Определение поверхности атаки сложных систем с помощью Natch

Довгалюк Павел

ИСП РАН

Что такое Natch

- **Автоматизированный анализ поверхности атаки**
  - Интерфейсы приложений
  - Компоненты и функции программ
- **x86_64 и немножко AArch64**
- **C/C++, Go, Python, Java, JavaScript***
- **Регрессионное тестирование**
- **Отчёты для аналитиков и тестировщиков**

Динамический анализ

- **Можно увидеть подробности о конкретном сценарии работы**
- **Более сложная настройка, чем при статическом анализе**
- **Нужно много времени на анализ**

Динамический анализ – это сложно

- Полносистемный анализ
- Отслеживание потока управления и потоков данных
- Нужна тщательная настройка сценариев
- Не обойтись без отладочных символов

Почему анализ может быть медленным?

- Медленный многоядерный процессор вместо одного быстрого ядра
- Используется графический интерфейс
- Есть лишние программы
- Чрезмерно большой сценарий
- Непроизводительные паузы в сценарии
- Natch не слишком быстр

Лишние программы и GUI

- Тратят процессорное время
- Попадают в лог записи/воспроизведения
- Попадают в результаты анализа
- GUI усложняет автоматизацию тестирования

- unattended-upgrades в Ubuntu

Сокращение сценария

- Чем короче сценарий, тем лучше
- Нужно как можно быстрее отправить входные данные и завершить ВМ
- Лучше стартовать анализ для уже запущенных программ, подведённых к нужному состоянию

- Но для Java-приложений старт должен попадать в сценарий

Поиск поверхности атаки

- Какие данные важны?
- Куда они не должны проникать?
- Как они должны обрабатываться?

- Все потоки данных сразу отследить невозможно

Последовательность работы

- Настроить ВМ, удалить лишние сервисы
- Определить содержание сценария
- Определить входные данные
- Записать сценарий, чётко выделив временные границы
- Пометить выбранные входные данные
- Воспроизвести сценарий

Примеры

- **Поиск поверхости атаки в тестовом приложении на Java – spring-petclinic**
- **Анализ утечек данных в настоящих СУБД**

Spring-petclinic

Owner

First Name

myownerfirst

Last Name

myownerlast

Address

myowneraddr

City

myownercity

Telephone

1234567890

Add Owner

Spring-petclinic

Owner Information

Name	**myownerfirst myownerlast**
Address	myowneraddr
City	myownercity
Telephone	1234567890

Edit Owner | Add New Pet

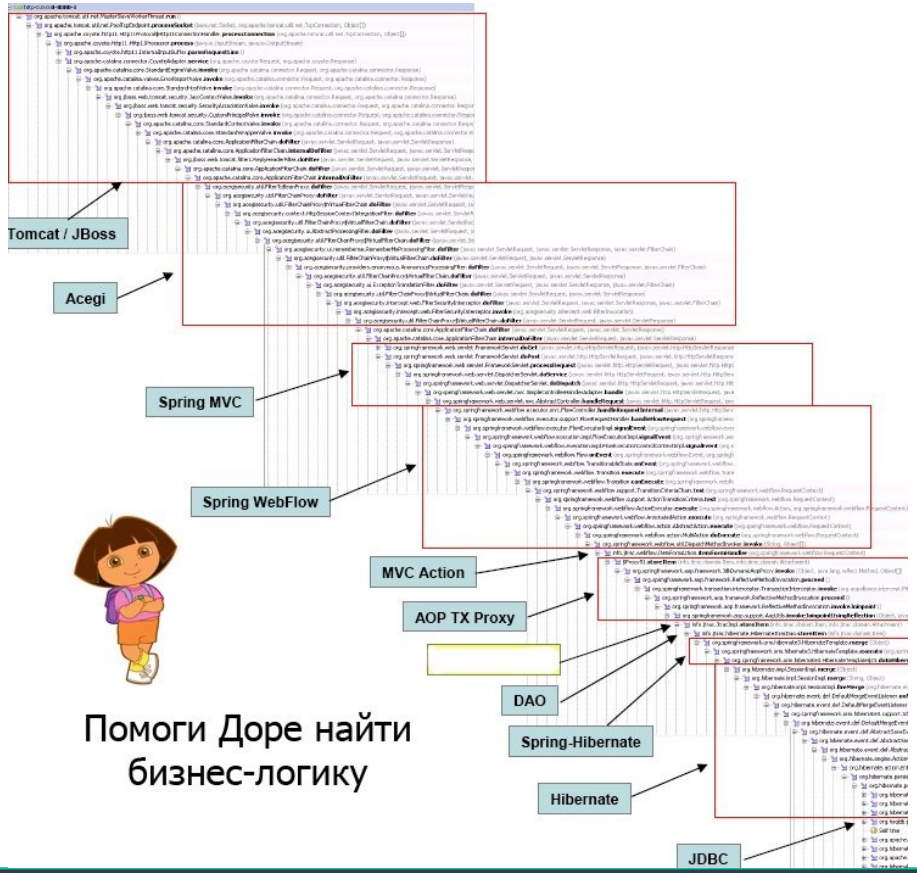
Pets and Visits

Анализ поверхности атаки petclinic

- Поверхность атаки сложно найти вручную
- В sbom внутри jar-файла описаны 92 зависимости

- Много абстракций
- Глубина стека 200+ функций

На что похож backtrace / call graph



План исследования приложения petclinic

- **В веб-интерфейсе**
  - Создать карточку владельца
  - Просмотреть карточку
- **В инструменте Natch**
  - Пометить полученные данные формы для отслеживания
  - Убедиться, что помеченные данные вернулись назад пользователю при просмотре формы
- **В интерфейсе анализа SNatch**
  - Выбрать интересные модули и функции
- **Фаззинг или поиск багов вручную**

Входные данные

```
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.
8,application/signed-exchange;v=b3;q=0.7
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: http://127.0.0.1:49152/owners/new
Accept-Encoding: gzip, deflate, br, zstd
Accept-Language: en-US,en;q=0.9,ru;q=0.8
Cookie: sidenav-state=pinned; csrftoken=v82VHdvacMs8EraSTY528xP49j9Efbs8; django_language=en-us; sessio
n=eyJsb2dnZWRfaW4iOmZhbHNlfQ.ZyHHTA.-IH3r05WSU8-MNKDjZzuowFfwcQ; JSESSIONID=D1625244AC34D07B3F8D54F79F7
7CC51

firstName=myownerfirst&lastName=myownerlast&address=myowneraddr&city=myownercity&telephone=1234567890
```

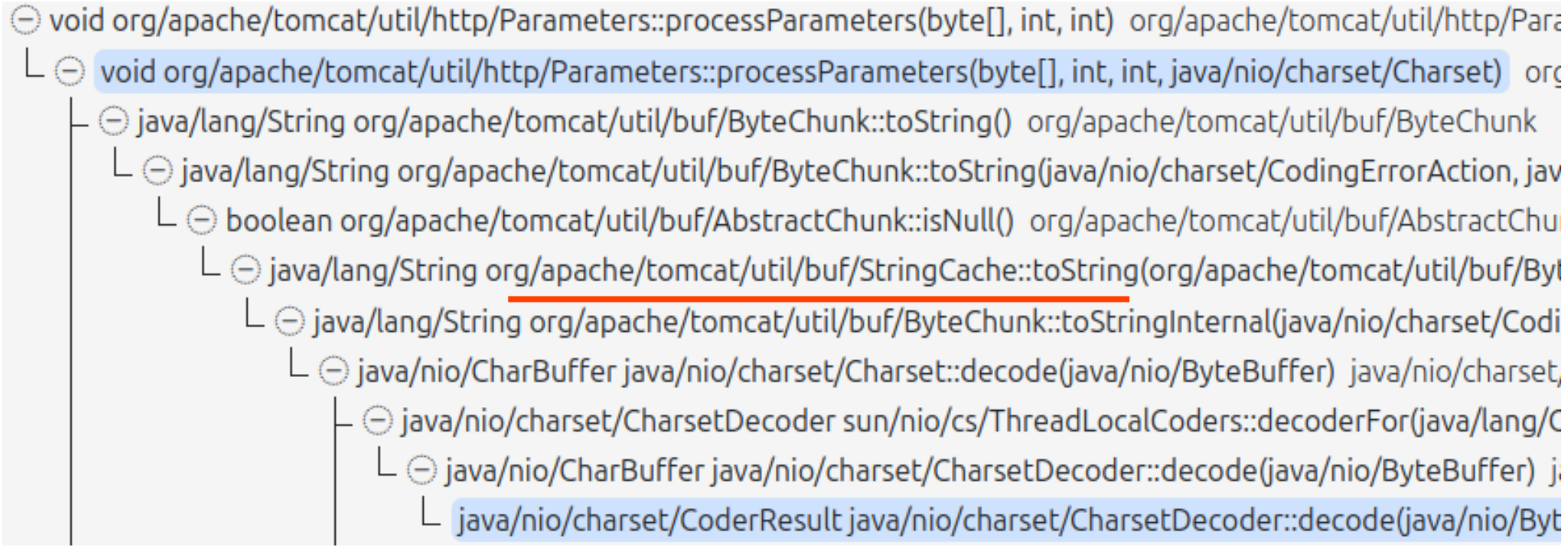
Страница с карточкой владельца с подсвеченными входными данными

```
<table class="table table-striped">
  <tr>
    <th>Name</th>
    <td><b>myownerfirst myownerlast</b></td>
  </tr>
  <tr>
    <th>Address</th>
    <td>myowneraddr</td>
  </tr>
  <tr>
    <th>City</th>
    <td>myownercity</td>
  </tr>
  <tr>
    <th>Telephone</th>
    <td>1234567890</td>
  </tr>
</table>
```

Изучение поверхности атаки

- Разбор запроса
- Валидация входных данных
- Работа с БД
- Вывод html

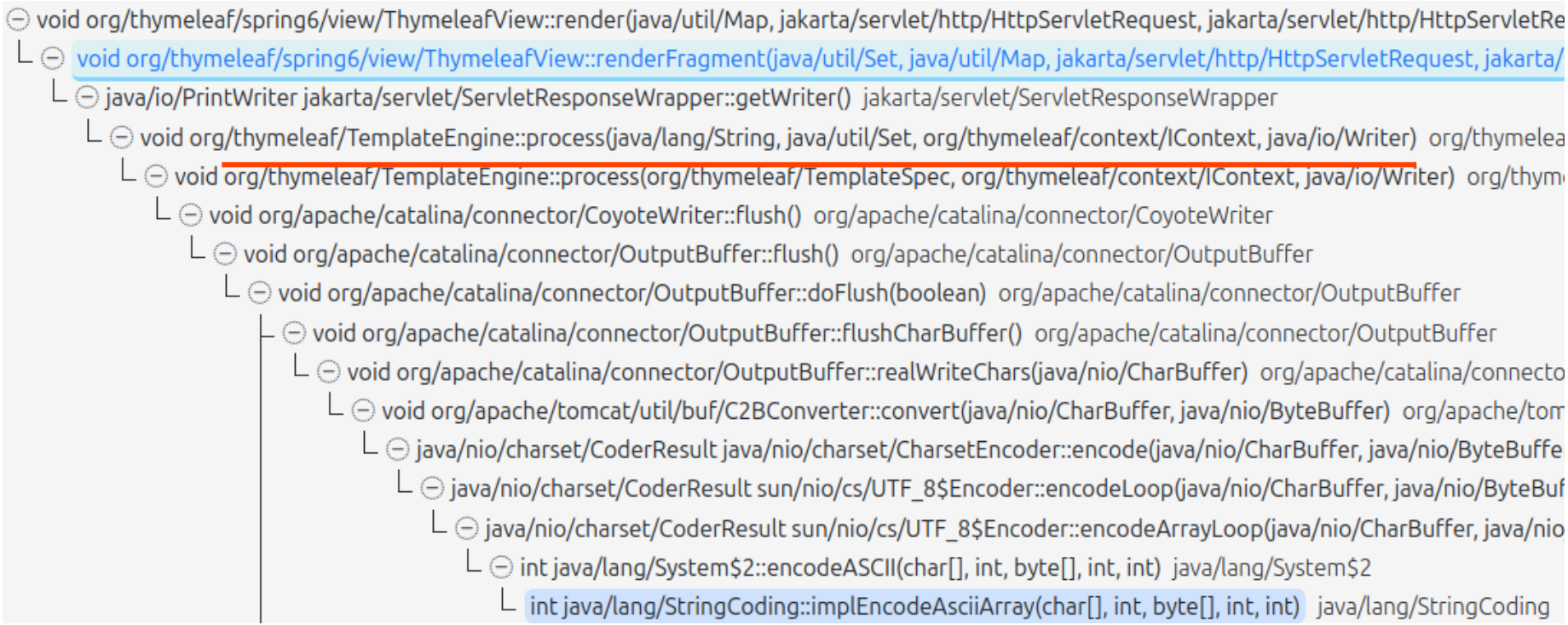
Разбор запроса: apache



Валидация входных данных: hibernate

Вставка значений в БД: h2

Рендер страницы: thymeleaf



Экранирование вывода: unbescape

unbescape

- HtmlEscapeUtil
- Функция escape выглядит не очень сложной
- А вот в unescape много ветвлений и циклов
- Возможности для багов: escape и unescape существуют в трёх (!) почти одинаковых экземплярах (одна для строк, другая для потоков, третья для массивов)

unescape()

```
1:268,269c
3:252,253c
                if ((f - (i + 1)) <= 0) {
                    // We weren't able to consume any alphanumeric
2:295,297c
                if (escapei == 0) {
                    // We weren't able to consume any decimal chars
                    writer.write(c1);
```

Фаззинг escape + unescape

```
== libFuzzer crashing input ==
MS: 0 ; base unit: 0000000000000000000000000000000000000000
0x26,0x23,0x78,0xc2,0xc4,0xc4,0xc4,0xc4,0xc4,0xc4,0xc5,0xc4,0xc4,0xc4,0xc4,0xc4,0xc4,0xc4,0xc4,
&#x\302\304\304\304\304\304\304\305\304\304\304\304\304\304\304\304
artifact_prefix='./'; Test unit written to ./crash-05d6a8daa2d5eeced1d3cfcabe650329617db4d9
Base64: JiN4wsTExMTExMXExMTExMTExA==
 in: &#x␂ĄĄąĄĄĄĄ
esc: &amp;#x&#132;&Aogon;&Aogon;&aogon;&Aogon;&Aogon;&Aogon;&Aogon;
out: &#x„ĄĄąĄĄĄĄ
```

Результат анализа petclinic

- **Библиотеки, входящие в поверхность атаки**
  - unbescape
  - thymeleaf
  - hibernate
  - h2
- **В коде в явном виде фигурируют не все**
- **Заимствованный код тоже нужно тестировать**
- **Собственные тесты в библиотеках есть, но без фаззинга**
- **Возможно фаззинг делается разработчиками, но тесты не публикуются**

Natch для поиска утечек

1. **Выделить чувствительные данные**
2. **Запустить анализ сценария работы системы**
3. **Получить поверхность атаки**
   - легальное использование данных – надо протестировать
   - утечка – нужно предотвратить

Кейс – удаление данных в СУБД

- Если удалить данные с помощью DELETE, исчезнут ли они из БД?
- А если удалить таблицу?
- А если даже всю базу?

Кейс – удаление данных в СУБД

- **Добавляем данные в таблицу**
- **Удаляем данные**
  - DELETE FROM <table>
  - DROP DATABASE <db>
- **Проверяем, где они остались**

Кейс – удаление данных в СУБД

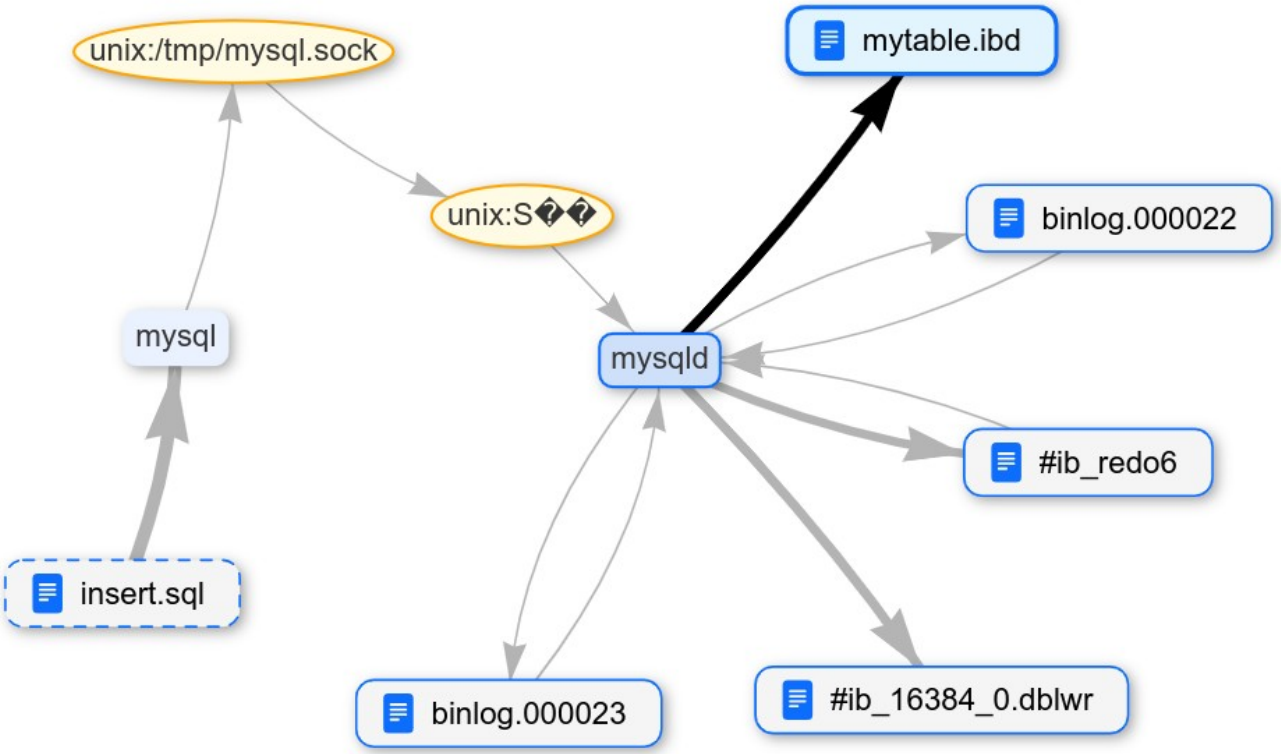
- MySQL
- MariaDB
- PostgreSQL

MySQL. Куда попадают данные?

mysql -u user -p -D my < insert.sql

- **Пометка insert.sql в Natch**
  - можно проследить путь байтов из строки

MySQL. Куда попадают данные?



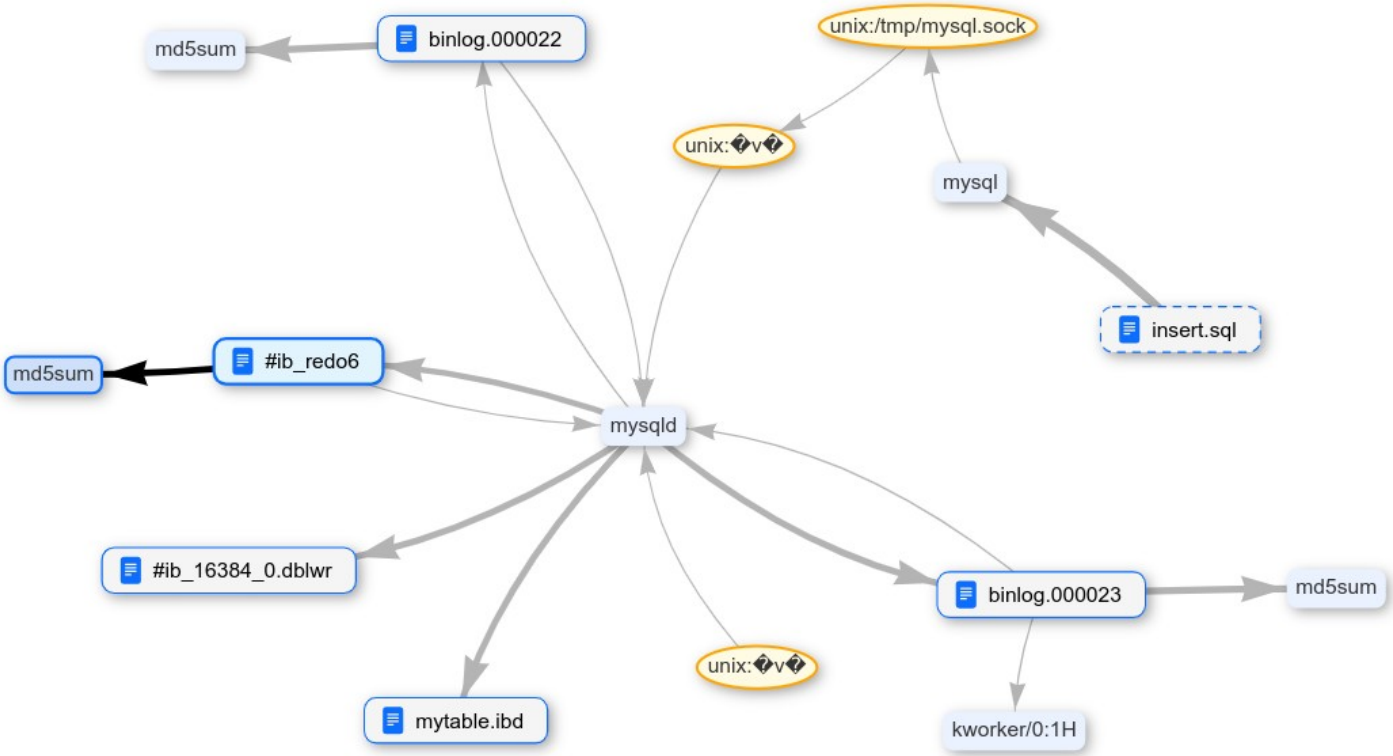
MySQL. Где остались данные после удаления?

DROP DATABASE my;

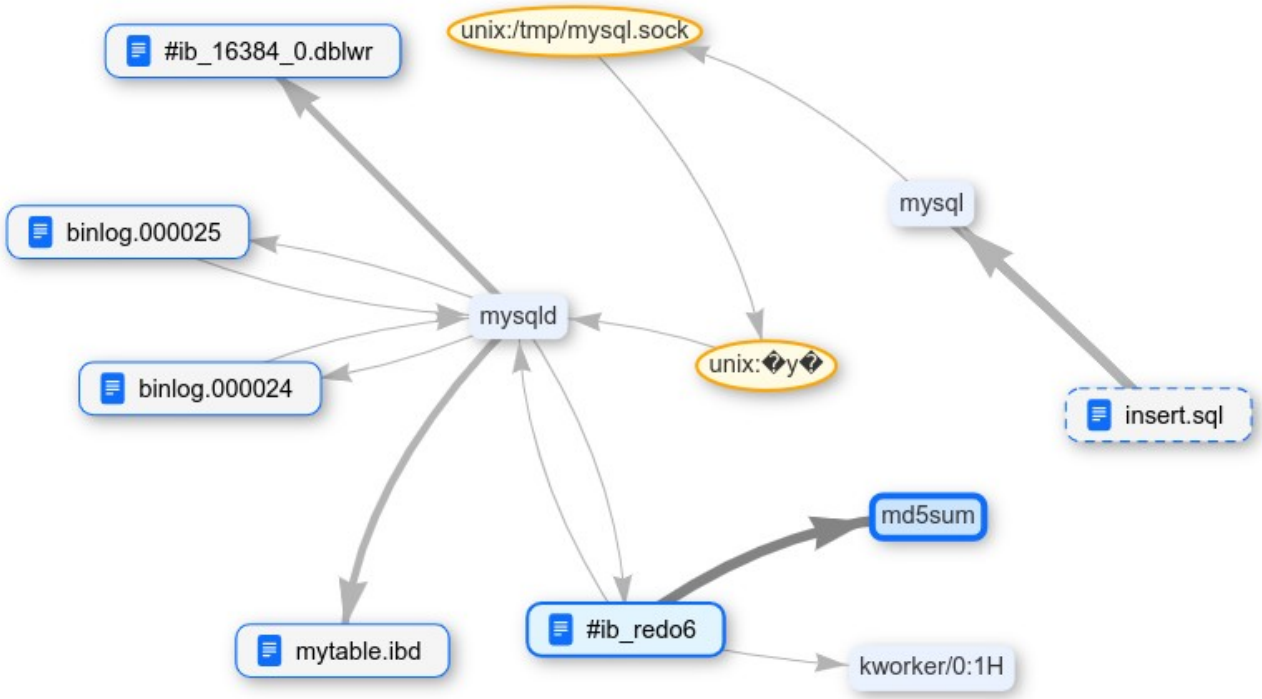
sudo md5sum <файлы с данными>

- **md5sum**
  - **читает все байты**
  - **лучше, чем find, потому что заметит отдельные байты**
  - **Natch показывает все операции чтения**
  - **Natch определит чтение помеченных данных, если файл был закеширован**

MySQL. Где остались данные после удаления?

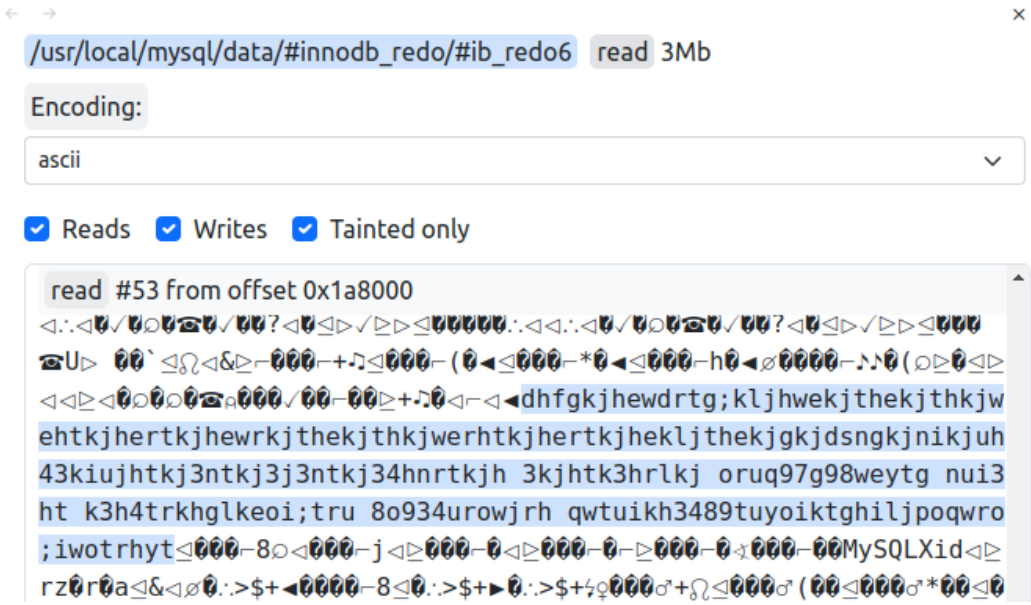


MySQL. Удаление БД



MySQL. Удаление БД

Данные остаются в /usr/local/mysql/data/#innodb_redo/#ib_redo6



MySQL. Выводы

- Данные не удаляются из внутренних логов
- И даже остаются в файловом кэше

Выводы по протестированным СУБД

- ✗ **MySQL**
- ✗ **MariaDB**
- ✗ **PostgreSQL**

- **Свободные СУБД не затирают все данные**
  - лучше всех MariaDB – хотя бы очищает память с закэшированными файлами
  - не стоит их использовать в сертифицируемых системах
- **Подробнее на хабре**
  - https://habr.com/ru/companies/isp_ras/articles/827830/

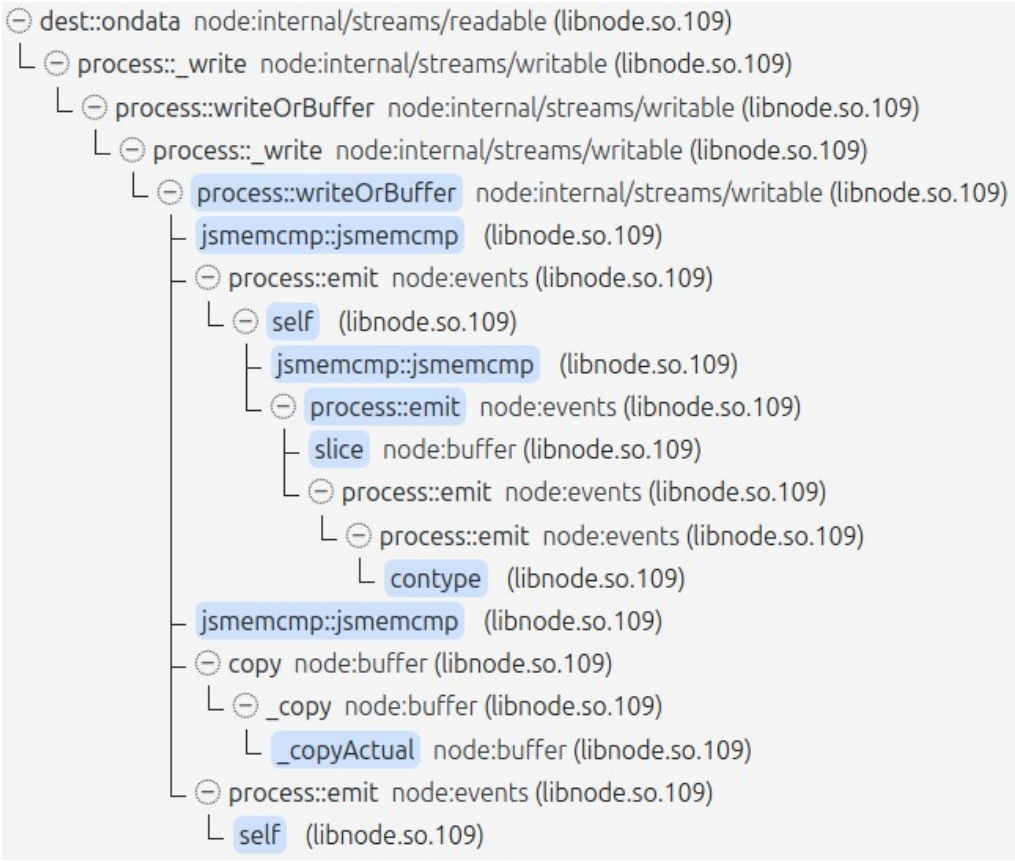
Natch 3.3

- 3 марта 2025
- Работает быстрее
- Фильтрация пакетов для пометки
- Анализ JavaScript (V8)
- Помечается расшифрованный https-трафик, если помечены входящие пакеты

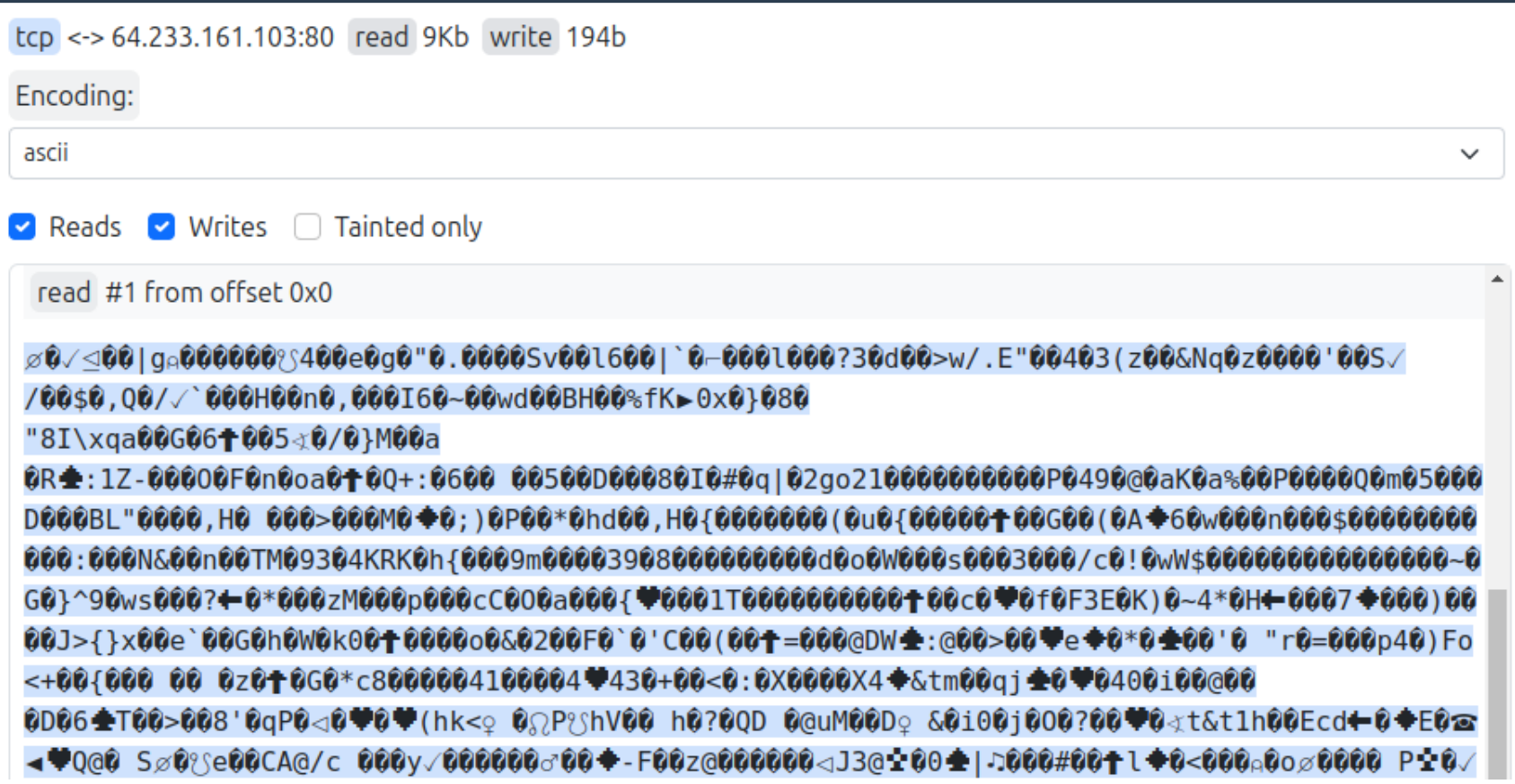
Фильтрация пакетов

- **Выбор пакетов для пометки через скрипт, создаваемый пользователем**
- **Можно помечать фрагменты пакетов**
  - например, только вводимый пароль

Анализ JavaScript



Пометка https и сжатого трафика



Пометка https и сжатого трафика

/home/user/index.html write 21Kb

Encoding:

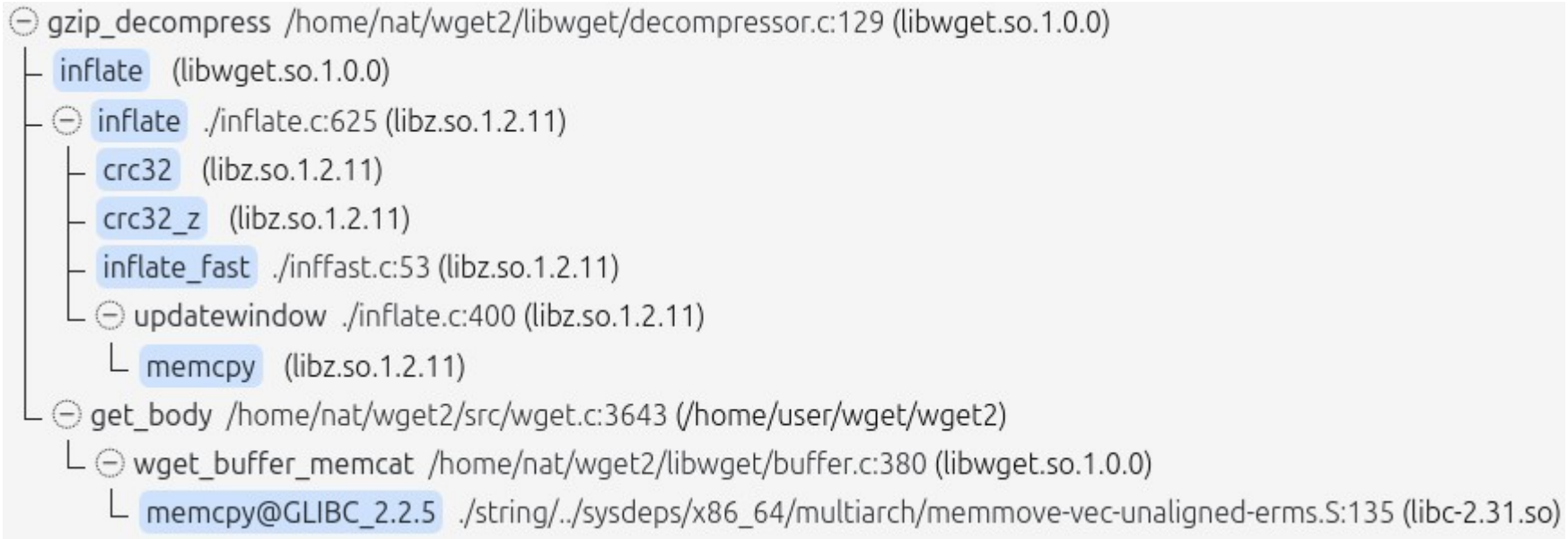
ascii

Reads Writes Tainted only

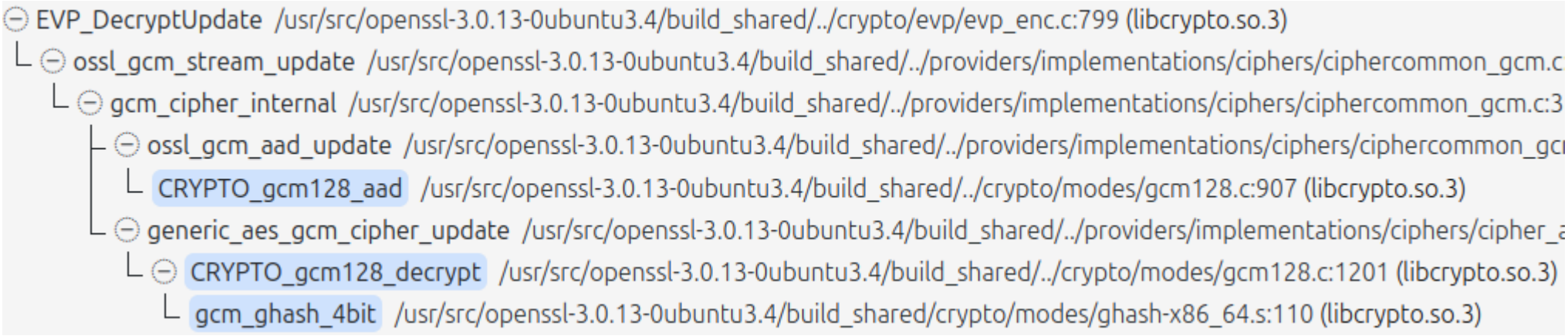
write #1 from offset 0xd20

```
ement)if(a.tagName==="A"){a=a.getAttribute("data-nohref")==="1";break a}a=!1}a&&b.preventD
efault()},!0);}).call(this);</script><style>#gbar,#guser{font-size:13px;padding-top:1px !i
mportant;}#gbar{height:22px}#guser{padding-bottom:7px !important;text-align:right}.gbh,.gb
d{border-top:1px solid #c9d7f1;font-size:1px}.gbh{height:0;position:absolute;top:24px;widt
h:100%}@media all{.gb1{height:22px;margin-right:.5em;vertical-align:top}#gbar{float:left}}
a.gb1,a.gb4{text-decoration:underline !important}a.gb1,a.gb4{color:#00c !important}.gbi .g
b4{color:#dd8e27 !important}.gbf .gb4{color:#900 !important}
</style><style>body,td,a,p,.h{font-family:sans-serif}body{margin:0;overflow-y:scroll}#gog{
padding:3px 8px 0}td{line-height:.8em}.gac_m td{line-height:17px}form{margin-bottom:20px}.
h{color:#1967d2}em{font-weight:bold;font-style:normal}.lst{height:25px;width:496px}.gsfi,.
lst{font:18px sans-serif}.gsfs{font:17px sans-serif}.ds{display:inline-box;display:inline-
```

libz


gzip_decompress /home/nat/wget2/libwget/decompressor.c:129 (libwget.so.1.0.0)
inflate (libwget.so.1.0.0)
inflate ./inflate.c:625 (libz.so.1.2.11)
crc32 (libz.so.1.2.11)
crc32_z (libz.so.1.2.11)
inflate_fast ./inffast.c:53 (libz.so.1.2.11)
updatewindow ./inflate.c:400 (libz.so.1.2.11)
memcpy (libz.so.1.2.11)
get_body /home/nat/wget2/src/wget.c:3643 (/home/user/wget/wget2)
wget_buffer_memcat /home/nat/wget2/libwget/buffer.c:380 (libwget.so.1.0.0)
memcpy@GLIBC_2.2.5 ./string/../sysdeps/x86_64/multiarch/memmove-vec-unaligned-erms.S:135 (libc-2.31.so)

libcrypto


EVP_DecryptUpdate /usr/src/openssl-3.0.13-0ubuntu3.4/build_shared/../crypto/evp/evp_enc.c:799 (libcrypto.so.3)
ossl_gcm_stream_update /usr/src/openssl-3.0.13-0ubuntu3.4/build_shared/../providers/implementations/ciphers/ciphercommon_gcm.c
gcm_cipher_internal /usr/src/openssl-3.0.13-0ubuntu3.4/build_shared/../providers/implementations/ciphers/ciphercommon_gcm.c:3
ossl_gcm_aad_update /usr/src/openssl-3.0.13-0ubuntu3.4/build_shared/../providers/implementations/ciphers/ciphercommon_gc
CRYPTO_gcm128_aad /usr/src/openssl-3.0.13-0ubuntu3.4/build_shared/../crypto/modes/gcm128.c:907 (libcrypto.so.3)
generic_aes_gcm_cipher_update /usr/src/openssl-3.0.13-0ubuntu3.4/build_shared/../providers/implementations/ciphers/cipher_a
CRYPTO_gcm128_decrypt /usr/src/openssl-3.0.13-0ubuntu3.4/build_shared/../crypto/modes/gcm128.c:1201 (libcrypto.so.3)
gcm_ghash_4bit /usr/src/openssl-3.0.13-0ubuntu3.4/build_shared/crypto/modes/ghash-x86_64.s:110 (libcrypto.so.3)

Телеграм-канал Natch

- https://t.me/ispras_natch
- Ссылки на документацию и релизы
- Вопросы от пользователей
- Разборы кейсов
- Анонсы вебинаров
- Уведомления о новых релизах
- Собственный стикерпак с нарвалами :)