MOSCOW FORENSICS DAY '25

АНДРЕЙ ШАВЛОВСКИЙ

«ВОЗМОЖНОСТИ ИССЛЕДОВАНИЯ ИНФОРМАЦИИ МЕТОДОМ ДИНАМИЧЕСКОГО АНАЛИЗА В ПЕРСОНАЛЬНЫХ КОМПЬЮТЕРАХ НА БАЗЕ MACOS»

MFD

СЛЕДСТВЕННЫЙ КОМИТЕТ РОССИЙСКОЙ ФЕДЕРАЦИИ

Возможности исследования информации методом динамического анализа в персональных компьютерах на базе macOS

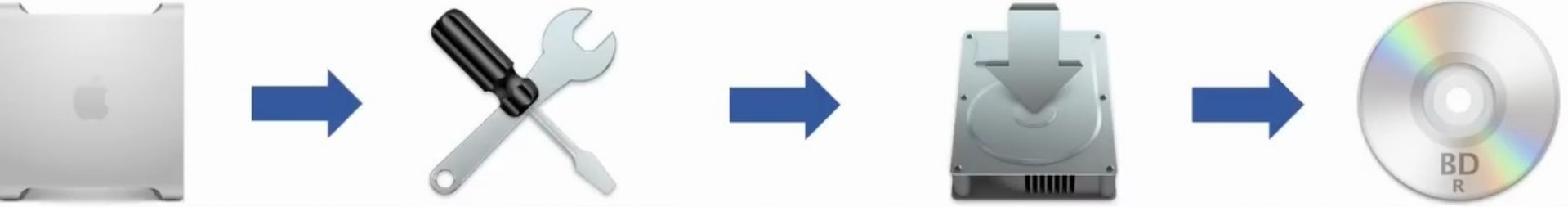
Шавловский Андрей Борисович

ashavlovsky@mail.ru

Moscow Forensics Day ‘25

Общие подходы в цифровой криминалистике

- Осмотр и фотофиксация внешнего вида объекта
- Обеспечение сохранности и неизменности информационного содержимого объекта
- Диагностика состояния объекта
- Создание точной копии информационного содержимого объекта
- Анализ данных (файловая система, файловая структура, служебные файлы ОС и др.)
- Поиск файлов с криптографической защитой, восстановление удаленной информации
- Установление сведений об информационных объектах
- Интерпретация данных
- Сохранение полученных результатов



Статический / динамический анализ

Статический анализ (Static Analysis) – исследование информации «неработающей» компьютерной системы / электронного носителя. Выполняется изучение свойств объекта (файла, кода, всей компьютерной системы) без запуска его на исполнение.

Динамический анализ (Dynamic Analysis) – исследование работающего объекта (файла, кода, всей компьютерной системы), изучаются его свойства при запуске на исполнение.

Статический / динамический анализ

	Статический анализ	Динамический анализ
+	**1.** Обеспечение сохранности и неизменности данных **2.** Результаты воспроизводимы и повторяемы	**1.** Возможность получения доступа к данным из защищенных областей информационного содержимого (в некоторых случаях) **2.** Анализ запущенных процессов, сетевых соединений
–	**1.** Могут быть пропущены данные из защищенных областей информационного содержимого **2.** Отсутствует доступ к процессам	**1.** Проблема обеспечения сохранности и неизменности данных **2.** Проблема воспроизводимости и повторяемости

Организация хранения данных в macOS

macOS использует файловую систему APFS (Apple File System) для хранения данных, введённую в macOS High Sierra (10.13) и актуальную на 2025 год. Используется AES-256 для шифрования содержимого.

Keychain (связка ключей) – хранилище учетных данных и паролей. Доступ возможен только при прохождении аутентификации при помощи пароля учётной записи, Touch ID, Face ID.

Организация хранения данных в macOS

/Users/<username>/ – «домашний» каталог пользователя, содержащий пользовательские данные. Включает подкаталоги «Documents», «Downloads», «Desktop» и др. Формируется при регистрации учетной записи.

/Users/<имя>/Library/ – пользовательские данные: история приложений, plist-файлы, SQLite-базы.

📁 Каталог	Назначение ❓
/System/Library/	⚙️ Системные файлы операционной системы
/Library/	📜 Настройки приложений, связки ключей (keychain)
/Users/	👤 Пользовательские каталоги
/private/var/	📖 Временные файлы, журналы, кэш
/etc/ /private/etc/	📡 Конфигурационные файлы
/Volumes/	💿 Монтированные носители (тома)
/Applications/	📚 Инсталлированное программное обеспечение

О файлах формата PLIST

🔍 **Property List (plist)** — формат представления данных в macOS, iPadOS и iOS. Используется для хранения настроек приложений, конфигураций системы, учётных записей, журналов активности приложений и др.

💾 **Основные форматы plist-файлов**. Plist-файлы поддерживают несколько форматов представления данных:

📌 **XML** – текстовый формат на основе XML. Используется для редактирования и отладки.

📌 **Binary** – машинно-ориентированный формат. Нечитаем без конвертации. Большинство системных plist (например, в /Library/Preferences/ или /var/db/dslocal/) хранятся именно так.

О файлах формата PLIST

Фрагмент содержимого plist-файла формата **binary**

Фрагмент содержимого plist-файла формата **xml**

Учетные записи в macOS

Информация о зарегистрированных учетных записях (локальных пользователях) содержатся в plist-файлах 📁: **/var/db/dslocal/nodes/Default/users/<username>.plist.** Данные файлы содержат различные сведения об учетной записи, включая раздел с данными «*ShadowHashData*».

convert

```
</string>
    </array>
    <key>ShadowHashData</key>
    <array>
        <data>
        YnBsaXN0MDDSAQIDCl8QHlNSUC1SRkM1MDU0LTQwOTYtU0hBNTEyLVBCS0RG
        Ml8QFFNBTFRFRC1TSEE1MTItUEJLREYy0wQFBgcICVh2ZXJpZmllclRzYWx0
        Wml0ZXJhdGlvbnNPEQIAuNCn0qCScFgI91kVmeLr0t7/aZKCWs2LQrztDYDE
        GlV8dUOP9LvKqyTg6P+137WnbtCEvq9l0Fh3RppFw6cEtPYAKK/Z0xKZ9cHL
        LkVy7RWXa4wDDR0hk2RXbaukj63gy0Buis0PTJDXPUNbmPwR3wWV9eUFFJVT
        LdeI1CdjOmBTDYsmCQc1G0NJ9Ut1rWhsvaGE1cWGn5UMU+nglDRA+HRp2rEW
        iQLQWWpiyy2xzUko7wUEtM2Us0qlmvVm0irSS7KdAMhFvTtThspnIjPo4MLH
        xRypCzsQ4glGS/23Uv4Ow3k89H0t5Y+c4KAiLip6ku6ShqST+3NM+UJYIjYx
        DyClWFO/aHVV+nBp3hGB5jY+wFE1mPj+gZOMazQKE34r8D7afN6V9S5e4fc/
        l8uUHEJg08KXxCXHoga38m5C3ksOs3dOfKSCfsF3koy1k5iaojaXmY7ysY1a
        oSQ4wnv+LmCJDWF3cyDB/ZhFnkP7Eb8nzokQKvtzLBFSolWOfkGxu5O4/NUf
        sTQjvgozuKCvQYCds3EQgPPXsaIXacscUE7sYZPwoq73IYis2NhrdyxR4GhN
        EXuxTHGH4UlWr9EuPHS+hyQ4WWF1nbV3nxEq1hZ7O/sjGYcCDx3gfBvm1sry
        dwxNS8c3rmCegNBJGkBDB0vxi5ttEfb39Z6JLU9ivXpPECDLXYNPkja4JEtO
        cpW6N9uPdcXQSamYjntPM6PYFIMAyhH0JNMLBQYMDQ5XZW50cm9weU8QgN4N
        fsj0ywRDZBzKiv+iWEvwVKdYr00rK/OlV53xmZJgyStlT+8+CDQfF0qXq7wg
        OUJNVThS196fpxIbUQguQTbiYwnUYP4EDzEYE9xlFhw4iuWjiCpIfm/YM0Cf
        UDdfyTImFb6eRnGiB0fjVG9QJwn3DoxDe0HNzEWjItSDSM9DTxAgGqnSYdN3
        kxjGCh9+lUiYARFxLPf/Tu1T37xhvNLeKlwR8p8ACAANAC4ARQBMAFUAWgBl
        AmkCjAKPApYCngMhA0QAAAAAAAACAQAAAAAAAAAPAAAAAAAAAAAAAAAAAAAD
        Rw==
        </data>
    </array>
```

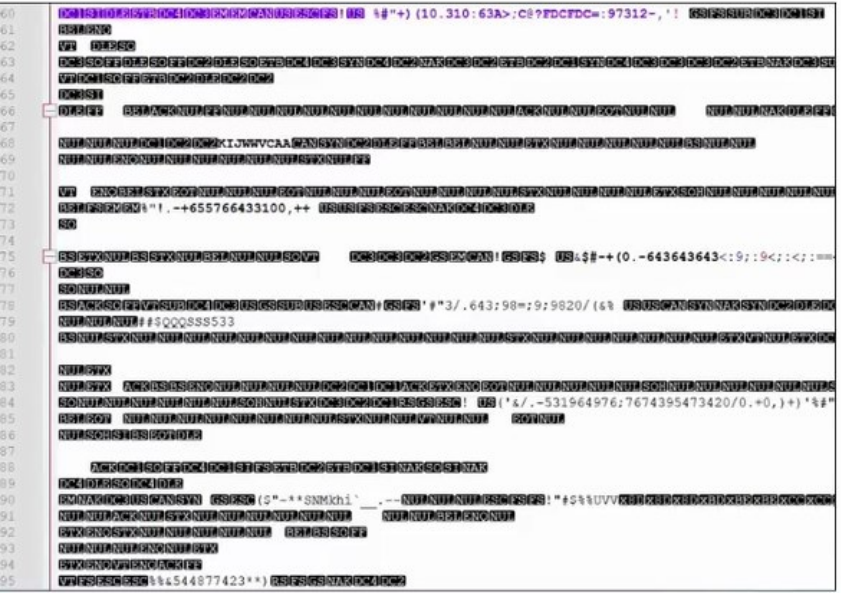
Извлечение хеша пароля

1. Обеспечение доступа к файлу:

Расположение: **/var/db/dslocal/nodes/Default/users/<username>.plist**

2. Декодирование его содержимого из binary в xml формат:

[bash] > **plutil -convert xml1 <username>.plist -o user.xml** *либо* использование специализированных программных продуктов / скриптов



convert

Извлечение хеша пароля

1. Обеспечение доступа к файлу:

Расположение: **/var/db/dslocal/nodes/Default/users/<username>.plist**

2. Декодирование его содержимого из binary в xml формат:

[bash] > **plutil -convert xml1 <username>.plist -o user.xml** *либо* использование специализированных программных продуктов / скриптов

3. Декодирование (base64) «ShadowHashData»

[bash] > **echo "*ShadowHashData*" | base64 --decode**

```
</string>
    </array>
    <key>ShadowHashData</key>
    <array>
        <data>
        YnBsaXN0MDDSAQIDCl8QHlNSUC1SRkMlMDU0LTQwOTYtU0hBNTEyLVBCS0RG
        Ml8QFFNBTFRFRC1TSEE1MTItUEJLREYy0wQFBgcICVh2ZXJpZmllclRzYWx0
        Wml0ZXJhdGlvbnNPEQIAuNCn0qCScFgI91kVmeLr0t7/aZKCWs2LQrztDYDE
        GlV8dUOP9LvKqyTg6P+137WnbtCEvq910Fh3RppFw6cEtPYAKK/Z0xKZ9cHL
        LkVy7RWXa4wDDR0hk2RXbaukj63gy0Buis0PTJDXPUNbmPwR3wWV9eUFFJVT
        LdeIlCdjOmBTDYsmCQc1G0NJ9Ut1rWhsvaGE1cWGn5UMJ+nglDRA+HRp2rEW
        iQLQWWpiyy2xzUko7wUEtM2Us0qlmvVm0irSS7KdAMhFvTtThspnIjPo4MLH
        xRypCzsQ4glGS/23Uv4Ow3k89H0t5Y+c4KAiLip6ku6Shq5T+3NM+UJYIjYx
        DyClWFO/aHVV+nBp3hGB5jY+wFElmPj+gZOMazQKE34r8D7afN6V9S5e4fc/
        l8uUHEJg08KXkCXHoga38m5C3ksOs3dOfKSCfsF3koylk5iaojaXmY7ysY1a
        oSQ4wnv+LmCJDWF3cyDB/ZhFnkP7Eb8nzokQKvtzLBFSolWOfkGxu5O4/NUf
        sTQjvgozuKCvQYCds3EQgFPXsaIXacscUE7sYZPwoq73IYis2NhrdyxR4GhN
        EXuxTHGH4UlWr9EuPHS+hyQ4WWFlnbV3nxEq1hZ7O/sjGYcCDx3gfBvmlszy
        dwxNS8c3rmCeqNBJGkBDB0vxi5ttEfb39Z6JLU9ivXpPECDLXYNPkja4JEtO
        cpW6N9uPdcXQ5amYjntPM6PYFIMAyhH0JNMLBQYMDQ5XZW50cm9weU8QgN4N
        fsj0ywRDZBzKiv+iWEvwVKdYr00rK/OlV53xmZJgy5tlT+8+CDQfF0qXq7wg
        OUJNVThS196fpxIbUQguQTbiYwnUYP4EDzEYE9xlFhw4iuWjiCpIfn/YM0Cf
        UDdfyZImFb6eRnGiB0fjVG9QJwn3DoxDe0HNzEWjItSDSM9DTxAgGqnSYdN3
        kxjGCh9+lUiYARFxLPf/TulT37xhvNLeKlwR8p8ACAANAC4ARQBMAFUAWgBl
        AmkCjAKPApYCngMhA0QAAAAAAAACAQAAAAAAAAAPAAAAAAAAAAAAAAAAAAAD
        Rw==
        </data>
    </array>
```

base64 decode

```
{
  "SALTED-SHA512-PBKDF2" => {
    "entropy" => <de0d7ec8 f4cb0443 641cca8a ffa2584b f054a758 af4d2b2b f3a5579d
 f1999260 c92b654f ef3e0834 1f174a97 abbc2039 424d5538 52d7de9f a7121b51 082e413
6 e26309d4 60fe040f 311813dc 65161c38 8ae5a388 2a487e6f d833409f 50375fc9 322615
be 9e4671a2 0747e354 6f502709 f70e8c43 7b41cdcc 45a322d4 8348cf43>
    "iterations" => 62111
    "salt" => <1aa9d261 d3779318 c60a1f7e 95489801 11712cf7 ff4eed53 dfbc61bc d2
de2a5c>
  }
```

Хеш пароля от учетной записи


```
{
  "SALTED-SHA512-PBKDF2" => {
    "entropy" => <de0d7ec8 f4cb0443 641cca8a ffa2584b f054a758 af4d2b2b f3a5579d
 f1999260 c92b654f ef3e0834 1f174a97 abbc2039 424d5538 52d7de9f a7121b51 082e413
6 e26309d4 60fe040f 311813dc 65161c38 8ae5a388 2a487e6f d833409f 50375fc9 322615
be 9e4671a2 0747e354 6f502709 f70e8c43 7b41cdcc 45a322d4 8348cf43>
    "iterations" => 62111
    "salt" => <1aa9d261 d3779318 c60a1f7e 95489801 11712cf7 ff4eed53 dfbc61bc d2
de2a5c>
  }
```

PBKDF2-SHA512 — криптографический алгоритм для вычисления хешей на основе значения пароля, который использует функцию PBKDF2 (Password-Based Key Derivation Function 2) в комбинации с хеш-функцией SHA-512 (описан в RFC 2898 и обновлённый в RFC 8018). Некоторые его особенности:

📌 **«entropy»** – итоговый результат работы хеш-функции. Хранится в шестнадцатеричной форме.

📌 **«iterations»** – установленное количество итераций PBKDF2 (в данном случае 62111).

📌 **«salt»** – уникальная последовательность байт, которая добавляется к паролю перед хешированием.

Microsoft Windows & Apple macOS

mac OS

🔒 NT-хеш
без соли и итераций

🔐 Хеш
PBKDF2-SHA512
(соль + итерации)

Подготовка к перебору пароля

ml<iterations>$<salt-hex>$<entropy-hex>

1. **$ml** – обозначает версию операционной системы Mac OS X Lion (10.8) и выше. С этой версии ОС организовано хеширование паролей по алгоритму PBKDF2-SHA512. По данному обозначению hashcat определяет алгоритм PBKDF2-SHA512 в macOS.

2. **$<iterations>** – количество итераций.

3. **$<salt-hex>** – «соль», представленная в hex-виде.

4. **$<entropy-hex>** – итоговой хеш после расчета.

```
$ml$62111$1aa9d261d3779318c60a1f7e9548980111712cf7ff4eed53dfbc61bcd2de2a5c$de0d7ec8
f4cb0443641cca8affa2584bf054a758af4d2b2bf3a5579df1999260c92b654fef3e08341f174a97abbc2
039424d553852d7de9fa7121b51082e4136e26309d460fe040f311813dc65161c388ae5a3882a487e
6fd833409f50375fc9322615be9e4671a20747e3546f502709f70e8c437b41cdcc45a322d48348cf43
```

Полученный хеш для подбора пароля

Извлечение хеша пароля: скрипты


```
$ml$62111$1aa9d261d3779318c60a1f7e954898011171
2cf7ff4eed53dfbc61bcd2de2a5c$de0d7ec8f4cb0443641c
ca8affa2584bf054a758af4d2b2bf3a5579df1999260c92b6
54fef3e08341f174a97abbc2039424d553852d7de9fa712
1b51082e4136e26309d460fe040f311813dc65161c388ae
5a3882a487e6fd833409f50375fc9322615be9e4671a207
47e3546f502709f70e8c437b41cdcc45a322d48348cf43
```

Хеш для подбора пароля

```
Windows PowerShell
PS D:\scripts> py .\script.py .\administrator.plist
$ml$62111$1aa9d261d3779318c60a1f7e9548980111712cf7ff4eed53dfbc61bcd2de2a5c$de0d7ec8
f4cb0443641cca8affa2584bf054a758af4d2b2bf3a5579df1999260c92b654fef3e08341f174a97abb
c2039424d553852d7de9fa7121b51082e4136e26309d460fe040f311813dc65161c388ae5a3882a487e
6fd833409f50375fc9322615be9e4671a20747e3546f502709f70e8c437b41cdcc45a322d48348cf43
```

Результат работы скрипта

```python
#!/usr/bin/env python3
import plistlib
import io
import sys
from pathlib import Path
def extract_shadowhash(plist_path: Path):
    with open(plist_path, "rb") as f:
        plist_data = plistlib.load(f)
    shadow = plist_data.get("ShadowHashData")
    if not shadow:
        raise ValueError("В файле нет ключа 'ShadowHashData'")
    blob = shadow[0]
    inner = plistlib.load(io.BytesIO(blob))
    pbkdf2 = inner.get("SALTED-SHA512-PBKDF2")
    if not pbkdf2:
        raise ValueError("Нет секции 'SALTED-SHA512-PBKDF2'")
    iterations = pbkdf2["iterations"]
    salt_hex = pbkdf2["salt"].hex()
    entropy_hex = pbkdf2["entropy"].hex()
    hashcat_line = f"$ml${iterations}${salt_hex}${entropy_hex}"
    return hashcat_line
def main():
    if len(sys.argv) < 2:
        print(f"Использование: {sys.argv[0]} user.plist [output.txt]")
        sys.exit(1)
    plist_path = Path(sys.argv[1])
    if not plist_path.exists():
        print(f"Файл не найден: {plist_path}")
        sys.exit(1)
    hash_line = extract_shadowhash(plist_path)
    if len(sys.argv) >= 3:
        out_path = Path(sys.argv[2])
        with open(out_path, "w") as f:
            f.write(hash_line + "\n")
        print(f"[+] Хэш сохранён в {out_path}")
    else:
        print(hash_line)
if __name__ == "__main__":
    main()
```

Извлечение хеша пароля: программы

ДАННЫЕ АРТЕФАКТА

Имя пользователя	**administrator**
Пароль или токен	**ml62111 $1aa9d261d3779318c60a1f7e9548980111712cf7ff4eed53dfbc61bcd 2de2a5c $de0d7ec8f4cb0443641cca8affa2584bf054a758af4d2b2bf3a5579df1 999260c92b654fef3e08341f174a97abbc2039424d553852d7de9fa712 1b51082e4136**
Тип	Пароли и токены
ИД элемента	2602254
Исходный артефакт	User Accounts - macOS

ml62111$1aa9d261d3779318c60a1f7e9548980111712cf7ff4eed53dfbc61bcd2de2a5c$de0d7ec8f4cb0443641cca8affa2584bf054a758af4d2b2bf3a5579df1999260c92b654fef3e08341f174a97abbc2039424d553852d7de9fa7121b51082e4136e26309d460fe040f311813dc65161c388ae5a3882a487e6fd833409f50375fc9322615be9e4671a20747e3546f502709f70e8c437b41cdcc45a322d48348cf43

Отсутствующая часть хеша

⚔️ Подбор пароля: hashcat

🔎 hashcat -m 7100 -a 0 hash.txt wordlist.txt

Windows PowerShell

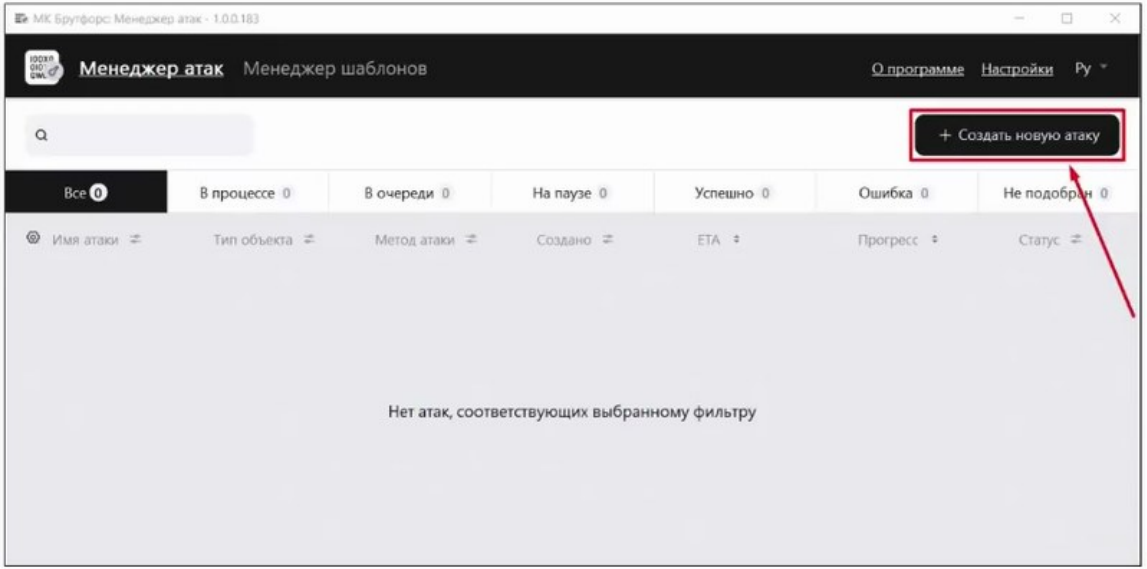
```
PS C:\Portable Programs\hashcat-7.1.2\hashcat-7.1.2> .\hashcat.exe -m 7100 -a 0 .\hashmac.txt .\wordlist.txt
hashcat (v7.1.2) starting
```

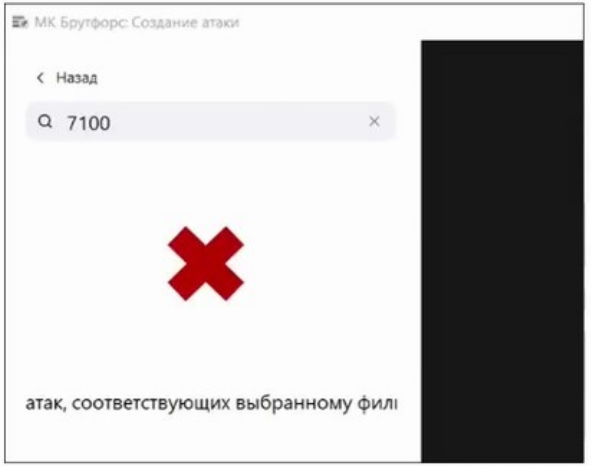
🔄 Выполнение hashcat

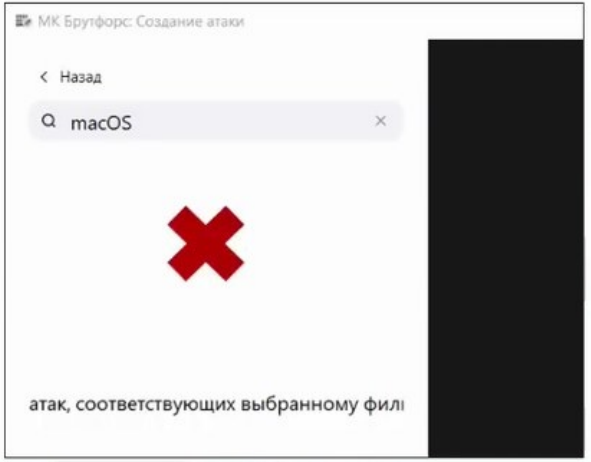
```
$ml$62111$1aa9d261d3779318c60a1f7e9548980111712cf7ff4eed53dfbc61bcd2de2a5c$de0d7ec8f4cb0443641cca8
affa2584bf054a758af4d2b2bf3a5579df1999260c92b654fef3e08341f174a97abbc2039424d553852d7de9fa7121b510
82e4136e26309d460fe040f311813dc65161c388ae5a3882a487e6fd833409f50375fc9322615be9e4671a20747e3546f5
02709f70e8c437b41cdcc45a322d48348cf43:22012

Session..........: hashcat
Status...........: Cracked
Hash.Mode........: 7100 (macOS v10.8+ (PBKDF2-SHA512))
Hash.Target......: $ml$62111$1aa9d261d3779318c60a1f7e9548980111712cf7f...48cf43
```

✅ Результат работы – установлено значение пароля

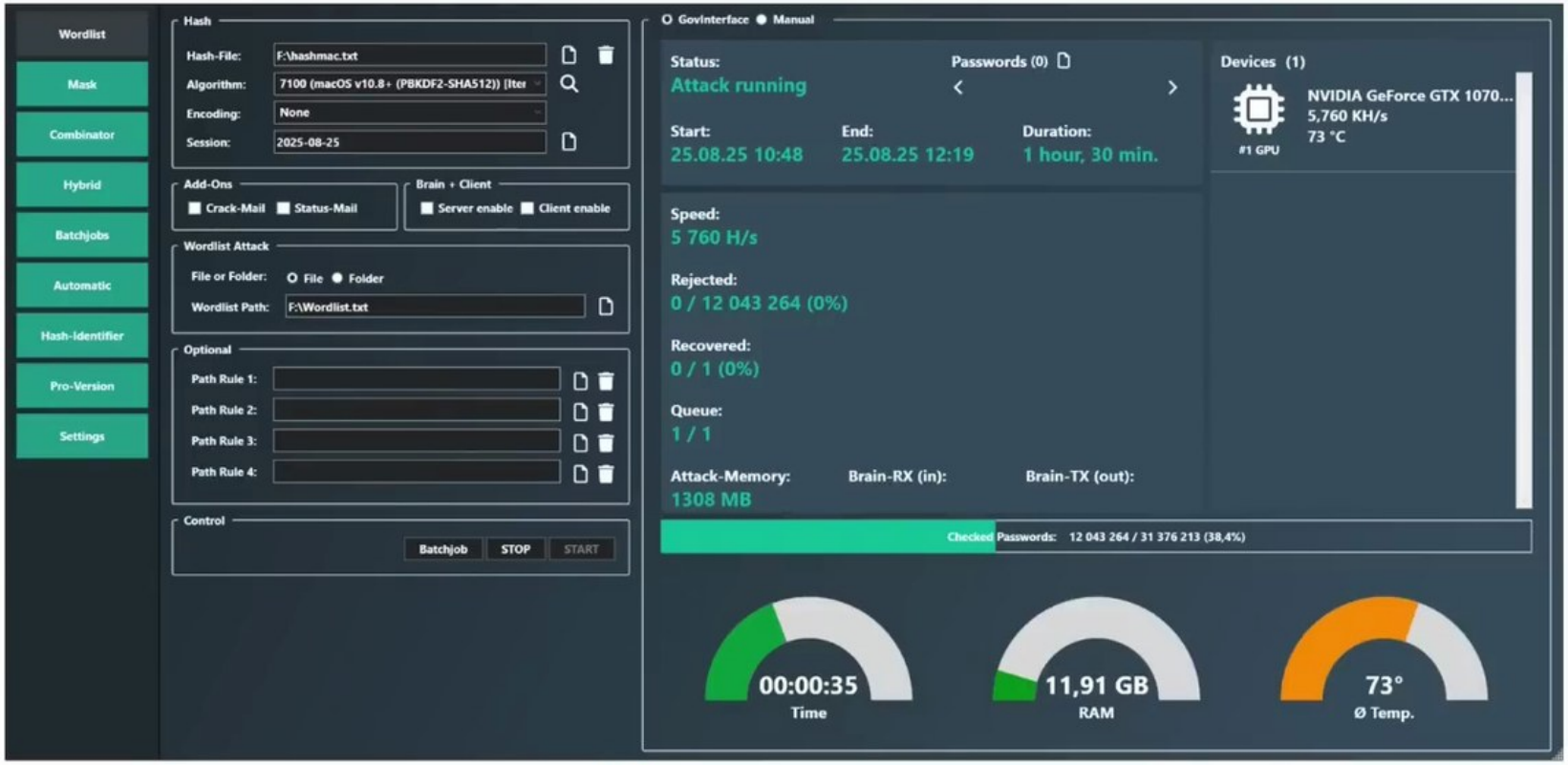
Подбор пароля: gui на основе hashcat



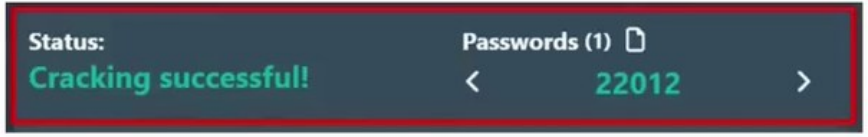




Подбор пароля: gui на основе hashcat



Результат работы – установлено значение пароля

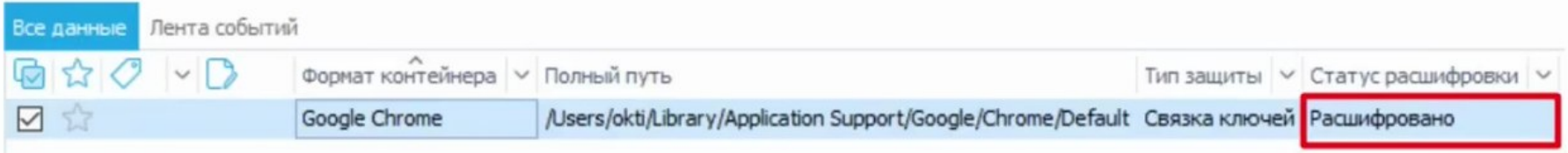


Анализ и интерпретация данных

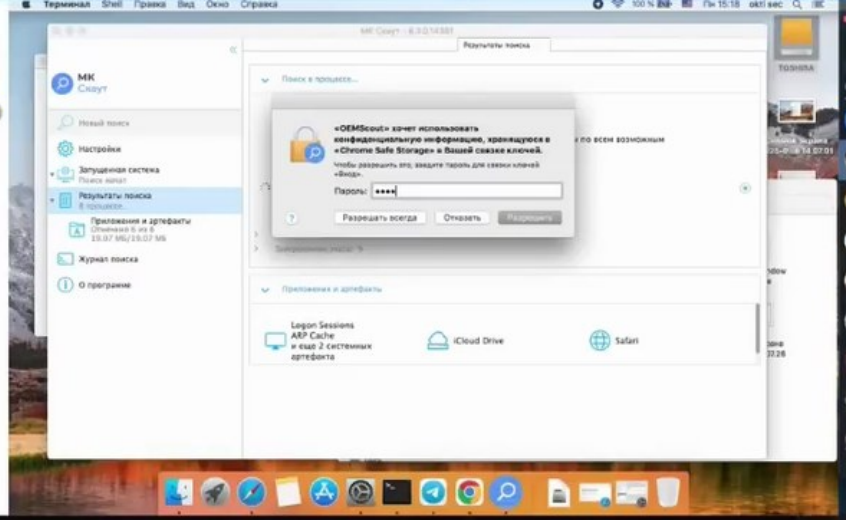
📌 Виртуализация объекта и исследование его точной копии.

📌 Исследование непосредственно самого объекта с предварительно полученным разрешением на внесение изменений в соответствии с п. 3 ч. 4 ст. 57 УПК РФ (если речь о судебной экспертизе ⚖️).

Фрагмент интерпретации данных **без обеспечения** доступа к связке ключей



Фрагмент интерпретации данных **с обеспечением** доступа к связке ключей


Терминал Shell Правка Вид Окно Справка
МК Скаут
Результаты поиска
Поиск в процессе...
Настройки
Запущенная система
Результаты поиска
Приложения и артефакты
Журнал поиска
О программе
«OEMScout» хочет использовать конфиденциальную информацию, хранящуюся в «Chrome Safe Storage» в вашей связке ключей.
Пароль:
Разрешать всегда
Отказать
Разрешить
Приложения и артефакты
Logon Sessions
ARP Cache
и еще 2 системных артефакта
iCloud Drive
Safari


Чаты
Архив
Фотография
Напряжены сидим
А сейчас(?)

Итог

- Понимание механизма образования следов в цифровой среде для обеспечения проверяемости и достоверности получаемых результатов
- Методы статического и динамического анализа взаимодополняют процесс исследования, обеспечивая его полноту и всесторонность

СЛЕДСТВЕННЫЙ КОМИТЕТ РОССИЙСКОЙ ФЕДЕРАЦИИ

Спасибо за внимание!

Вопросы?

Шавловский Андрей Борисович

ashavlovsky@mail.ru

Moscow Forensics Day ‘25