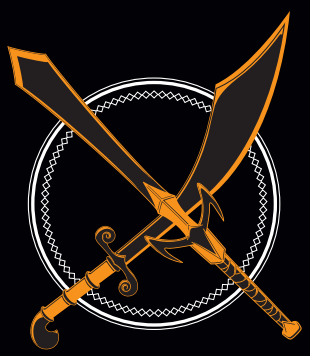
Сергей Белов
sergeybelove.ru

WORLD of WarCraft

КАК ЛОМАЛИ ПИРАТКИ

Для новеньких и для тех, кто не особо в курсе, что вообще происходило и происходит в мире World of Warcraft, сперва небольшое интро.



Все читатели слышали о World of Warcraft, а многие и играли в эту легко затягивающую игру. Я сам посвятил ей лет шесть-семь своей жизни. Примерно год я играл (с 1.12.1 и по 2.*) и в это же время начал администрировать (2007 год). К сожалению, полностью завязать не получается и по сей день, то и дело проверяю игровые и девелоперские форумы.

ИНТРО

В 2004 году Blizzard представила миру игру, которая надолго войдет в историю. Они взяли Warcraft, который уже в то время был неимоверно популярен, и сделали из него полноценную online RPG (MMORPG). Тысячи фанатов собрались в одном месте на серверах Blizzard. Модель игры была следующей: есть свободно распространяемый клиент WoW и платный доступ к серверам («офф»).

Естественно, среди игроков не обошлось без людей с навыками реверс-инжиниринга, знанием сетей, клиент-серверной архитектуры и многого другого. Довольно быстро стало понятно, что хочется свой сервер, где-нибудь в локалке, с меньшим пингом, большим фаном (позвать друзей, внести свои изменения в игру и прочее). Так и родилась идея создавать свои собственные эмуляторы официального сервера WoW — многие называют их «пиратки» или «фришки».

ПЕРВЫЕ ШАГИ, РАЗРАБОТКА

Чтобы тебе была понятна суть статьи и принципы подхода к атакам, я должен немного рассказать о разработке. Долгое время в корне клиента (вплоть до версии клиента 4.x, конец 2010 года) находился текстовый файл realmlist с незатейливым расширением wtf (warcraft text file). Там указывался IP-адрес (домен) сервера, куда конектиться клиенту. То есть перенаправить подключение клиента было очень просто, можно сказать любым своим друзьям: замените realmlist.wtf на кастомный — и хоп, они уже подключились к тебе.

И собирались программисты, сетевые инженеры, снифали пакеты при игре на официальном сервере (авторизация, вход в игру, сама игра), парсили кеш (клиент WoW кешировал данные о мире — предметы, мобы и прочее) и пытались воссоздать официальный сервер в своих кулуарах. За это время было много разных подходов к реализации сервера (Delphi, C#, Java, C++) и очень, очень много команд. В итоге самыми устойчивыми оказались две — проект MaNGOS и его форк TrinityCore, написанные в большинстве своем на C++.

Конечная картина такая: берется клиент, перенаправляется на свой собственный сервер, и на нем уже по мере сил эмулируется официальный. Многие данные брались из снифов, но и многое приходилось делать «на глаз», например — ну вот тут босс кастует пять секунд и секунду бегает по кругу. Вот такой громадный труд разработчиков, о котором можно писать отдельную статью :).

В итоге такие серверы набирали невероятную популярность, ведь их админы могли реализовать штуки, которых не было на официальных серверах (внутриигровые события, кастомные вещи, другие эффекты от заклинаний и подобное). И на многих из них был так называемый донат, где игроки могли приобрести вещи или услуги для своих персонажей. Было несколько прецедентов, когда Blizzard закрывала такие серверы. Это я о том, как в итоге далеко зашла разработка и что это не просто детские игры в реверсеров.

Ну и разумеется, в конечном счете люди начали искать способы атаки на «фришки».

DENIAL OF SERVICE

Первым по популярности можно поставить этот вид атак. Действует назло админам и позволяет повысить ЧСВ школьникам. Средний аптайм фри-сервера составляет примерно одни сутки. Порой даже ничего не надо было делать, он мог упасть сам из-за кривого кода :). Но все же чаще всего находились такие места, как призыв мобов (неигровых персонажей, «серверных созданий»), у которых не было КД (cooldown, время между повторным использованием заклинания), и можно было просто флудить, вызывая большое количество этих самых мобов. В итоге сервер ложился, админы смотрели краш-логи (если хватало знаний) и пытались зафиксить багу, пока все игроки разносили игровой форум с темами «Сервер снова лег?!».

Также часто серверы роняли и через небезызвестную утилиту LOIC (low orbital ionic cannon). Защищались обычно через iptables + connlimit.

WPE

Любой читер, кто играл в WoW (хотя не только), знает эту программу. Winsock Packet Editor (WPE) — это пакетный снифер/редактор, который в основном предназначен для читерства в онлайн-играх. WPE позволяет изменять данные на уровне TCP. Используя WPE, можно выбрать один из запущенных процессов и изменить данные до того, как они будут отправлены на сервер. Также можно просто сохранять все пакеты для последующего их анализа. Он поддерживает различные фильтры, которые можно сохранять и применять, когда понадобится. WPE часто используют для пентеста «тонких клиентов» (например, разные апплеты в браузере, работающие не по HTTP). Итак, здесь открывается целое поле для атаки!

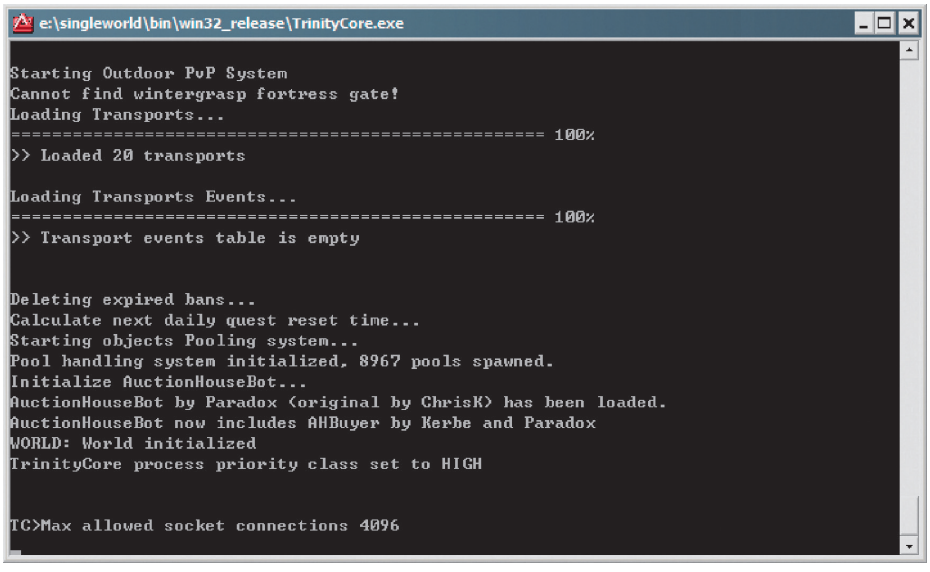
Разберем обычную ситуацию — покупку предмета у вендора. На сетевом уровне это выглядит так: игрок отправляет пакет на покупку с ID выбранного предмета, сервер смотрит, рядом ли игрок с вендором, достаточно ли денег у игрока на покупку этой вещи, и если все OK — «кладет» эту вещь ему в сумку.

Но что произойдет, если мы заменим ID покупаемой вещи? Во-первых — как его узнать? Есть база WoW — wowhead.com. Находим нужную вещь (какой-нибудь крутой элемент экипировки), смотрим в URL, там ID вещи и так же находим текущую вещь, которую продает вендор. Теперь смотрим сетевой трафик через WPE, находим ID вещи (в Hex, причем «обратной записью»), которую покупаем сейчас, и заменяем на то, что нашли на wowhead. В итоге на сервере смотрится стоимость (обычно вещи, которые нужно «выбивать» с боссов, не имели цены в базе или имели очень низкую, так как и не задумывалось, что их можно покупать), и если игрок рядом — ему отдается вещь, которой и не было в продаже у вендора :). Уязвимость успешно работала во времена патча 1.*. Потом ее логично зафиксили — проверяли список предметов, которые вообще есть у вендора.

В эту же копилку — использование заклинаний игроком. Смотрим ID спелла (заклинания), который используем сейчас, подменяем на ID спелла какого-нибудь босса с большим дамагом — в итоге ходим и кастуем как босс :). Это работало вообще очень долго и было исправлено вроде только в WotLK (патч 3.*).

Фильтры для WPE или программы, автоматизирующие подмену пакетов, обычно легко находились на читерских форумах в свободном доступе.

И по этой же схеме работали всякие speed hack'и, teleporter'ы и тому подобное. Пространство для «обмана» сервера было велико, античиты в то время были в основном на серверной стороне (если вообще были) — всякие AC, AC2. У Blizzard'а на официальном сервере был (и есть) Warden, который работает по принципу анализа запущенных процессов на клиенте, снимает сигнатуры и отправляет на сервер (игроки даже пытались раздуть судебный процесс на этой почве). Его реализация на эмуляторах появилась сравнительно недавно. Нельзя не упомянуть о платных античитах для клиентов, играющих на эмуляторах, но они стоят денег, и обычно серверная часть только под Windows, когда чаще всего сервер WoW работает под Linux (конечно, есть wine, но это уже костыли, да и игроков бывает по несколько тысяч на сервер).



Консоль сервера, эмулятора TrinityCore

WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.

ЗАМЕНА ФАЙЛОВ КЛИЕНТА

Похожим способом эксплуатации уязвимостей с подменой пакетов была подмена файлов на клиенте. Стоит сделать отступление и сказать, что многие файлы на сервере и клиенте были связаны один в один (карты, DBC-файлы — информация о спеллах, музыке и других вещах), в том числе и MPQ-файлы. Можно было их распаковать и найти в них много интересного. Начали появляться измененные MPQ-файлы, где были среди прочего заменены ID спеллов. При их использовании на сервер отправлялся другой пакет.

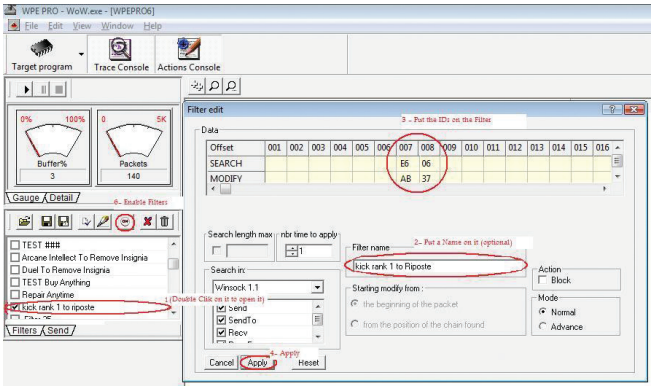
WEB?!

Конечно же, любой уважающий себя сервер WoW должен был иметь сайт. Логично, что, допустив ошибку в коде на сайте, можно было получить некоторый доступ к базе. Чаще всего базой для сервера WoW выбиралась всем известная MySQL, то есть к одному серверу БД подключался и сайт (где был скрипт регистрации, мониторинга и прочее), и сервер. На сайтах, посвященных серверам WoW, можно было найти различные поделки от энтузиастов. Обычно это просто рипнутый дизайн с официальной странички WoW Blizzard и свой серверный код (в большинстве своем PHP). Были и просто ужасные подделки, которые хорошо работали, но я сам порой находил в них за 5–10 минут SQL-инъекции и репортил разработчикам (при этом, конечно, не используя вообще их скрипты). Но были и разработки от действительно хороших программистов (надо сказать, что среди community встречалось очень много талантливых людей с опытом, в том числе разработчики из банков в Швейцарии и не только). И, по моему скромному мнению, одна из самых серьезных работ — это скрипт armory от Shadez. Армори (оружейная) — это обширная база данных с прозрачным и удобным интерфейсом, по которой можно производить поиск. Все данные поступают напрямую из игровых миров, поэтому в армори можно найти самую полную и свежую информацию о персонажах, командах Арены, гильдиях, предметах и наградах для фракций World of Warcraft. И написан скрипт правда хорошо, практически с полной функциональностью официального армори, с применением ООП, шаблонизатора и так далее. И у него была недописанная админка с… SQL-инъекцией, которая всплыла довольно поздно, когда скрипт уже был установлен на множестве серверов. Наверное, это один из самых громких и массовых взломов серверов WoW через веб.

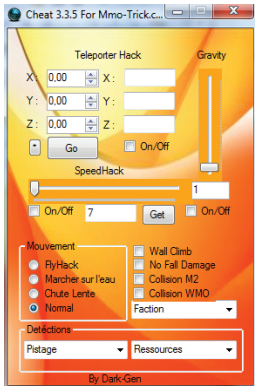
АТАКИ ИЗ КЛИЕНТА

А как насчет атаки на сервер прямо из клиента WoW? Ведь клиент отправляет данные, сервер сохраняет их в базе… SQL-базе. Тикет https://github.com/TrinityCore/TrinityCore/issues/4287, следующий код был на серверной стороне

```
bool ObjectMgr::AddGameTele(GameTele↵
& tele)
{
  ...
WorldDatabase.PExecute("INSERT INTO ↵
game_tele (id, position_x, position_y,↵
position_z, orientation, map, name) ↵
VALUES (%u, %f, %f, %f, %f, %d, '%s')",↵
new_id, tele.position_x, tele.↵
position_y, tele.position_z, tele.↵
orientation, tele.mapId, tele.name.↵
c_str());
  ...
}
```



Запущенный WPE и выбор фильтров



Пример читерской программы, использующей подмену пакетов

Данная команда (по дефолту) требовала некоторый уровень доступа (что-то типа модератора). Она добавляла место для телепорта на текущую позицию с заданным именем. Стоишь в данной точке карты, пишешь в чат .tele add namefortele, а потом просто в нужный момент .tele namefortele и оказываешься на этой позиции. Как можешь заметить, все переменные, кроме последней, рассчитывались на серверной стороне. И последнюю переменную сервер должен был принять и положить в базу от клиента. Из-за отсутствующего экранирования спецсимволов была возможна атака с вектором: `.tele add namefortele') %Injection_Here% -- - !` Не правда ли, это не так привычно — использовать SQL-инъекцию не через браузер/Burp/sqlmap? :) Фикс был логичен, добавить escape, присущий текущему подключению к базе:

```
std::string safeName(tele.name);
WorldDatabase.escape_string(safeName);
```

и заменив в запросе

```
tele.name.c_str()
```

на

```
safeName.c_str()
```

НЕТИПИЧНЫЕ РАСШИРЕНИЯ

WoW — легко расширяемая игра. В ней возможно «законное» подключение аддонов, написанных на Lua. В итоге было множество реализаций для автоматического выполнения некоторых действий (например, рыбалки. Кинул удочку, кликнул в нужный момент, вытащил рыбу). Довольно много таких аддонов можно было просто найти в Сети.

ЧТО ДЕЛАЛИ ЕЩЕ?

Дюпали вещи (копирование вещей в рюкзаке, через WPE), находили бажные квесты (берешь дейли-квест, оказывается, что ничего делать не надо, сдаешь. И так каждый день), находили бажных мобов (с малым количеством здоровья / малой атакой) и разные другие вещи, но это уже не совсем технические аспекты атак, которые мы рассматриваем.

ОТСТУПЛЕНИЕ

Можно рассказывать еще очень много, в том числе о самой разработке с точки зрения security. Например, когда Blizzard сделала ход конем. В момент, когда WoW достигла своей, наверное, максимальной популярности (конец патча WotLK, переход на Cataclysm), разработчики начали с каждым патчем (которые выходили примерно раз в месяц-три) рандомно менять опкоды между клиентом и сервером, что полностью рушило уже имеющийся, наработанный за несколько лет и постоянно пополняемый протокол. Мало того что надо было реализовывать серверную часть и наполнение контентом, так еще и каждый раз анализировать новые опкоды и обновлять их на сервере. И плюс они убрали возможность свободно менять realmlist.wtf. Да, это исправлялось небольшим патчем EXE-файла, но на этой почве начались разногласия между разработчиками — до этого, все эти годы, не надо было изменять клиент (бинарную структуру), что никак не нарушало лицензию Blizzard. Плюс официально запретили снифать трафик между сервером и клиентом, извлекать данные из кеша и другое (да, об этом никто бы и не узнал, но снова — нарушение лицензии). Это и многое другое затормозило разработку подобных эмуляторов с отставанием примерно на год, но потом все же появились инструменты (с открытым кодом) для автоматизации парсинга опкодов и обхода других костылей. Но провал в разработке случился громадный. Кстати, из забавного. Порой в опкодах можно было найти «приветы» реверсерам от разработчиков WoW :).

Или, когда выходила Diablo 3 и еще не было официального релиза, а уже был эмулятор для бета-клиента (Mooege) с более-менее работающим протоколом и основной механикой, Blizzard просто взяли и закрыли «по-хорошему» эмулятор, договорившись об удалении репозитория на GitHub, всех веток форумов и роспуске команды. Они уже знали, насколько могут быть фанатичны и игроки, и комьюнити. Опыт нескольких лет с World of Warcraft.

FIN

Надеюсь, тебе было интересно немного окунуться в другой мир — целую эпоху для open source комьюнити и для многих игроков (и вынести ценную инфу о взломе и защите онлайн-игр). Спасибо тебе за прочтение и спасибо всем тем, кто узнал меня в этой статье :). Были просто потрясающие времена, с тысячами игроков, и это было по-настоящему круто.