**Харківський національний економічний університет**

**імені Семена Кузнеця**

**ЗВІТ**

**З ВИКОНАННЯ ЛАБОРАТОРНОЇ РОБОТИ № 8**

**за дисципліною: *“*Технології програмування**”

**на тему: “Робота з файлами”**

**Варіант: 4**

**Виконав: студент факультету Інформаційних технологій**

**2 курсу, спец. Кібербезпека,**

**групи 6.04.125.010.21.2**

**Бойко Вадим Віталійович**

**Перевірив:**

**Міхєєв Іван Андрійович**

**ХНЕУ ім. С. Кузнеця**

**2022**

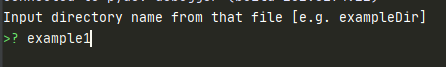
Визначити список IP адрес, з яких відправлялися запити до серверу, та кількість статус кодів серверу з категорії «4хх»

Код

*import* re  
  
q = open('example1/access.log', 'r')  
  
ipRegex = '[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}'  
errorStatusRegex = ' 4[0-9]{2} '  
  
ipList = []  
  
c = 0  
  
*for* line *in* q:  
 ip = re.findall(ipRegex, line)  
 errorStatusExists = re.findall(errorStatusRegex, line)  
 ipList.append(ip[0])  
 *if* (len(errorStatusExists)):  
 c += 1  
  
print('Ip addresses: ', ipList)  
print(f'Count of status code 404 = {c}')  
q.close()

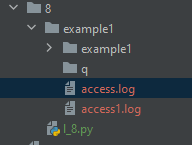
Додаткові завдання

Передбачити можливість аналізу декількох файлів із логами роботи веб-серверу, що розміщені у вказаній директорії.





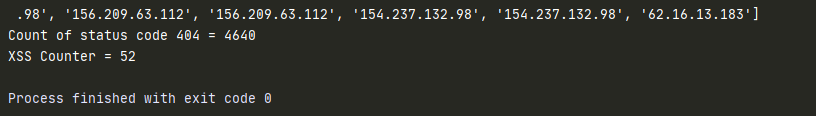
Структура файлів



Код

*import* re  
*import* os  
  
ipRegex = '[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}'  
errorStatusRegex = ' 4[0-9]{2} '  
  
directoryName = input('Input directory name from that file [e.g. exampleDir] ')  
  
files = os.listdir(directoryName)  
  
files = [f *for* f *in* files *if* os.path.isfile(directoryName + '/' + f)]  
  
  
*class* Info:  
 *def \_\_init\_\_*(self, *ipList* = [], *errorCounter*=0):  
 self.\_ipList = *ipList* self.\_errorCounter = *errorCounter  
  
 def* addIp(self, *ip*):  
 self.\_ipList.append(*ip*)  
  
 *def* addError(self):  
 self.\_errorCounter += 1  
  
 *def* getIpList(self):  
 *return* [] + self.\_ipList  
  
 *def* getErrorCount(self):  
 *return* self.\_errorCounter  
  
  
info = Info()  
  
  
*def* hasErrorCode(*line*=''):  
 errorStatusExists = re.findall(errorStatusRegex, *line*)  
 *return* len(errorStatusExists)  
  
  
*def* getIpFromLine(*line*=''):  
 *return* re.findall(ipRegex, *line*)  
  
  
*def* addDataFromfile(*file*):  
 *for* line *in file*:  
 ip = getIpFromLine(line)  
 *if* (len(ip)):  
 info.addIp(ip[0])  
 *if* (hasErrorCode(line)):  
 info.addError()  
  
  
*for* file *in* files:  
 *with* open(f'{directoryName}/{file}', 'r') *as* currentFile:  
 addDataFromfile(currentFile)  
  
print('Ip addresses: ', info.getIpList())  
print(f'Count of status code 404 = {info.getErrorCount()}')

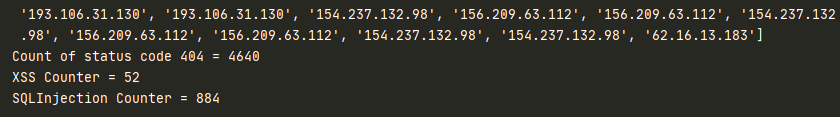
Дослідити наявність слідів XSS атак на сервер.



Код

*import* re  
*import* os  
  
ipRegex = '[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}'  
errorStatusRegex = ' 4[0-9]{2} '  
  
directoryName = input('Input directory name from that file [e.g. exampleDir] ')  
  
files = os.listdir(directoryName)  
  
files = [f *for* f *in* files *if* os.path.isfile(directoryName + '/' + f)]  
  
  
*class* Info:  
 *def \_\_init\_\_*(self, *ipList* = [], *errorCounter* = 0, *xssCounter* = 0):  
 self.\_ipList = *ipList* self.\_errorCounter = *errorCounter* self.\_xssCounter = *xssCounter  
  
 def* addIp(self, *ip*):  
 self.\_ipList.append(*ip*)  
  
 *def* addError(self):  
 self.\_errorCounter += 1  
  
 *def* getIpList(self):  
 *return* [] + self.\_ipList  
  
 *def* getErrorCount(self):  
 *return* self.\_errorCounter  
  
 *def* addXSSInfo(self, *line*):  
 xssPattern = ['%3C', '<img', '<a href', '<body', '<script', '<b', '<h', '<marquee']  
 *for* regex *in* xssPattern:  
 xssMatch = re.findall(regex, *line*)  
 self.\_xssCounter += len(xssMatch)  
  
 *def* getXSSCounter(self):  
 *return* self.\_xssCounter  
  
  
  
  
info = Info()  
  
  
*def* hasErrorCode(*line*=''):  
 errorStatusExists = re.findall(errorStatusRegex, *line*)  
 *return* len(errorStatusExists)  
  
  
*def* getIpFromLine(*line*=''):  
 *return* re.findall(ipRegex, *line*)  
  
  
*def* addDataFromfile(*file*):  
 *for* line *in file*:  
 ip = getIpFromLine(line)  
 *if* (len(ip)):  
 info.addIp(ip[0])  
 *if* (hasErrorCode(line)):  
 info.addError()  
 info.addXSSInfo(line)  
  
  
  
*for* file *in* files:  
 *with* open(f'{directoryName}/{file}', 'r') *as* currentFile:  
 addDataFromfile(currentFile)  
  
print('Ip addresses: ', info.getIpList())  
print(f'Count of status code 404 = {info.getErrorCount()}')  
print(f'XSS Counter = {info.getXSSCounter()}')

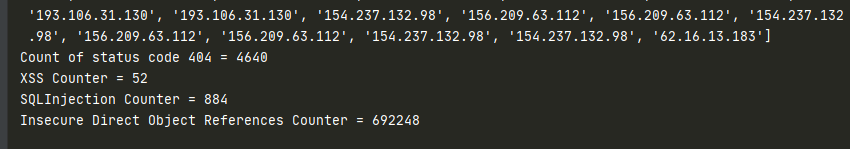
Дослідити наявність слідів атак на сервер за допомогою SQL ін’єкцій



Код

*import* re  
*import* os  
  
ipRegex = '[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}'  
errorStatusRegex = ' 4[0-9]{2} '  
  
# directoryName = input('Input directory name from that file [e.g. exampleDir] ')  
directoryName = 'example1'  
  
files = os.listdir(directoryName)  
  
files = [f *for* f *in* files *if* os.path.isfile(directoryName + '/' + f)]  
  
*class* Info:  
 *def \_\_init\_\_*(self, *ipList* = [], *errorCounter* = 0, *xssCounter* = 0, *sqlInjectionCounter* = 0):  
 self.\_ipList = *ipList* self.\_errorCounter = *errorCounter* self.\_xssCounter = *xssCounter* self.\_sqlInjectionCounter = *sqlInjectionCounter  
  
 def* addIp(self, *ip*):  
 self.\_ipList.append(*ip*)  
  
 *def* addError(self):  
 self.\_errorCounter += 1  
  
 *def* getIpList(self):  
 *return* [] + self.\_ipList  
  
 *def* getErrorCount(self):  
 *return* self.\_errorCounter  
  
 *def* addXSSInfo(self, *line*):  
 xssPattern = ['%3C', '<img', '<a href', '<body', '<script', '<b', '<h', '<marquee']  
 *for* regex *in* xssPattern:  
 xssMatch = re.findall(regex, *line*)  
 self.\_xssCounter += len(xssMatch)  
  
 *def* getXSSCounter(self):  
 *return* self.\_xssCounter  
  
 *def* addSQLInjectionInfo(self, *line*):  
 sqlInjectionPattern = ['%27', '--', '%3B', 'exec', 'union+', 'union\*', 'system\(', 'eval\(', 'group\_concat', 'column\_name', 'order by', 'insert into', '@version',]  
  
 *for* regex *in* sqlInjectionPattern:  
 sqlMatch = re.findall(regex, *line*)  
 self.\_sqlInjectionCounter += len(sqlMatch)  
  
 *def* getSQLInjectionCounter(self):  
 *return* self.\_sqlInjectionCounter  
  
info = Info()  
  
*def* hasErrorCode(*line*=''):  
 errorStatusExists = re.findall(errorStatusRegex, *line*)  
 *return* len(errorStatusExists)  
  
  
*def* getIpFromLine(*line*=''):  
 *return* re.findall(ipRegex, *line*)  
  
  
*def* addDataFromfile(*file*):  
 *for* line *in file*:  
 ip = getIpFromLine(line)  
 *if* (len(ip)):  
 info.addIp(ip[0])  
 *if* (hasErrorCode(line)):  
 info.addError()  
 info.addXSSInfo(line)  
 info.addSQLInjectionInfo(line)  
  
*for* file *in* files:  
 *with* open(f'{directoryName}/{file}', 'r') *as* currentFile:  
 print(1)  
 addDataFromfile(currentFile)  
  
print('Ip addresses: ', info.getIpList())  
print(f'Count of status code 404 = {info.getErrorCount()}')  
print(f'XSS Counter = {info.getXSSCounter()}')  
print(f'SQLInjection Counter = {info.getSQLInjectionCounter()}')

Дослідити наявність слідів Insecure Direct Object References атак на сервер



Код

*import* re  
*import* os  
  
ipRegex = '[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}'  
errorStatusRegex = ' 4[0-9]{2} '  
  
# directoryName = input('Input directory name from that file [e.g. exampleDir] ')  
directoryName = 'example1'  
  
files = os.listdir(directoryName)  
  
files = [f *for* f *in* files *if* os.path.isfile(directoryName + '/' + f)]  
  
*class* Info:  
 *def \_\_init\_\_*(self, *ipList* = [], *errorCounter* = 0, *xssCounter* = 0, *sqlInjectionCounter* = 0, *insecureDirectObjectReferencesCounter* = 0):  
 self.\_ipList = *ipList* self.\_errorCounter = *errorCounter* self.\_xssCounter = *xssCounter* self.\_sqlInjectionCounter = *sqlInjectionCounter* self.\_insecureDirectObjectReferencesCounter = *insecureDirectObjectReferencesCounter  
  
 def* addIp(self, *ip*):  
 self.\_ipList.append(*ip*)  
  
 *def* addError(self):  
 self.\_errorCounter += 1  
  
 *def* getIpList(self):  
 *return* [] + self.\_ipList  
  
 *def* getErrorCount(self):  
 *return* self.\_errorCounter  
  
 *def* addXSSInfo(self, *line*):  
 xssPattern = ['%3C', '<img', '<a href', '<body', '<script', '<b', '<h', '<marquee']  
 *for* regex *in* xssPattern:  
 xssMatch = re.findall(regex, *line*)  
 self.\_xssCounter += len(xssMatch)  
  
 *def* getXSSCounter(self):  
 *return* self.\_xssCounter  
  
 *def* addSQLInjectionInfo(self, *line*):  
 sqlInjectionPattern = ['%27', '--', '%3B', 'exec', 'union+', 'union\*', 'system\(', 'eval\(', 'group\_concat', 'column\_name', 'order by', 'insert into', '@version',]  
  
 *for* regex *in* sqlInjectionPattern:  
 sqlMatch = re.findall(regex, *line*)  
 self.\_sqlInjectionCounter += len(sqlMatch)  
  
 *def* getSQLInjectionCounter(self):  
 *return* self.\_sqlInjectionCounter  
  
 *def* addIDORInfo(self, *line*):  
 idorPattern = ['../', '%2e%2f', '%2e%2e/', '.%2f', '..%c1%9', '..%c0%af', '/usr/',  
 '/passwd', '/grub', 'boot.ini', '/conf/', '/etc/', '/proc/', '/opt/',  
 '/sbin/', '/dev/', '/tmp/', '/kern/', '/root/', '/sys/', '/system/',  
 '/windows/', '/winnt/', '/inetpub/', '/localstart/', '/boot/']  
 *for* regex *in* idorPattern:  
 idorMatch = re.findall(regex, *line*)  
 self.\_insecureDirectObjectReferencesCounter += len(idorMatch)  
  
  
 *def* getIDORCounter(self):  
 *return* self.\_insecureDirectObjectReferencesCounter  
  
info = Info()  
  
*def* hasErrorCode(*line*=''):  
 errorStatusExists = re.findall(errorStatusRegex, *line*)  
 *return* len(errorStatusExists)  
  
  
*def* getIpFromLine(*line*=''):  
 *return* re.findall(ipRegex, *line*)  
  
  
*def* addDataFromfile(*file*):  
 *for* line *in file*:  
 ip = getIpFromLine(line)  
 *if* (len(ip)):  
 info.addIp(ip[0])  
 *if* (hasErrorCode(line)):  
 info.addError()  
 info.addXSSInfo(line)  
 info.addSQLInjectionInfo(line)  
 info.addIDORInfo(line)  
  
*for* file *in* files:  
 *with* open(f'{directoryName}/{file}', 'r') *as* currentFile:  
 addDataFromfile(currentFile)  
  
print('Ip addresses: ', info.getIpList())  
print(f'Count of status code 404 = {info.getErrorCount()}')  
print(f'XSS Counter = {info.getXSSCounter()}')  
print(f'SQLInjection Counter = {info.getSQLInjectionCounter()}')  
print(f'Insecure Direct Object References Counter = {info.getIDORCounter()}')

Побудувати стовбчасту діаграму для топ 10 результатів аналізу



Код

*import* re  
*import* os  
*import* sqlite3  
  
ipRegex = '[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}'  
errorStatusRegex = ' 4[0-9]{2} '  
  
# directoryName = input('Input directory name from that file [e.g. exampleDir] ')  
directoryName = 'example1'  
  
files = os.listdir(directoryName)  
  
files = [f *for* f *in* files *if* os.path.isfile(directoryName + '/' + f)]  
  
*class* Info:  
 \_con = sqlite3.connect('request.db')  
 # \_con = sqlite3.connect('request.db').cursor()  
  
 *def \_\_init\_\_*(self, *ipList* = [], *errorCounter* = 0, *xssCounter* = 0, *sqlInjectionCounter* = 0, *insecureDirectObjectReferencesCounter* = 0):  
 self.\_ipList = *ipList* self.\_errorCounter = *errorCounter* self.\_xssCounter = *xssCounter* self.\_sqlInjectionCounter = *sqlInjectionCounter* self.\_insecureDirectObjectReferencesCounter = *insecureDirectObjectReferencesCounter* self.\_ipStat = {}  
 self.\_requestCount = 0  
 self.\_top10Requests = []  
 self.\_cursor = self.\_con.cursor()  
 self.initDB()  
  
 *def* initDB(self):  
 self.\_cursor.execute("CREATE TABLE IF NOT EXISTS requests(ip, count);")  
 # self.\_cursor.commit()  
  
 *def* closeDB(self):  
 self.\_con.commit()  
 self.\_con.close()  
  
 *def* getTop10Request(self):  
 ipList = []  
 *for* row *in* self.\_cursor.execute('SELECT \* FROM requests ORDER BY count DESC LIMIT 10;'):  
 ipList.append(row)  
 *return* ipList  
  
  
 *def* addIp(self, *ip*):  
 self.\_requestCount += 1  
 data = self.\_cursor.execute(f"SELECT \* FROM requests WHERE ip = '{*ip*}';").fetchone()  
 *if* data == *None*:  
 self.\_cursor.execute(f"INSERT INTO requests VALUES('{*ip*}', 1);")  
 *return* self.\_con.commit()  
 *else*:  
 self.\_cursor.execute(f"update requests SET count = count + 1 WHERE ip = '{*ip*}';")  
 *return* self.\_con.commit()  
  
 *def* addError(self):  
 self.\_errorCounter += 1  
  
 *def* getIpList(self):  
 ipList = []  
 *for* row *in* self.\_cursor.execute('SELECT ip FROM requests;'):  
 ipList.append(row[0])  
 *return* ipList  
  
 *def* getErrorCount(self):  
 *return* self.\_errorCounter  
  
 *def* addXSSInfo(self, *line*):  
 xssPattern = ['%3C', '<img', '<a href', '<body', '<script', '<b', '<h', '<marquee']  
 *for* regex *in* xssPattern:  
 xssMatch = re.findall(regex, *line*)  
 self.\_xssCounter += len(xssMatch)  
  
 *def* getXSSCounter(self):  
 *return* self.\_xssCounter  
  
 *def* addSQLInjectionInfo(self, *line*):  
 sqlInjectionPattern = ['%27', '--', '%3B', 'exec', 'union+', 'union\*', 'system\(', 'eval\(', 'group\_concat', 'column\_name', 'order by', 'insert into', '@version',]  
  
 *for* regex *in* sqlInjectionPattern:  
 sqlMatch = re.findall(regex, *line*)  
 self.\_sqlInjectionCounter += len(sqlMatch)  
  
 *def* getSQLInjectionCounter(self):  
 *return* self.\_sqlInjectionCounter  
  
 *def* addIDORInfo(self, *line*):  
 idorPattern = ['../', '%2e%2f', '%2e%2e/', '.%2f', '..%c1%9', '..%c0%af', '/usr/',  
 '/passwd', '/grub', 'boot.ini', '/conf/', '/etc/', '/proc/', '/opt/',  
 '/sbin/', '/dev/', '/tmp/', '/kern/', '/root/', '/sys/', '/system/',  
 '/windows/', '/winnt/', '/inetpub/', '/localstart/', '/boot/']  
 *for* regex *in* idorPattern:  
 idorMatch = re.findall(regex, *line*)  
 self.\_insecureDirectObjectReferencesCounter += len(idorMatch)  
  
  
 *def* getIDORCounter(self):  
 *return* self.\_insecureDirectObjectReferencesCounter  
  
 *def* printStat(self):  
 print('Ip addresses: ', self.getIpList())  
 print(f'Count of status code 404 = {self.\_errorCounter}')  
 print(f'XSS Counter = {self.\_xssCounter}')  
 print(f'SQLInjection Counter = {self.\_xssCounter}')  
 print(f'Insecure Direct Object References Counter = {self.\_insecureDirectObjectReferencesCounter}')  
 rows = self.getTop10Request()  
 *for* row *in* rows:  
 print(row[0] + '\t-\t' + '\*' \* row[1])  
  
info = Info()  
  
*def* hasErrorCode(*line*=''):  
 errorStatusExists = re.findall(errorStatusRegex, *line*)  
 *return* len(errorStatusExists)  
  
  
*def* getIpFromLine(*line*=''):  
 *return* re.findall(ipRegex, *line*)  
  
  
*def* addDataFromfile(*file*):  
 *for* line *in file*:  
 ip = getIpFromLine(line)  
 *if* (len(ip)):  
 info.addIp(ip[0])  
 *if* (hasErrorCode(line)):  
 info.addError()  
 info.addXSSInfo(line)  
 info.addSQLInjectionInfo(line)  
 info.addIDORInfo(line)  
  
*for* file *in* files:  
 *with* open(f'{directoryName}/{file}', 'r') *as* currentFile:  
 addDataFromfile(currentFile)  
  
info.printStat()  
info.closeDB()

Розробити фреймворк для отримання та формування \*.csv звітів із частковою/повною інформацією щодо роботи веб-серверу за наявними логами.

Код

*import* re  
*import* os  
*import* sqlite3  
*import* csv  
  
ipRegex = '[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}'  
errorStatusRegex = ' 4[0-9]{2} '  
  
# directoryName = input('Input directory name from that file [e.g. exampleDir] ')  
directoryName = 'example1'  
  
files = os.listdir(directoryName)  
  
files = [f *for* f *in* files *if* os.path.isfile(directoryName + '/' + f)]  
  
*class* Info:  
 \_con = sqlite3.connect('request.db')  
 # \_con = sqlite3.connect('request.db').cursor()  
  
 *def \_\_init\_\_*(self, *ipList* = [], *errorCounter* = 0, *xssCounter* = 0, *sqlInjectionCounter* = 0, *insecureDirectObjectReferencesCounter* = 0):  
 self.\_ipList = *ipList* self.\_errorCounter = *errorCounter* self.\_xssCounter = *xssCounter* self.\_sqlInjectionCounter = *sqlInjectionCounter* self.\_insecureDirectObjectReferencesCounter = *insecureDirectObjectReferencesCounter* self.\_ipStat = {}  
 self.\_requestCount = 0  
 self.\_top10Requests = []  
 self.\_cursor = self.\_con.cursor()  
 self.initDB()  
  
 *def* initDB(self):  
 self.\_cursor.execute("CREATE TABLE IF NOT EXISTS requests(ip, count);")  
 # self.\_cursor.commit()  
  
 *def* closeDB(self):  
 self.\_con.commit()  
 self.\_con.close()  
  
 *def* getTop10Request(self):  
 ipList = []  
 *for* row *in* self.\_cursor.execute('SELECT \* FROM requests ORDER BY count DESC LIMIT 10;'):  
 ipList.append(row)  
 *return* ipList  
  
  
 *def* addIp(self, *ip*):  
 self.\_requestCount += 1  
 data = self.\_cursor.execute(f"SELECT \* FROM requests WHERE ip = '{*ip*}';").fetchone()  
 *if* data == *None*:  
 self.\_cursor.execute(f"INSERT INTO requests VALUES('{*ip*}', 1);")  
 *return* self.\_con.commit()  
 *else*:  
 self.\_cursor.execute(f"update requests SET count = count + 1 WHERE ip = '{*ip*}';")  
 *return* self.\_con.commit()  
  
 *def* addError(self):  
 self.\_errorCounter += 1  
  
 *def* getIpList(self):  
 ipList = []  
 *for* row *in* self.\_cursor.execute('SELECT ip FROM requests;'):  
 ipList.append(row[0])  
 *return* ipList  
  
 *def* getErrorCount(self):  
 *return* self.\_errorCounter  
  
 *def* addXSSInfo(self, *line*):  
 xssPattern = ['%3C', '<img', '<a href', '<body', '<script', '<b', '<h', '<marquee']  
 *for* regex *in* xssPattern:  
 xssMatch = re.findall(regex, *line*)  
 self.\_xssCounter += len(xssMatch)  
  
 *def* getXSSCounter(self):  
 *return* self.\_xssCounter  
  
 *def* addSQLInjectionInfo(self, *line*):  
 sqlInjectionPattern = ['%27', '--', '%3B', 'exec', 'union+', 'union\*', 'system\(', 'eval\(', 'group\_concat', 'column\_name', 'order by', 'insert into', '@version',]  
  
 *for* regex *in* sqlInjectionPattern:  
 sqlMatch = re.findall(regex, *line*)  
 self.\_sqlInjectionCounter += len(sqlMatch)  
  
 *def* getSQLInjectionCounter(self):  
 *return* self.\_sqlInjectionCounter  
  
 *def* addIDORInfo(self, *line*):  
 idorPattern = ['../', '%2e%2f', '%2e%2e/', '.%2f', '..%c1%9', '..%c0%af', '/usr/',  
 '/passwd', '/grub', 'boot.ini', '/conf/', '/etc/', '/proc/', '/opt/',  
 '/sbin/', '/dev/', '/tmp/', '/kern/', '/root/', '/sys/', '/system/',  
 '/windows/', '/winnt/', '/inetpub/', '/localstart/', '/boot/']  
 *for* regex *in* idorPattern:  
 idorMatch = re.findall(regex, *line*)  
 self.\_insecureDirectObjectReferencesCounter += len(idorMatch)  
  
  
 *def* getIDORCounter(self):  
 *return* self.\_insecureDirectObjectReferencesCounter  
  
 *def* printStat(self):  
 print('Ip addresses: ', self.getIpList())  
 print(f'Count of status code 404 = {self.\_errorCounter}')  
 print(f'XSS Counter = {self.\_xssCounter}')  
 print(f'SQLInjection Counter = {self.\_sqlInjectionCounter}')  
 print(f'Insecure Direct Object References Counter = {self.\_insecureDirectObjectReferencesCounter}')  
 rows = self.getTop10Request()  
 *for* row *in* rows:  
 print(row[0] + '\t-\t' + '\*' \* row[1])  
  
 *def* generateCSVFile(self):  
 *with* open('output.csv', 'w') *as* outputFile:  
 writer = csv.writer(outputFile)  
 header = ['Name', 'Count', 'Description']  
 writer.writerow(header)  
 writer.writerow(['404 status', self.\_errorCounter, 'Count of status code 404'])  
 writer.writerow(['XSS', self.\_xssCounter, 'Count of possible XSS'])  
 writer.writerow(['SQL Injection', self.\_sqlInjectionCounter, 'Count of possible SQL Injection'])  
 writer.writerow(['IDOR', self.\_insecureDirectObjectReferencesCounter, 'Count of possible Insecure Direct Object References'])  
 writer.writerow(['IP Address ', 'Request Counter for IP Address'])  
 *for* row *in* self.\_cursor.execute('SELECT \* FROM requests ORDER BY count DESC'):  
 writer.writerow([row[0], row[1]])  
  
info = Info()  
  
*def* hasErrorCode(*line*=''):  
 errorStatusExists = re.findall(errorStatusRegex, *line*)  
 *return* len(errorStatusExists)  
  
  
*def* getIpFromLine(*line*=''):  
 *return* re.findall(ipRegex, *line*)  
  
  
*def* addDataFromfile(*file*):  
 *for* line *in file*:  
 ip = getIpFromLine(line)  
 *if* (len(ip)):  
 info.addIp(ip[0])  
 *if* (hasErrorCode(line)):  
 info.addError()  
 info.addXSSInfo(line)  
 info.addSQLInjectionInfo(line)  
 info.addIDORInfo(line)  
  
*for* file *in* files:  
 *with* open(f'{directoryName}/{file}', 'r') *as* currentFile:  
 addDataFromfile(currentFile)  
  
info.generateCSVFile()  
info.closeDB()

Логи, які були згенеровані

