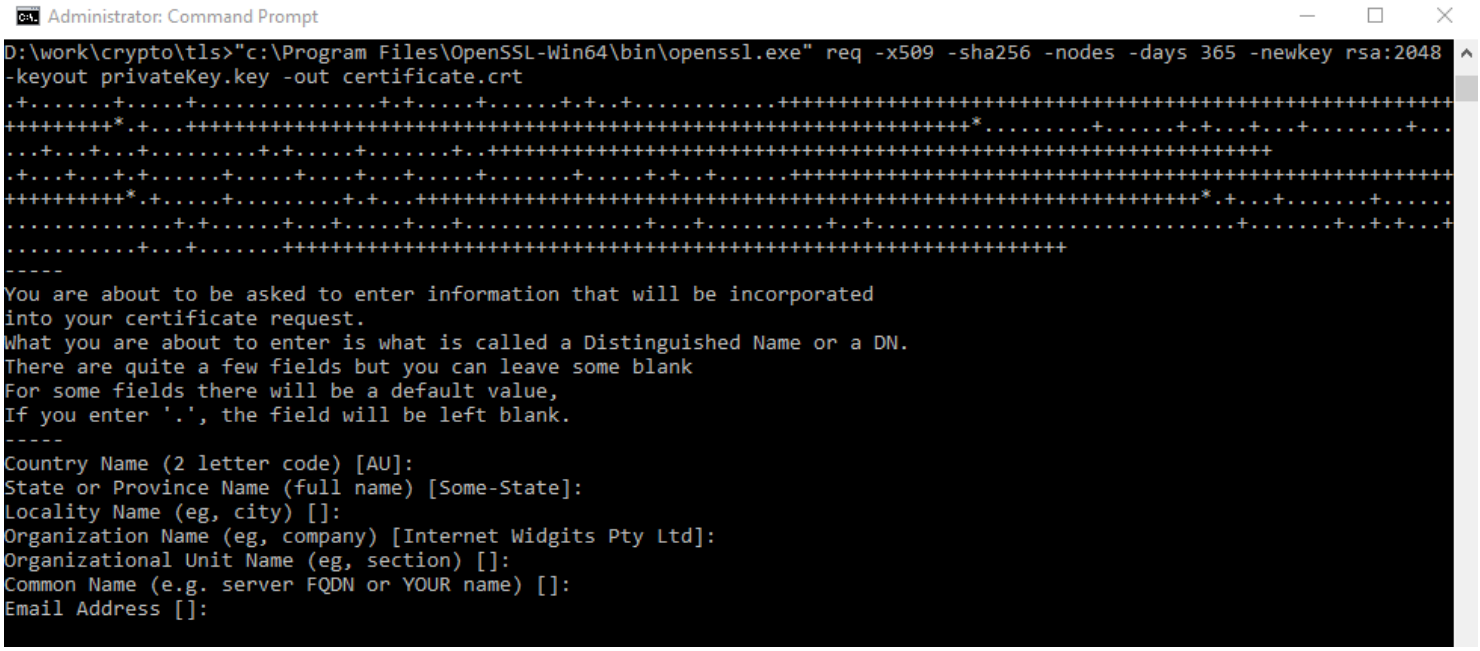
כדי ליצור את המפתח והתעודה נוריד את התוכנה openssl. בד"כ התוכנה מגיעה כקבציי קוד ויש ליצור אותה על המחשב, אך יש מספר גופים אשר יצרו עבורנו קבצי התקנה. ניתן למצוא את הקישורים ב- https://wiki.openssl.org/index.php/Binaries וניתן להוריד את ההתקנה מהאתר https://slproweb.com/products/Win32OpenSSL.html.

לאחר שהתקנו את התוכנה, נפתח cmd במצב אדמיניסטרטור וניגש לספריית פרויקט הפייתון שלנו.

נריץ את openssl עם הפרמטרים הבאים:

```
req -x509 -sha256 -nodes -days 365 -newkey rsa:2048 -keyout privateKey.key -out certificate.crt
```

שימו לב שיש לתת את הנתיב המלא להתקנה של openssl במחשב שלכם (בדוגמה הנתיב הוא c:\Program Files\OpenSSL-Win64\bin\openssl.exe)



קוד השרת

נבצע יבוא של ספריות התקשורת וההצפנה

```
import socket
import ssl
```

נייצר עצם שידאג לאבטחה, נטען את קובץ התעודה ואת המפתח הפרטי:

```
    context = ssl.SSLContext(ssl.PROTOCOL_TLS_SERVER)
    context.load_cert_chain(CERT_FILE, KEY_FILE)
```

נייצר socket רגיל ונקשר אותו לפורט של מערכת ההפעלה.

```
    server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    try:
        server_socket.bind((IP_ADDR, PORT))
        server_socket.listen(QUEUE_LEN)
```

נעטוף את ה-socket בעצם האבטחה ונקבל socket מאובטח. נמתין להתקשרות מהלקוח בעזרת ה-socket המאובטח.

```
        ssock = context.wrap_socket(server_socket, server_side=True)
        conn, addr = ssock.accept()
```

מכאן הקוד זהה לקוד של שרת לא מאובטח.

```
            msg = conn.recv(PACKET_LEN).decode()
            while msg != EXIT_CMD:
                print('received ' + msg)
                conn.send(MSG.encode())
                msg = conn.recv(PACKET_LEN).decode()
            conn.send(EXIT_RES.encode())
            print('exiting')
```

קוד השרת המלא:

```python
"""
Author: Nir Dweck
Date: 25/10/21
Description: a SSL server
"""
import socket
import ssl

IP_ADDR = '0.0.0.0'
PORT = 8443
QUEUE_LEN = 1
PACKET_LEN = 1024
CERT_FILE = 'certificate.crt'
KEY_FILE = 'privateKey.key'
MSG = 'have a nice day'
EXIT_CMD = 'exit'
EXIT_RES = 'by by'


def main():
    """
    listens for a single client connection.
    receives commands from the client and answers with a single response.
    once receives the 'exit' command, responses with 'by by' and exit's
    :return: None
    """
    context = ssl.SSLContext(ssl.PROTOCOL_TLS_SERVER)
    context.load_cert_chain(CERT_FILE, KEY_FILE)
    server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    try:
        server_socket.bind((IP_ADDR, PORT))
        server_socket.listen(QUEUE_LEN)
        ssock = context.wrap_socket(server_socket, server_side=True)
        conn, addr = ssock.accept()
        try:
            msg = conn.recv(PACKET_LEN).decode()
            while msg != EXIT_CMD:
                print('received ' + msg)
                conn.send(MSG.encode())
                msg = conn.recv(PACKET_LEN).decode()
            conn.send(EXIT_RES.encode())
            print('exiting')
        except socket.error as sock_err:
            print(sock_err)
        finally:
            conn.close()
    except socket.error as sock_err:
        print(sock_err)
    finally:
        server_socket.close()


if __name__ == '__main__':
    main()
```

קוד הלקוח

נבצע יבוא של ספריות התקשורת וההצפנה

```
import socket
import ssl
```

נייצר עצם שידאג לאבטחה, מאחר ואנו משתמשים בתעודה חתומה עצמית (self-signed certificate) נורא ללקוח לא לוודא את התעודה:

```
# create the ssl context
context = ssl.create_default_context()
# allow self signed certificate
context.check_hostname = False
context.verify_mode = ssl.CERT_NONE
```

נפתח socket, נעטוף אותו ע"י עצם האבטחה ונקבל socket מאובטח:

```
my_socket = socket.socket()
conn = context.wrap_socket(my_socket, server_hostname=HOST_NAME)
```

מכאן הקוד זהה לקוד של לקוח לא מאובטח:

```
    conn.connect((HOST_NAME, PORT))
    msg = input(USER_INPUT)
    while True:
        conn.send(msg.encode())
        answer = conn.read(MSG_LEN).decode()
        print(answer)
        if answer == EXIT_CMD:
            break
        msg = input(USER_INPUT)
    print('exiting')
```

קוד הלקוח המלא:

```python
"""
Author: Nir Dweck
Date: 25/10/21
Description: a SSL client
"""
import socket
import ssl

HOST_NAME = '127.0.0.1'
PORT = 8443
MSG_LEN = 1024
EXIT_CMD = 'by by'
USER_INPUT = 'please enter a command'


def main():
    """
    creates a secure connection with the server, receives commands, sends them ot the server and receives a response
    from the server.
    exits when receives the 'by by' response
    :return: None
    """
    # create the ssl context
    context = ssl.create_default_context()
    # allow self signed certificate
    context.check_hostname = False
    context.verify_mode = ssl.CERT_NONE
    my_socket = socket.socket()
    conn = context.wrap_socket(my_socket, server_hostname=HOST_NAME)
    try:
        conn.connect((HOST_NAME, PORT))
        msg = input(USER_INPUT)
        while True:
            conn.send(msg.encode())
            answer = conn.read(MSG_LEN).decode()
            print(answer)
            if answer == EXIT_CMD:
                break
            msg = input(USER_INPUT)
        print('exiting')
    except socket.error as sock_err:
        print(sock_err)
    finally:
        conn.close()


if __name__ == '__main__':
    main()
```

כתבו שרת ולקוח בסיסיים המשתמשים בתקשורת מאובטחת.

פתחו wireshark ועקבו אחרי פתיחת הקשר עד אשר נקבע מפתח ההצפנה הסימטרי.