1. Министерство образования и науки Российской Федерации
2. Санкт-Петербургский Политехнический Университет Петра Великого
3. —
4. Институт кибербезопасности и защиты информации

**ЛАБОРАТОРНАЯ РАБОТА № 2**

**Основы проведения расследований инцидентов информационной безопасности**

по дисциплине «Методы и средства противодействия целенаправленным атакам»

1. Выполнила
2. студентка гр. 5151003/90801 Кулеева А.Г.

1. Руководитель
2. доцент, к.т.н. Жуковский Е.В.
3. Санкт-Петербург
4. 2023

# **Цель**

Изучение способов проведения криминалистического анализа (digital forensics), связанного с проведением расследований инцидентов информационной безопасности. Изучение способов обнаружения признаков компрометации системы и определения путей проникновения и распространения в системе атакующего.

# Формулировка задания

В ходе выполнения лабораторной работы необходимо выполнить следующие действия:

1. Получить у преподавателя дамп оперативной памяти, соответствующий варианту задания (таблица 1).
2. Изучить информацию, представленную в качестве сценария инцидента информационной безопасности (таблица 1).
3. Изучить основные команды и возможности средства анализа оперативной памяти volatility и информацию по принципам его использования.
4. Провести анализ полученного дампа памяти и ответить на указанные в таблице 1 вопросы.

В отчете привести описание выполненной для анализа последовательность действий с приведением снимков экрана или вывода используемых средств анализа.

# Ход работы

## Описание варианта

Был получен вариант с условным названием Team. Его сценарий заключается в следующем. Сотрудник сообщил, что его машина начала вести себя странно после получения подозрительного письма с файлом документа. Группа реагирования на инциденты захватила несколько дампов памяти с подозрительных машин для дальнейшей проверки. Как SOC-аналитик, проанализируйте дампы и помогите команде реагирования на инциденты (IR) выяснить, что произошло!

## ecorpoffice

### Предварительный анализ дампов

Используем утилиту [Volatility Framework 2.6.1](https://github.com/volatilityfoundation/volatility) — инс­тру­мент, реали­зован­ный на Python 2. Пред­назна­чен для извле­чения арте­фак­тов из образцов энер­гозави­симой памяти.

Преж­де чем искать вирус­ную активность, получим пер­вичную информа­цию о сис­теме, имя компь­юте­ра, сетевой адрес и вер­сию опе­раци­онной сис­темы (рисунок 1).

volatility\_2.6\_win64\_standalone.exe -f win7ecorpoffice2010-36b02ed3.vmem imageinfo

Вер­сия ОС — Win7SP1x64. Будем ее ука­зывать при поис­ке дру­гих арте­фак­тов в качес­тве про­филя в Volatility.

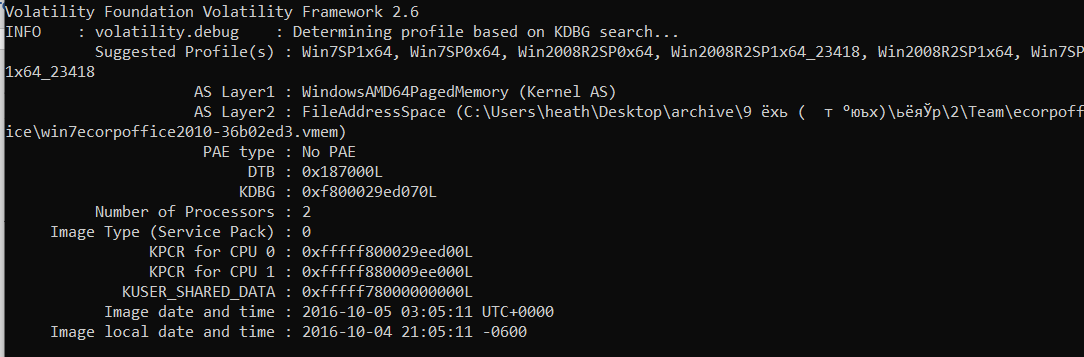


Рисунок 1 — Предварительные данные о системе

По­лучим имя компь­юте­ра, для это­го про­верим сле­дующий ключ реестра (рисунок 2):

SYSTEM\ControlSet001\Control\ComputerName\ComputerName

volatility\_2.6\_win64\_standalone.exe -f win7ecorpoffice2010-36b02ed3.vmem --profile=Win7SP1x64 printkey –K "ControlSet001\Control\ComputerName\ComputerName"

Имя компь­юте­ра: WIN-191HVE3KTLO.

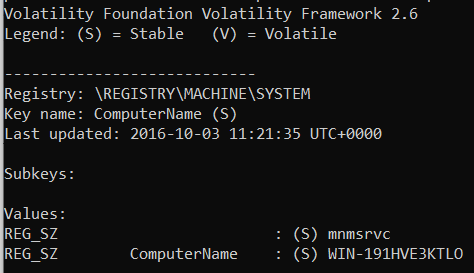


Рисунок 2 — Имя машины

### Под каким PID запущен вредоносный файл?

Про­ана­лизи­руем запущен­ные про­цес­сы в сис­теме (рисунок 3). Наиболее подозрительными являются процессы с под именами *SkypeC2AutoUpd* (1364), *OSPPSVC.EXE* (3532), *OUTLOOK.EXE* (*2692*).

volatility\_2.6\_win64\_standalone.exe -f win7ecorpoffice2010-36b02ed3.vmem --profile=Win7SP1x64 pstree

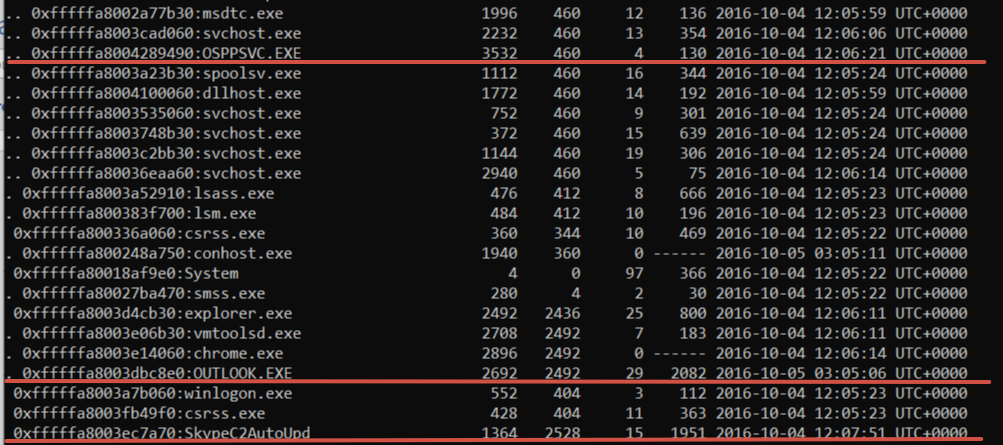


Рисунок 3 — Анализ процессов

### Каков IP-адрес сервера C2?

Используем плагин netscan для анализа сети (рисунок 4).

volatility\_2.6\_win64\_standalone.exe -f win7ecorpoffice2010-36b02ed3.vmem --profile=Win7SP1x64 netscan

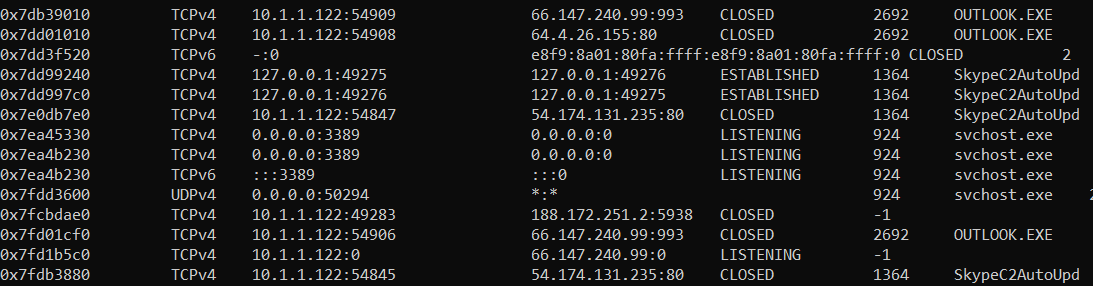
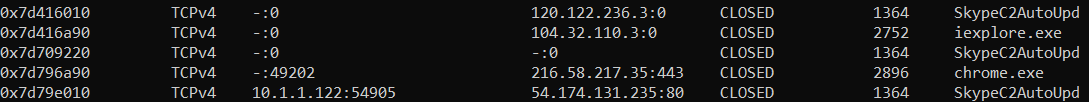


Рисунок 4 ― Результаты анализа

В таблице присутствует 7 записей об открытии сетевых сеансов для процесса SkypeC2AutoUpd и 2 записи для процесса OUTLOOK.EXE. Процесс SkypeC2AutoUpd обращается по сетевым адресам 120.122.236.3:0 и 54.174.131.235:80. Процесс OUTLOOK.EXE обращается по адресам 66.147.240.99:993 и 64.4.26.155:80.

### Какая версия Teamviewer используется вредоносным файлом?

Для исследуемых процессов был получен дамп оперативной памяти. Использовалась команда:

volatility\_2.6\_win64\_standalone.exe -f win7ecorpoffice2010-36b02ed3.vmem --profile=Win7SP1x64 memdump -p 1364 -D ./

Полученный дамп был открыт в Hex-редакторе, в котором с помощью поиска по строкам удалось идентифицировать используемую процессом версию TeamViewer 6.0 (рисунок 5).

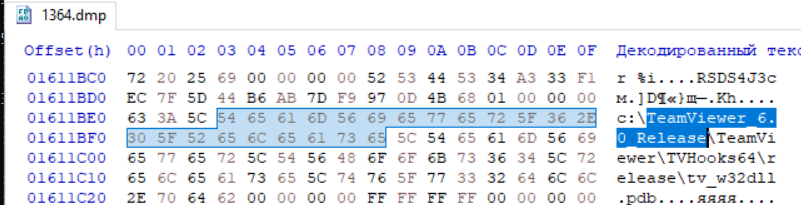


Рисунок 5 — Анализ дампа процесса

### Какой пароль использовал вредоносный файл для получения удаленного доступа к системе?

В утилите Volatility был запущен модуль editbox (рисунок 6). Он извлекает текст из элементов управления Windows Edit, из текстовых полей, созданных элементами управления Windows Common Controls. Удалось идентифицировать пароль, используемый вредоносным процессом для доступа в системе: P59fS93m. Для этого использовалась команда

volatility\_2.6\_win64\_standalone.exe -f win7ecorpoffice2010-36b02ed3.vmem --profile=Win7SP1x64 editbox

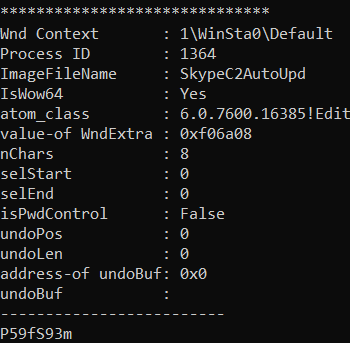


Рисунок 6 — Пароль, используемый вредоносным файлом

### Какой адрес электронной почты был у отправителя фишингового письма?

Для идентификации адреса электронной почты отправителя письма использовался модуль yarascan утилиты Volatility. Этот плагин используется для поиска последовательностей байтов, регулярных выражений, строк ANSI или Unicode в пользовательском режиме или памяти ядра. В результате удалось идентифицировать адрес электронной почты отправителя: [karenmiles@t-online.de](mailto:karenmiles@t-online.de) (рисунок 7). Для этого использовалась команда

volatility\_2.6\_win64\_standalone.exe -f win7ecorpoffice2010-36b02ed3.vmem --profile=Win7SP0x64 yarascan –Y “From:”

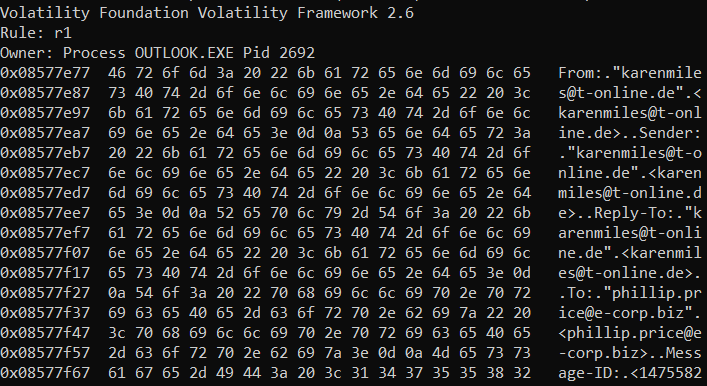


Рисунок 7 — Поиск фишингового адреса

### Каков MD5-хэш вредоносного документа?

Средствами модуля dumpfiles утилиты Volatility было получено содержимое всех текстовых файлов формата PST. Файлы формата .pst в ОС Windows используются программами Microsoft для хранения событий календаря, контактов и сообщений электронной почты. В результате были обнаружены файлы Outlook.pst, [phillip.price@e-corp.biz.pst](mailto:phillip.price@e-corp.biz.pst) (рисунок 8). Использовалась команда:

volatility\_2.6\_win64\_standalone.exe -f win7ecorpoffice2010-36b02ed3.vmem --profile=Win7SP1x64 dumpfiles -n -u -r pst$ -D ./

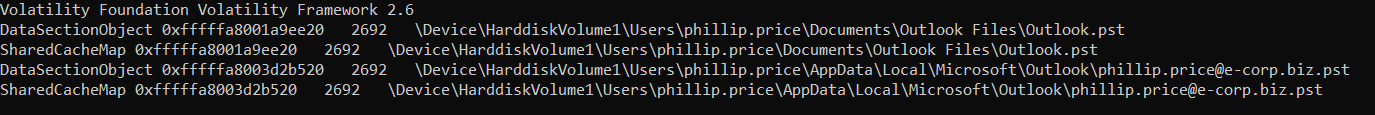


Рисунок 8 — Вывод модуля dumpfiles

Файл формата .pst представляет собой своеобразный архив текстовых файлов. Для разархивации таких файлов применяется linux-утилита pffexport.

Для использования утилиты в консоли вводится команда pffexport и ресурс для распаковки. После чего в консоль выводится сообщение об успешном экспорте (рисунок 9) и на рабочем столе создаётся директория с названием, аналогичным файлу.

Изображение выглядит как снимок экрана, текст, дисплей, Мультимедийное программное обеспечение

Автоматически созданное описание

Рисунок 9 — Процесс экспорта в консоли

Для более наглядного представления содержимого созданной директории была использована утилита «tree», предназначенная для просмотра дерева директорий в командной строке. Вывод содержимого данной утилитой показан на рисунке 10.

Изображение выглядит как снимок экрана, текст, Мультимедийное программное обеспечение, программное обеспечение

Автоматически созданное описание

Рисунок 10 — Полученная иерархия почтовых файлов и системных событий

Среди содержимого архива было обнаружено 11 входящих сообщений и 4 исходящих. Для каждого сообщения в файле InternetHeaders.txt содержится информация об отправителе. Было обнаружено только одно сообщение, полученное с адреса [kernmiles@t-online.de](mailto:kernmiles@t-online.de) (рисунок 11). Это же сообщение содержит во вложениях файл 1\_bank\_statement\_088452.doc (рисунок 12), предположительно, являющийся вредоносным.

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Рисунок 11 — Информация об отправителе письма

Изображение выглядит как текст, снимок экрана, дисплей, Шрифт

Автоматически созданное описание

Рисунок 12 — Вложения электронного письма

Для обнаруженного файла был вычислен хеш-образ md5 с помощью утилиты md5sum, что показано на рисунке 13.

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 13 — Полученный хеш-образ

### Каков адрес биткойн-кошелька, с которого требовали выкуп?

Файл Message.txt в каталоге письма содержит текстовое представление письма. В файле Message.txt сообщения Message00010 был обнаружен текст с угрозами, включающий искомый адрес BTC-кошелька 25UMDkGKBe484WSj5Qd8DhK6xkMUzQFydY (рисунок 14).

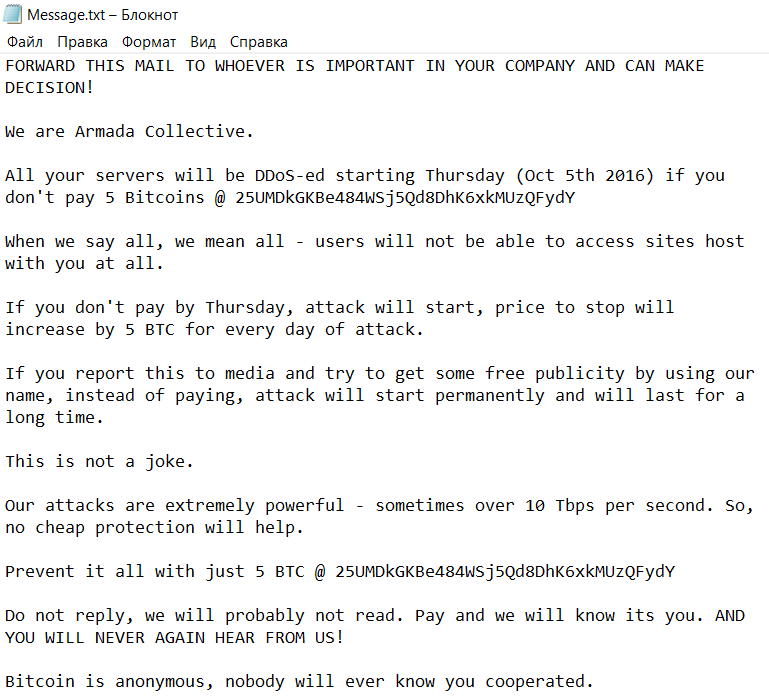


Рисунок 14 — Содержимое письма с угрозами

### Какой идентификатор присвоен системе вредоносным файлом для удаленного доступа?

Идентификатор был обнаружен с помощью уже упомянутого модуля editbox (рисунок 15). Значение идентификатора 528 812 561.

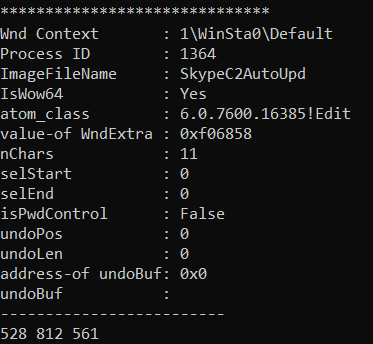


Рисунок 15 — Идентификатор вредоносного файла в системе

### По какому IPv4-адресу агент в последний раз подключался к системе с помощью средства удаленного доступа?

Средствами плагина memdump был получен снимок оперативной памяти процесса. Полученный дамп был проанализирован в Hex-редакторе, в котором с помощью поиска по строке “teamviewer” было обнаружено последние вхождение IPv4-адрес агента (рисунок 16).

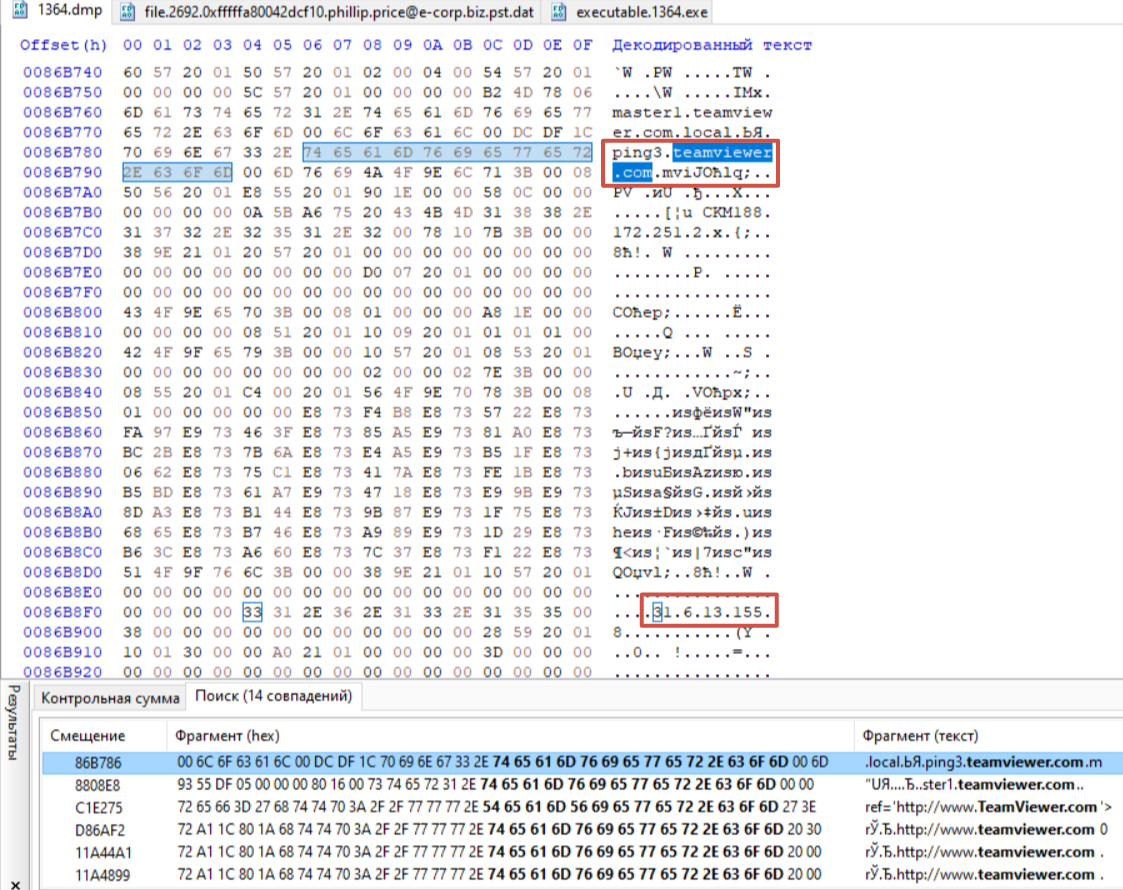


Рисунок 16 — Обнаружение IPv4-адреса агента

### Какая публичная функция в документе word возвращает полную строку команды, которая в итоге выполняется в системе?

Для анализа VBA-макросов, встроенных во вредоносный документ, использовалась Python-утилита olevba, которая является частью пакета oletools. Удалось идентифицировать место вызова процедуры и ее определение (рисунок 17). Использовалась команда:

olevba -c 1\_bank\_statement\_088452.doc



Рисунок 17 — Вызов VBA-макроса из вредоносного файла

## ecorpwin7

Образ памяти был проанализирован средствами плагина imageinfo утилиты volatility (рисунок 18).

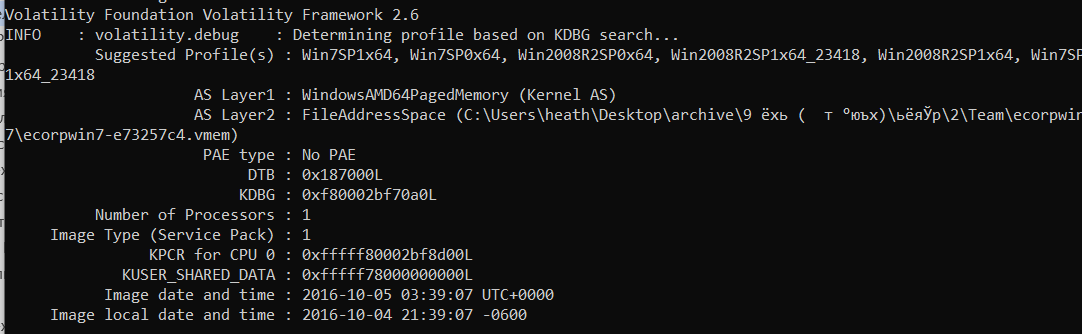


Рисунок 18 — Предварительные данные о системе

### Каков MD5-хэш вредоносного документа?

Средствами плагина dumpfiles были получены почтовые файлы из снимка оперативной памяти. Были обнаружены два доступных файла: Outlook.pst и [Outlscott.knowles@e-corp.biz-0000004.pst](mailto:Outlscott.knowles@e-corp.biz-0000004.pst).

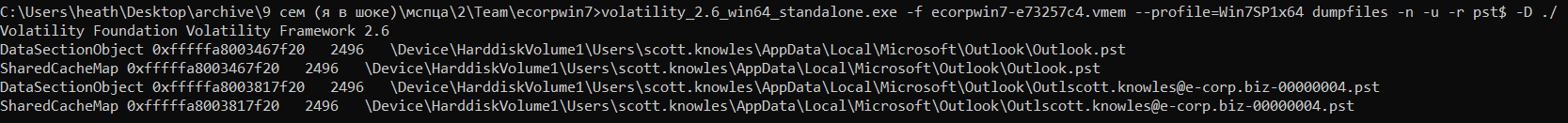


Рисунок 19 — Получение почтовых файлов

Было экспортировано содержимое второго файла (рисунок 20).

Изображение выглядит как текст, электроника, снимок экрана, дисплей

Автоматически созданное описание

Рисунок 20 — Экспорт содержимого файла

В каталоге “Top of Personal Folders” было обнаружено письмо Message0005, содержащее во вложениях подозрительный файл 1\_Important\_ECORP\_Lawsuit\_Washington\_Leak.rtf (рисунки 21-22).

Изображение выглядит как текст, снимок экрана, дисплей, программное обеспечение

Автоматически созданное описание

Рисунок 21 — Содержимое почтового ящика (Inbox)

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 22 — Содержимое Message00005

Для обнаруженного файла был вычислен хеш-образ md5 с помощью утилиты md5sum (рисунок 23).



Рисунок 23 — Полученный хеш-образ

### Каково общее имя загружаемого вредоносного файла?

В процессе анализа был запущен плагин cmdline утилиты volatility. Были обнаружены следы использования подозрительной динамической библиотеки (исполняемого файла) под названием test.dll (рисунок 24).

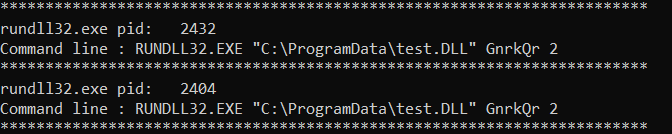


Рисунок — Вывод плагина cmdline

Среди полученных файлов были отфильтрованы файлы, содержащие в названии “test.dll”. Полученный образ был сдамплен и загружен на веб-ресурс VirusTotal, где было определено название вредоноса (рисунок 25).

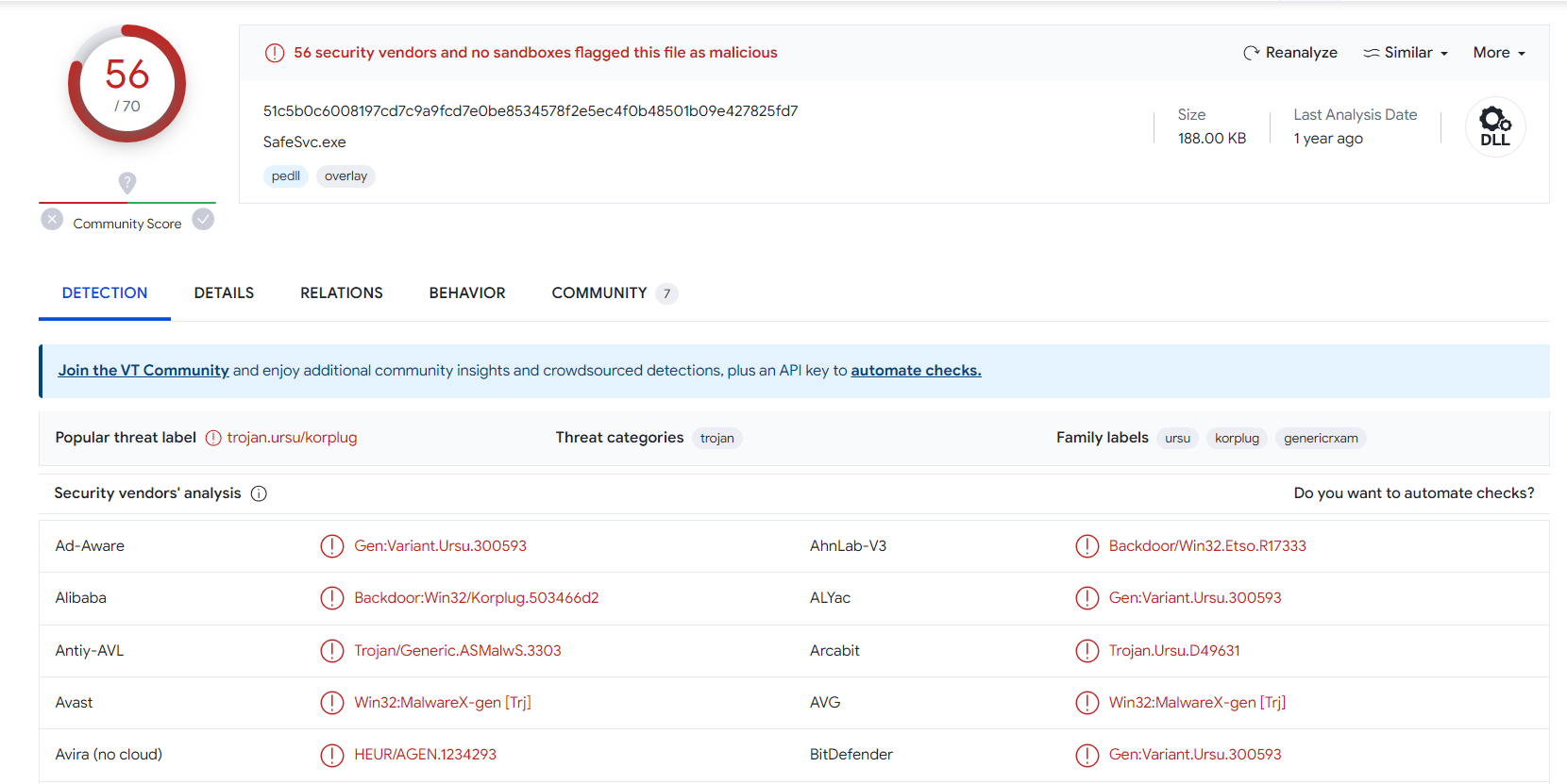


Рисунок 25 — Анализ обнаруженного файла в VirusTotal

### Какой пароль использует злоумышленник для выгрузки сжатого файла во время эксфильтрации?

В списке процессов, полученном из плагина pslist (рисунок 26) было обнаружено, что процесс 288 является родительским процессом для процессов, использующих rundll32.exe.

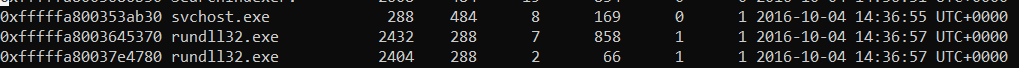


Рисунок 26 — Обнаружение родительского процесса

Для подозрительного процесса svchost.exe (pid = 288) был получен дамп оперативной памяти с использованием плагина memdump. Из полученного дампа были выбраны все строки, среди которых были отфильтрованы содержащие “.rar”. Среди полученных результатов удалось идентифицировать пароль (рисунок 27).

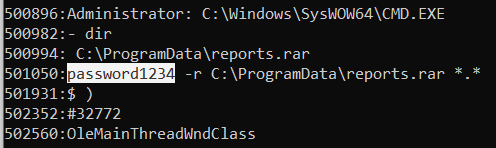


Рисунок — Обнаруженный пароль от архива

### Каков IP-адрес сервера c2 для вредоносного файла?

IPv4-адрес сервера C2 (52.90.110.169) был получен с использованием плагина netscan (рисунок 28).

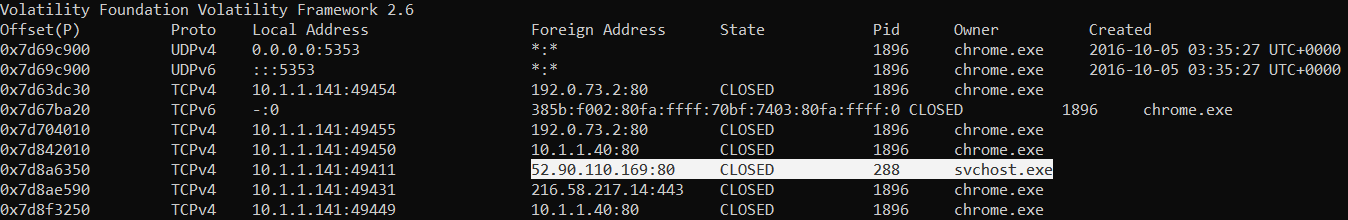


Рисунок 28 — IPv4-адрес сервера C2

### На какой адрес электронной почты было отправлено фишинговое письмо?

Адреса электронной почты отправителя ([lloydchung@allsafecybersec.com](mailto:lloydchung@allsafecybersec.com)) и получателя ([scott.knowles@e-corp.biz](mailto:scott.knowles@e-corp.biz)) были получены с использованием плагина yarascan утилиты volatility (рисунок 29).

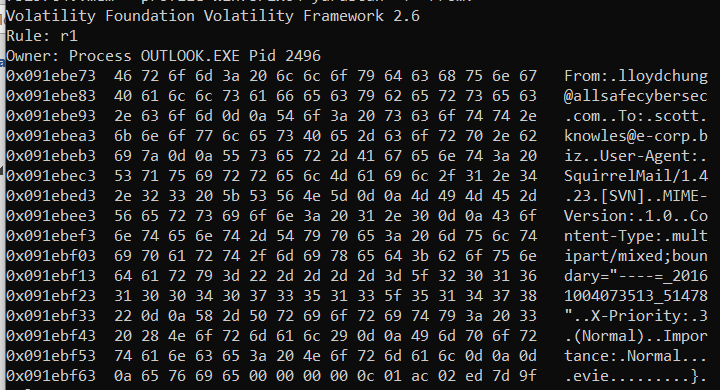


Рисунок — Адреса электронной почты отправителя и получателя

### Как называется deb-пакет, который злоумышленник инсценировал для заражения серверов E Coin?

Название .deb пакета, используемого злоумышленником для заражения, было получено из дампа оперативной памяти процесса svchost.exe путем поиска по строке “.deb” (рисунок 30).

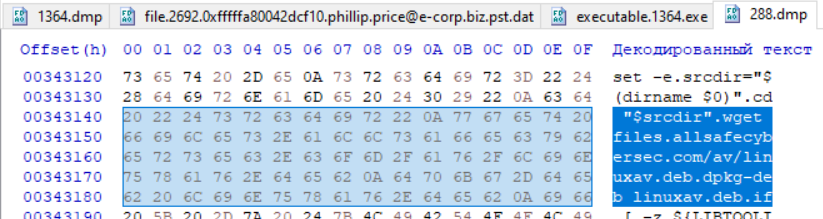


Рисунок — Извлечение названия deb-пакета, использованного для заражения

# Вывод

В ходе данной лабораторной работы с помощью утилиты Volatility 2.6 был исследован дамп оперативной памяти. Использовались служебные плагины, такие как imageinfo, procdump, pslist, filesdump, cmdline, netscan, filescan, editbox, memdump. Также использовались утилиты pffexport, md5sum, olevba. По итогам проделанной работы можно сделать следующие выводы: двум сотрудникам организации по электронной почте были отправлены зараженные файлы, содержащие макросы с вредоносным шелл-кодом. В результате, при запуске данных файлов, динамическая библиотека test.dll была загружена в процесс svchost.exe путем запуска утилиты rundll32.exe. Библиотека была идентифицирована как вредоносный файл с помощью ресурса VirusTotal. Рекомендацией для обеспечения безопасности компании в дальнейшем может быть проверка вложений писем в электронной почте на наличие вирусов и обучение персонала.