атака на VPN или жылдыз твой интернет

крис касперски ака мыщъх

виртуальные частные сети (они же Virtual Private Network или сокращенно VPN) продвигаются как идеальное средство для передачи конфиденциальных данных через Интернет или беспроводные сети, неподвластное хакерским атакам; на самом деле в действительности все обстоит совсем не так…

введение

Лет эдак тридцать назад, когда Интернет только-только зарождался, обмен конфиденциальными данными осуществлялся через выделенные каналы и X.25 сети, построенные на основе обычных телефонных сетей. Так же использовалось прямое модемное соединение, радиорелейная связь и прочие телекоммуникационные средства. Высокая степень защищенность "компенсировалась" столь же высокой ценой и смехотворной скоростью передачи.

Рисунок 1 комплекс оборудования для радиорелейной связи и внешняя антенна к нему же

С развитием криптографии все изменилось. Появилась возможность передавать зашифрованные данные через открытые сети, заведомо подверженные перехвату. Для этого в них пробиваются так называемые виртуальные туннели (virtual tunnel или piggy-back). Сам механизм передачи не изменился. К нему всего лишь добавилось шифрование. Но разве шифрование не использовалось раньше? В чем же революционность предложенной технологии? А в том, что традиционная симметричная криптография требует предварительной передачи секретного ключа, которым получатель будет расшифровывать текст. Следовательно, необходим защищенный канал связи. А такого канала чаще всего нет. Если же он есть, то можно не заморачиваться с шифрованием, а передать текст напрямую. Имеются и другие проблемы. Ключи нужно как можно чаще менять, причем необходимо защищаться не только от вскрытия шифра, но и от навязывания ложных паролей. Короче, сплошной геморрой.

Рисунок 2 тоннели – виртуальные и реальные

Виртуальные сети этих проблем лишены. Они действительно преобразили мир, освободив транснациональные корпорации от необходимости развития собственной инфраструктуры змеящихся кабелей. И хотя Microsoft отчаянно проталкивала их в малый бизнес, шестеренки вращались со скрипом и дальше рекламной шумихи дело не шло. Для удаленной работы с офисом защищенности обыкновенных Интернет-каналов вполне достаточно, а уж про "домашние" локальные сети речь и вовсе не идет.

А вот в беспроводных сетях, технология VPN оказалась как нельзя кстати. Радиоволны не знают границ и перехватываются на значительных расстояниях. Легкость взлома многократно усиливает требования к защищенности. Любой паренек, вооруженный сниффером, может посмотреть пароль на ящик и шутки ради изменить его, не говоря уже про чтение личной переписки. Грязными лапами в чужую душу. А VPN на что? Поставим его и будем чувствовать себя как за каменной стеной… Или она только с виду каменная, а в действительности это просто картон? Давайте разбираться!

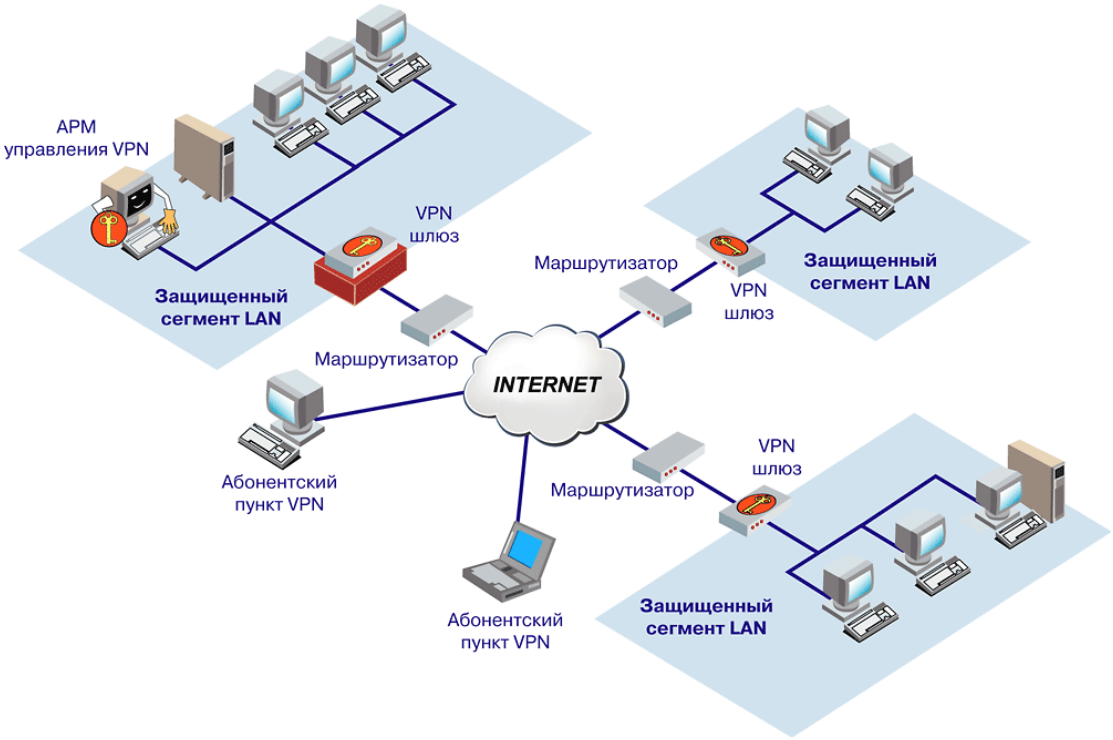

Рисунок 3 типичная VPN сеть с высоты птичьего помета

архитектура VPN

VPN представляет собой внушительное сооружение, использующее десятки разнообразных протоколов, гоняющих пакеты данных, словно кровь по артериям. Полный анализ архитектуры VPN не входит в нашу задачу (тем более, что она постоянно развивается и

совершенствуется). Взломщику, грабящему банк, совершенно необязательно знать расположение унитазов и держать в голове все изгибы канализационных труб. Хотя эта информация и будет очень полезной при планировании путей отступления, большинство ограниваются изучением основных маршрутов: двери — касса — сейф. Так же поступим и мы.

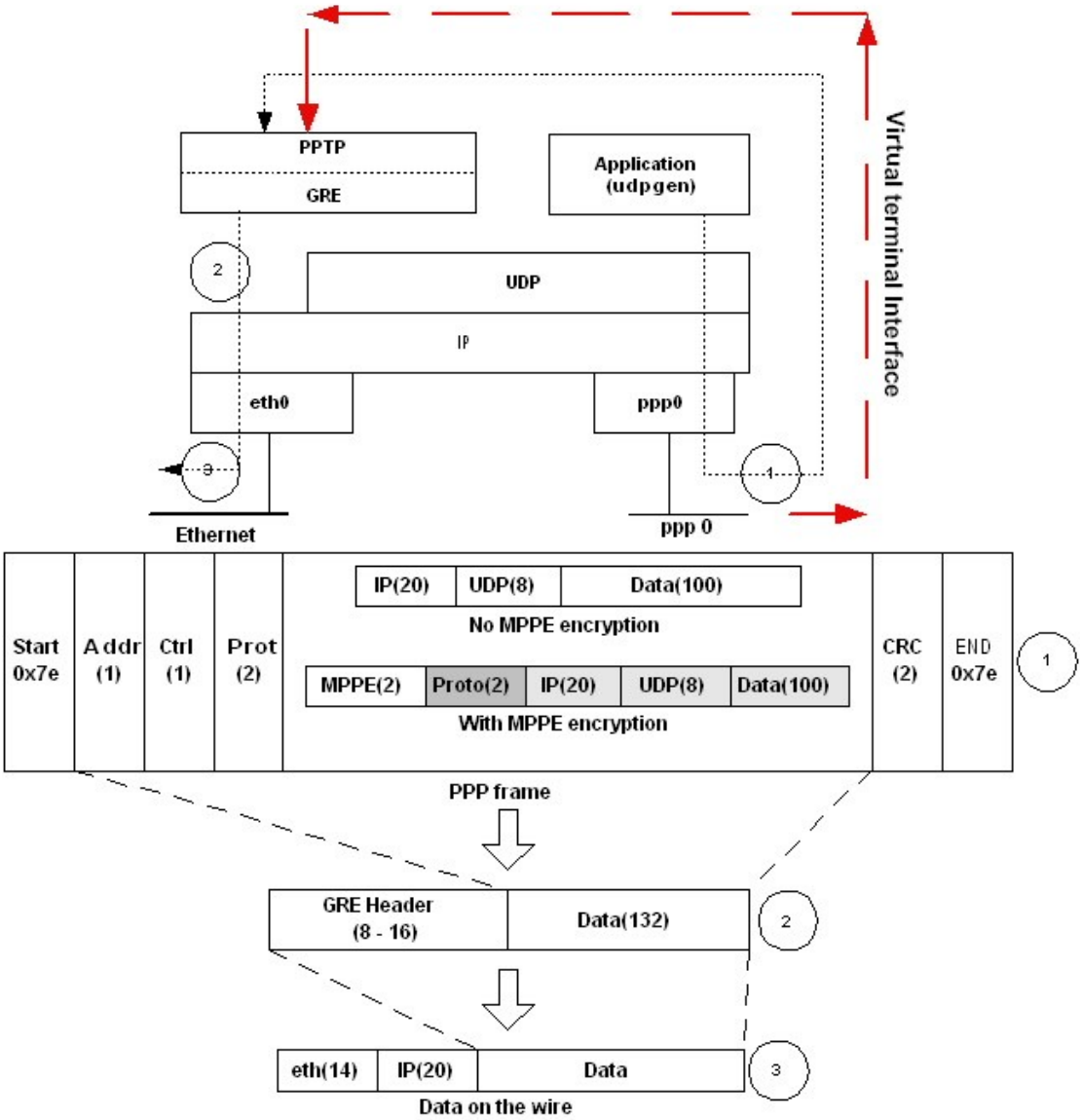

Рисунок 4 схематичная реализация VPN поверх Ethernet

внутри PPTP

Протокол PPTP (Point to Point Tunneling Protocol — Туннельный Протокол Точка — Точка) это основной протокол для организации VPN сетей под Windows. Он связывает две системы виртуальным каналом связи — тоннелем. На одной стороне сидит клиент (client), на другой — сервер (server), в результате чего обе части получаются сильно неравнозначны.

А что находится внутри тоннеля? Для поддержания своей работоспособности, сервер открывает 1723 TCP-порт, на котором висит система жизнеобеспечения, она же Control Connection — служебный канал связи, связывающий сервер с клиентом (изначально использовался 5678 порт, но IANA — Internet Assigned Numbers Authority организация, ведающая регистрацией доменов и портов, — решила иначе и утвердила за PPTP-сервером 1732 порт, список остальных зарегистрированных портов можно найти на http://www.iana.org/assignments/port-numbers). Клиентский порт может быть любым и выбирается им (а точнее его осью) самостоятельно. Никакой аутентификации здесь не

требуется, что открывает простор для всевозможных махинаций, например, хакер может принудительно закрыть чужую сессию.

Служебный канал в основном используется для управления скоростью поступления трафика, избегая холостых простоев и предотвращая "заторы". Это осуществляется путем рассылки специальных PPTP-сообщений (PPTP Control Connection message). Каждое такое сообщение начинается с заголовка, а заканчивается "производственной" информацией. Длина заголовка фиксирована и составляет 8 октетов. Тело сообщения включает в себя: поле длины (Total message length), тип сообщения: служебное или управляющее сообщение (Control Message/Management Message) и магическую последовательнсоть 4Dh 3C 2C 1Ah, по которой его можно легко идентифицировать в общем потоке трафика и которую использует программа deceit.c (http://packetstormsecurity.nl/new-exploits/deceit.c) для грабежа паролей, то есть не паролей, а их хэшей, с которыми мы разберемся позднее, а пока вернемся к Microsoft и ее баранам.

Сам тоннель не использует TCP и работает исключительно на IP уровне с использованием протокола инкапсуляции GRE (Generic Encapsulation Protocol — Общий Протокол Инкапсуляции). Это вполне самостоятельный протокол, никак не зависящий от PPTP, представленный двумя документами RFC 1701 RFC 1702 (однако, Microsoft использует собственные расширения, известные под именем GRE v2). Передаваемые данные разбиваются на пакеты и передаются по IP по протоколу 47, закрепленном на GRE. "Протокол" в данном случае — это номер поле protocol IP-пакета. Не путать его с портом! В IP нет никаких портов. Они реализованы в протоколах верхнего уровня — TCP и UDP. GRE это просто еще один протокол поверх IP.

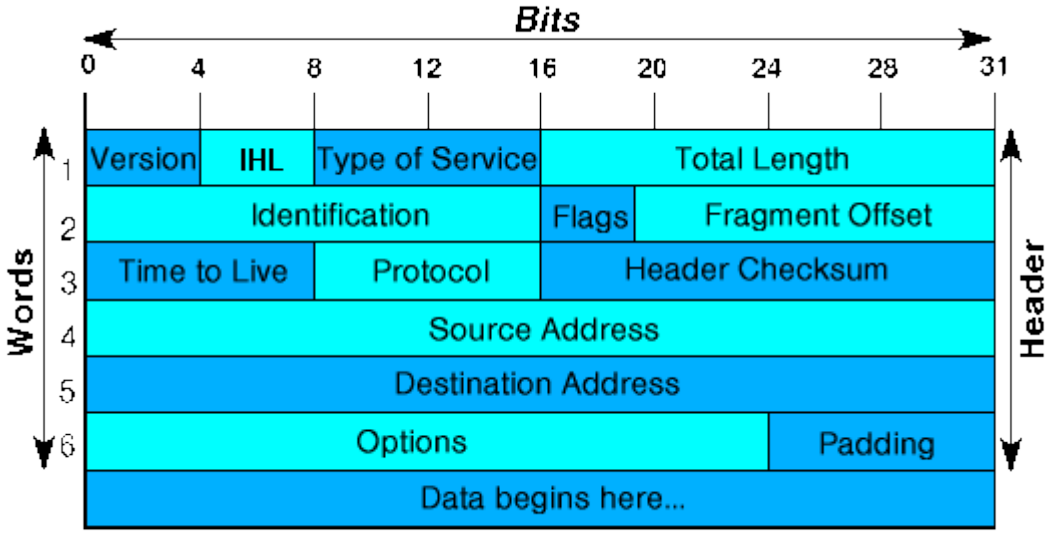

Рисунок 5 поле номера протокола в заголовке IP-пакета

Внешне GRE очень поход на TCP, в нем есть понятия скользящих окон (sliding window), сегментов (segments), номеров последовательности (sequence number) и номеров подтверждения (acknowledgement number). В практическом плане это означает, что непосредственный спуффинг PPP-пакетов невозможен. Если мы попытаемся навязать серверу (или клиенту) подложный пакет, произойдет рассинхронизация сессии и соединение окажется нарушено. В локальных сетях проблема решается посылкой 2^32 пакетов. Поскольку, поле sequence number занимает 32-бита, происходит его переполнение и значение счетчика восстанавливается. Но для атаки через Интернет это потребует немыслимого количества времени. Ситуация кажется безнадежной, но кое-какая лазейка все-таки есть.

0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1					
C		K	S	Reserved0	Ver	Protocol Type
Checksum (optional)						Reserved1 (Optional)
Key (optional)						
Sequence Number (Optional)						

Рисунок 6 заголовок GRE-пакета с S-флагом

В заголовке GRE-пакетов присутствует специальный флаг Sequence Number Present (бит 3), условно обозначаемый латинской буковой "S". Если он равен нулю, то номер последовательности признается недействительным и принимающая сторона должна его игнорировать. Во всяком случае, так говорит Стандарт. Естественно, конкретные реализации могут существенно отличаться. Тем не менее потенциальная дыра все-таки есть. Кстати говоря, Стандарт не дает никаких указаний как обрабатывать дубликаты (пакеты с одинаковым номером последовательности), оставляя это на совести конкретных реализаций. Принимающая сторона может либо отбросить один из пакетов или затребовать его повторную передачу. В обоих случаях рассинхронизации не происходит, то есть получается как бы самосинхронизующийся протокол.

Поверх GRE реализованы протоколы аутентификации и шифрования (а, при необходимости еще и сжатия, за которое чаще всего отвечает MPPE). Microsoft поставляет довольно большой ассортимент различных проколов, перечисленных в таблице 1 так что клиенту с сервером есть из чего выбирать! При желании шифрование можно и вовсе отключить, а аутентификацию осуществлять "прямым" текстом, но это будет слишком недальновидное решение, полностью обесценивающее идеологию VPN и открывающую двери для всех желающих. Нормальные администраторы так не поступают, предпочитая использовать более защищенные MS-CHAP и LANMAN Hash.

На самом деле, их защищенность сильно преувеличена и они уже давно взломаны. Подробнее об этом можно прочитать в моей "Технике сетевых атак", электронную копию которого можно бесплатно скачать с мыщъхиного ftp – nezumi.org.ru (nezumi по-японски мыщъх). К тому же, шифруются только пакеты с данными (DATA packets). Служебные протоколы, составляющие весьма значительную часть PPP-трафика, такие, например, как LCP — Link Control Protocol (Протокол управления каналом) – остаются незашифрованными со всеми вытекающими отсюда последствиями.

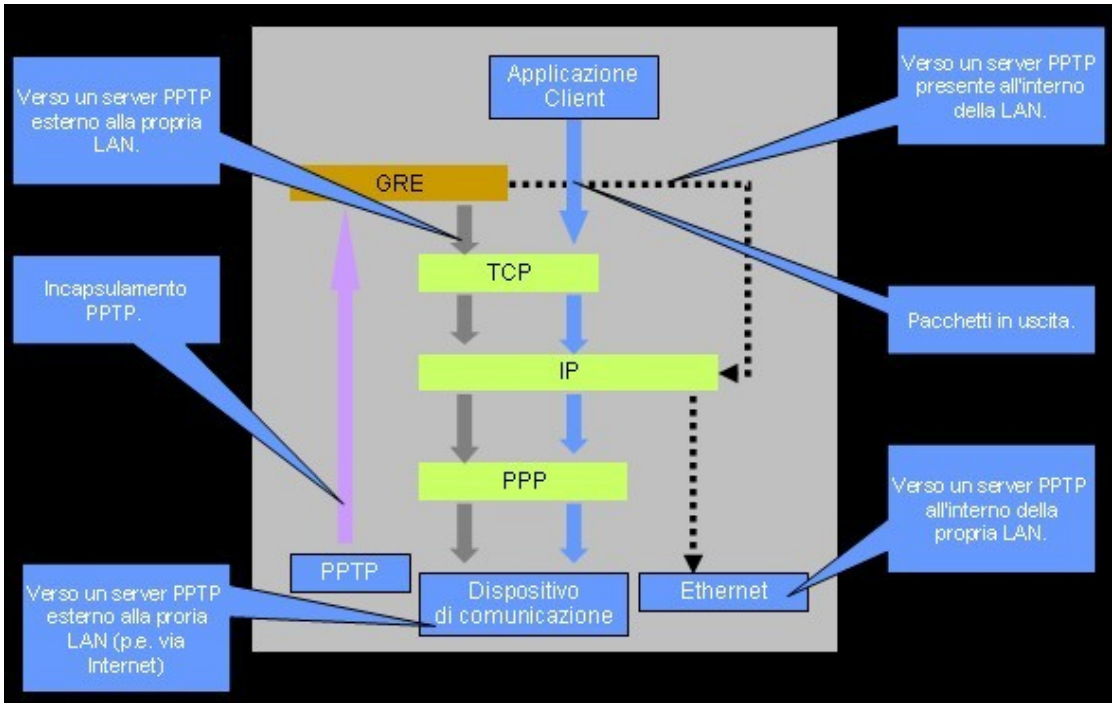

Рисунок 7 стек VPN

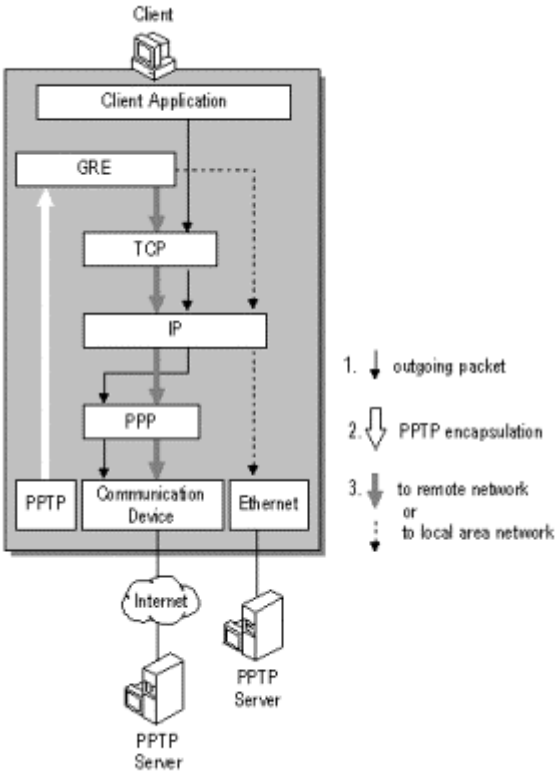

Рисунок 8 стек VPN

метод	назначение
Clear text password	аутентификация открытым текстом
LANMAN hashed password	алгоритм вычисления хэша
NT Encryption hashed password	алгоритм вычисления хэша
Challenge/Response MSCHAP version 1	алгоритм аутентификации
Challenge/Response MSCHAP version 2	алгоритм аутентификации

Таблица 1 методы хэширования и аутентификации используемые в VPN

аутентификация снаружи и изнутри

Аутентификация осуществляется либо отрытым тестом (clear text password), либо по схеме запрос/отклик (Challenge/Response). С прямым текстом все ясно. Клиент посылает серверу пароль. Сервер сравнивает это с эталоном и говорит "пошел на xyz" или "добро пожаловать". Хакер, вооруженный сниффером, может легко перехватить открытый пароль и защита пойдет лесом. Нам этот случай не интересен, поэтому не будем на нем останавливаться. К тому же открытая аутентификация в живой природе практически не встречается.

Схема запрос/отклик намного более продвинута. В общем виде она выглядит так:

- клиент посылает серверу запрос (request) на аутентификацию;
- сервер возвращает случайный отклик (challenge);
- клиент снимает со своего пароля хеш, шифрует им отклик и передает его серверу;
- то же самое проделывает и сервер, сравнивая полученный результат с ответом клиента;
- если зашифрованный отклик совпадает, аутентификация считается успешной;

Таким образом, для аутентификации знать оригинальный пароль вообще не нужен, достаточно угадать/перебрать/подсмотреть его хэш, однако, хэш не передается в открытом виде по сети и эта схема позиционируется как устойчивая к перехвату, что есть очень хорошо. А вот

ее недостатки: исходный хэш должен как-то попасть на сервер, поэтому для передачи пароля необходим защищенный канал. Это раз. Процедура аутентификации уязвима для brute-force атаки. Перехватив исходный и зашифрованный challenge, мы можем попытаться подобрать ключ шифрования, перебирая столько вариантов, сколько захотим. Ни сервер, ни клиент в этой затее никак не участвуют и не могут нам помешать. Это два. Стойкость системы определяется по формуле min(stelen(passwd),sizeof(hash)). Если хэш короток, то длина пароля не играет никакой роли и взлом может завершиться за очень короткое время. Это три. Но это все голая теория. Посмотрим, как обстоят дела на практике.

Microsoft Windows поддерживает два типа хэшей: "родной" NT-хэш и хэш LAN Manager, доставшейся ей в наследство от OS/2 (давным-давно эти системы шли вместе) и благополучно доживший до наших дней, несмотря на свою катастрофическую незащищенность. Как он рассчитывается? А вот так!

- клиентский пароль преобразуется в 14-байтовую ASCII-строку (более длинные пароли усекаются, а более короткие дополняются нулями);
- все символы приводятся к верхнему регистру;
- 14-символьный пароль разбиваются на две 7-сиивольные половинки;
- каждой 7-символьной "половинкой" зашифровывается постоянная константой AAD3B435B5140EEh по алгоритма DES;
- образуются две 8-байтовые строки;
- эти строки "склеиваются" друг с другом образуя 16-байтовый хэш;

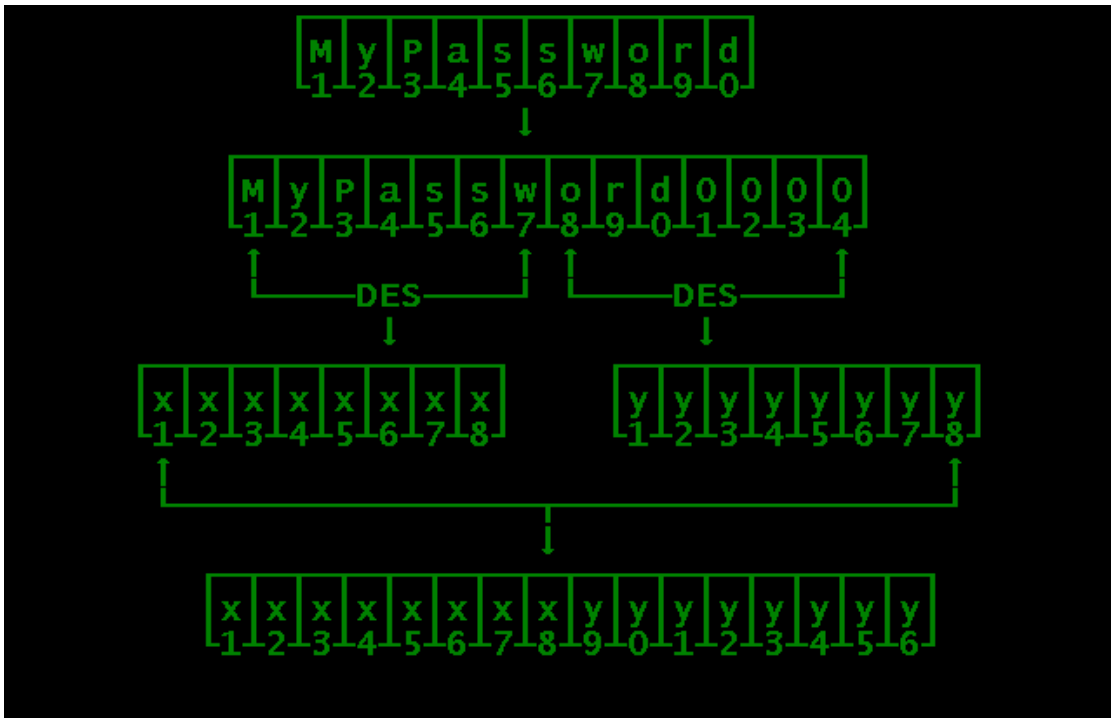

Рисунок 9 процедура вычисления LM-хэша

Независимое хеширование половинок пароля, в 1 000 000 000 000 000 раз уменьшает количество попыток, требующихся для его перебора (а вовсе не в два раза, как это может показаться на первый взгляд). Это же какой талант надо иметь, чтобы допустить такой ляп! Уж сколько раз твердили миру (то бишь разработчикам) – не разводите самодеятельность, используйте проверенные временем алгоритмы, да только все не впрок. Ситуация усугубляется тем, что алгоритм DES не требует громоздких вычислений и потому на современных процессорах LM-хэш может быть взломан за короткое время. Какое — не имеет значения. Ведь в открытом виде он все равно не передается. Правда, подобранный пароль может пригодится при входе в систему с клавиатуры.

А вот так вычисляется NT-хэш:

- в зависимости от настроек системы клиентский пароль преобразуется либо 14-символьной (по умолчанию), либо к 128-символьной ASCII строке, причем,

большинство администраторов используют длину по умолчанию, не утруждаясь ее поменять;
- ASCII-строка преобразуется в UNICODE;
- вычисляется 16-байтовый кэш по алгоритму MD4;

Как видно, NT-хэш намного более стоек, поэтому в случае с NT-хэшем хакеры предпочитают перебирать сам хэш, а с LM-хэшем — исходный пароль.

Теперь рассмотрим как работает процедура аутентификации. Обычно она осуществляется либо по протоколу MS-CHAP v1, либо по MS-CHAP v2. Начнем с первого из них, от работает так:

- клиент посылает запрос на аутентификацию VPN серверу, открыто передавая свой login
- сервер возвращает 8-байтовую случайный отклик;
- клиент снимает со своего пароля LM-хэш и генерирует три DES ключа;
- каждый из этих ключей зашифровывает отклик и получается три 8-байтовых строки;
- три 8-байтовых строки объединяются в одну 24-байтовую, которая передается серверу;
- сервер, извлекает из своей базы хэш данного клиента и расшифровывает строку;
- если результат расшифровки совпадает с исходным откликом, все ок и наоборот;

А вот так работает MS-CHAP v2:

- клиент посылает запрос на аутентификацию VPN серверу, открыто передавая свой login
- сервер возвращает клиенту 16-байтовый случайный отклик;
- клиент генерирует 16-байтовый PAC (Peer Authenticator Challenge – Равный Отклик Аутентификации);
- клиент объединяет PAC, отклик сервера и свое user name в одну строку;
- с полученной строки снимается 8-байтовый хэш по алгоритму SHA-1 и посылается серверу;
- сервер извлекает из своей базы хэш данного клиента и расшифровывает его ответ;
- если результат расшифровки совпадет с исходным откликом, все ок и наоборот;
- в последствии сервер берет PAC клиента и на основе хэша генерирует 20-байтовый AR (Authenticator Response – Аутентификационный Ответ), передавая его клиенту;
- клиент проделывает туже самую операцию и сравнивает полученный AR с ответом сервера;
- если все совпадает клиент аутентифицирует сервер;

Как видно, протокол MS-CHAP v2 выглядит более защищенным, чем MS-CHAP v1, однако, оба они уязвимы. Подробности можно найти в статье Брюса Шнайера "Cryptanalysis of Microsoft's MS CHAP v2" (http://www.schneier.com/paper-pptpv2.html), которая вопреки своему названию, описывает не только MS-CHAP v2, но и MS-CHAP v1.

Рисунок 10 Брюс Шнайер позирует с ноутбуком

Чем плох MS-CHAP v1? Тем, что его уже давно научились ломать. В сети можно найти множество готовых "отмычек", работающих в полностью автоматизированном режиме и не требующих никакой квалификации. Скачал — запустил — поимел. Самая известная (и самая древняя!) это L0phtcrack, но сейчас проект сдулся и переименован в LC 5

(http://www.securityfocus.com/tools/1005), а на прежнем адресе (http://www.l0phtcrack.com/) сейчас висит объявление о продаже. Тем не менее, L0phtcrack по-прежнему остаются в строю, поскольку в отличии от жаждущего денег LC 5, он распространяется бесплатно. Найти его можно на любом хакерском сайте.

Рисунок 11 один из рабоботчиков L0phtcrack

На Pentium-4 с таковой частотой 3 GHz, полный цикл перебора занимает не более 4 часов, а в среднем пароль подбирается за 2 часа. Вот тебе бабушка и безопасность! И это еще не предел! По сути L0phtcrack представлял собой тривиальный переборщик, работающий по принципу грубой силы (скажут копать — буду копать). Это неэлегантно и непроизводительно.

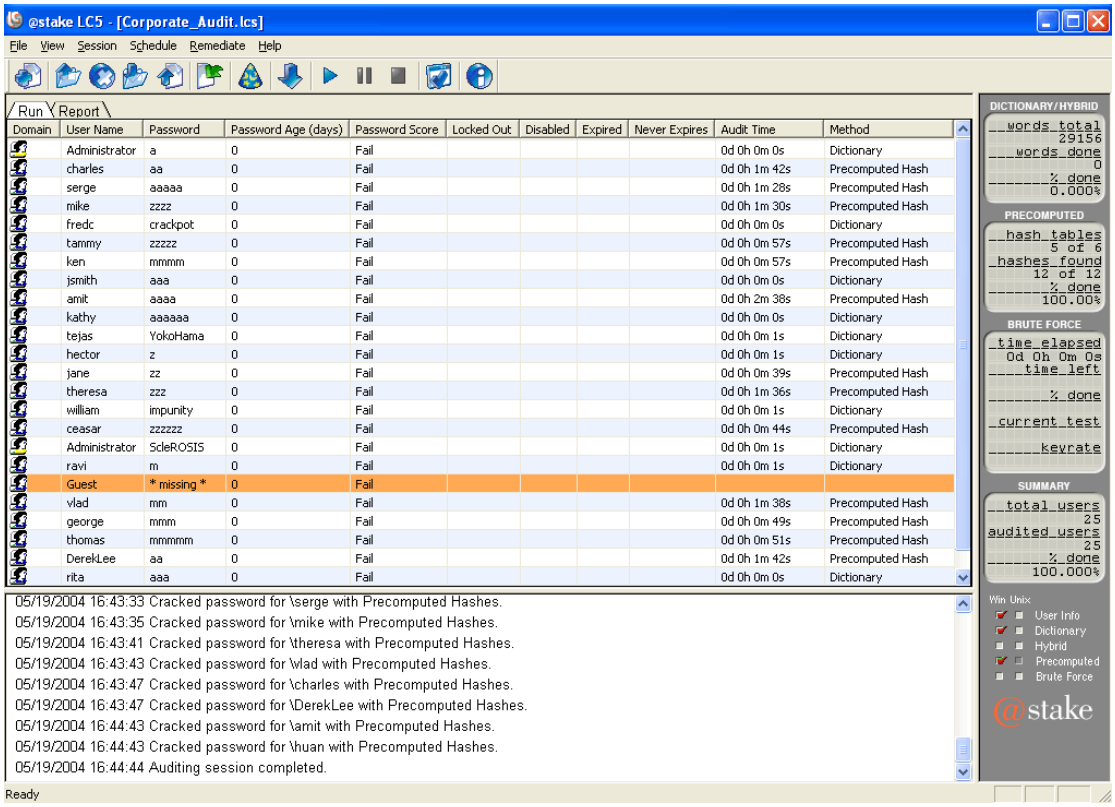

Рисунок 12 LC5 за работой

В начале 2004 года 20-летний шведский хакер Филипп Ошлин (Philippe Oechslin), ведущий научный ассистент лаборатории криптографии и защиты Швайцарского государственного технологического института в Лозанне, сообщил о принципиально новом методе ускоренного взлома (Faster Time-Memory Trade-Off Technique), основанном на предвычисленных таблицах, содержащих всех возможные комбинации символов в пароле. Отталкиваясь от работ Мартина Неллмана (Martin Hellman), Филлипп переработал и усилил алгоритм подбора паролей, "адоптировав" его к алгоритму LM-менеджера. Время взлома сократилось в ~12 раз и на AMD 2500+ с 1,5 Гбайтами ОЗУ составило… всего 13,6 секунд! Ну прямо взлом в реальном времени! А ведь это не самый мощный компьютер из всех доступных. (Объем памяти важен потому, что предвычисленные таблицы очень велики, а интенсивный своппинг на диск съедает всю производительность).

Рисунок 13 Филлип Ошлин

Все подробности можно найти в статье Филиппа, выложенной на Швейцарском сайте (http://lasecwww.epfl.ch/php_code/publications/search.php?ref=Oech03) и уже реализованных в

программе Rainbow Crack (http://www.antsight.com/zsl/rainbowcrack/), демонстрационная версия которой распространяется бесплатно, а вот за полную версию предвычисленных таблиц придется выложить 500$. Вообще-то, их можно найти в сети и бесплатно, но здесь есть одно "но". Для работы с ними потребуется установить 60 Гбайт оперативной памяти. Такой объем домашние компьютеры уже не поддерживают. Это уже нехилая серверная конфигурация, на фоне которой жалкие 500$ погоды не делают. Зато и пароль взламывается мгновенно! Причем, не только LM, но NT. Так что криптография не стоит на месте! И надежность парольных защит с каждым годом вызывает все большие и большие сомнения.

Но это не единственное слабое место MS-CHAP v1. Есть и другие. Например, атакующий может прикинуться сервером и послать клиенту запрос на смену пароля, который не требует аутентификации и будет воспринят как правильный. (как уже было показано выше, возможность аутентификации сервера клиентом появилась только в MS-CHAP v2). Клиент вводит свой старый и новый пароль, атакующий благополучно их "съедает" и, используя "старый" пароль тут же подключается к серверу. Комментарии, как говорится, излишне. Вот и следуй рекомендации почаще менять пароли! Ведь если "блокировка пароля" отключена и установлено бессрочное время его использование, запрос на смену пароля будет звучать весьма странно, если не сказать подозрительно, а если пароли меняются каждые несколько дней, к этому привыкают и перестают обращать внимание.

Протокол MS-CHAP v2 намного более защищен и фокус с "подделкой" сервера в нем уже не проходит, однако, он остается уязвим для утилит типа Rainbow Crack, которые взламывают его за очень короткое время, правда, при условии, что у хакера имеется достаточно количество оперативной памяти. На сегодняшний день "лишних" 60 Гбайт практически ни у кого нет (те же, у кого она есть, взломами скорее всего не интересуются), однако, можно составить "урезанные" таблицы, содержащие ограниченный набор символов, что позволит находить хотя бы часть паролей, или использовать подкачку на быстрый диск (но в этом случае время взлома составит часы). Тем не менее, объемы памяти растут как на дрожжах. Несколько лет назад даже 1,6 Гигабайта казалось нереальной цифрой, а сейчас это домашняя конфигурация. Через год-другой, 60 Гигабайт станут доступны большинству хакеров и если MS-CHAP v2 останется широко распространен (а это будет именно так) по сетям прокатится целая волна взломов! И на обломках Pentium'ов потомки напишут: "он слишком много доверял VPN и MS-CHAP v2" .

Но это все касалось аутентификации. Теперь разберем шифрование.

шифрование снаружи и изнутри

В настоящее время поддерживаются два метода шифрования — MPPE (Microsoft Point-to-Point Encryption – Протокол Шифрования от Microsoft типа Точка-Точка), задекларированный в RFC 3078 и IPSec, описываемый целым тандемом RFC, вот только основные из них: RFC 1825 — IP Security Architecture (Архитектура Безопасности IP), RFC 1826 — IP Autentication Header (Заголовок Аутентификации IP), RFC 1827 IP Encapsulating Security Payload, ESP — Инкапсулированная Секретная Начинка в IP, RFC 1828 — IP Autentication using Keyed MD5 (Метод Аутентификации по ключам MD5), RFC 1829 — ESP DES-Chpher Block Chaining Transform (Преобразование Сцепленных Блоков Инкапсулированной Начинки Зашифрованной по DES). Помимо этого, существуют и другие IPSec RFC, которые легко найти на http://www.rfc-editor.org/. Это целый талмуд, с которым толпа мудрецов не успеет разобраться даже за тысячу и одну ночь!

Оба протокола реализованы как в Microsoft Windows, так и вне ее (например, в *BSD), на алгоритмы работы VPN могут существенно отличаться. В NT (и производных от нее системах). Основные сведения приведены в таблице 2.

протокол	метод шифрования
старый MPPE	RSA RC4, 40-, 56-разрядные ключи
новый MPPE	RSA RC4, 128-разрядные ключи
IPSec	DES, 56-разрядные ключи
IPSec Triple	3DES

Таблица 2 основные механизмы хэширования и аутентификации, используемые в VPN

Выбор метода шифрования определяется типом и конфигурацией VPN-сервера. При подключении по PPTP, применяется шифрование MPPE, а при подключении по L2TP (Layer 2 Tunneling Protocol) – шифрование IPSec. Протокол L2TP был разработан рабочей группой IETF

PPP Extensions с целью объединения функциональности Cisco L2F плюс PPTP, и стандартизирован в 1999 году документом RFC под номером 2661. Сейчас происходит его активное внедрение. Внешне L2TP очень похож на PPTP, но если PPTP работает только в IP-сетях, то L2TP поддерживает Frame Relay, X.25 и ATM.

Если Microsoft VPN-клиент настроен на автоматический выбор типа сервера (как и происходит по умолчанию), сначала предпринимается попытка использовать протокол L2TP с алгоритмом шифрования IPSec, и только если эта попытка не увенчалась успехом, происходит переход на PPTP с шифрованием по MPPE.

Рассмотрим PPTP как наиболее простой и распространенный протокол шифрования. С точки зрения хакеров он привлекателен тем, что использует простой потоковый шифр RC4, уязвимости которого хорошо известны. За последние несколько лет криптоаналетики разработали множество эффективных атак, а вычислительные мощности возросли настолько что даже лобовой подбор 40-битного ключа представляет плевое дело. Перечислим три основные уязвимости RC4.

Атака по открытому тексту: как работает RC4? На основе ключа шифрования генерируется псевдослучайная последовательность (она же гамма), которой накладывается на шифруемый текст через XOR. Что может быть проще! Если атакующий знает содержимое и позицию шифруемого байта, то путем повторного применения XOR он может восстановить один байт гаммы. Поскольку, шифруемые пакеты содержат предсказуемую информацию (например, заголовки), а одни и те же участки гаммы многократно накладываются на различные участки шифруемого текста, хакер может восстановить всю гамму целиком путем тривиального сбора трафика!

Атака "переворотом битов" (bit-flipping attack): после начальной аутентификации и установки соединения, остальные пакеты не аутентифицируются, поэтому, атакующий может свободно менять содержимое зашифрованных битов. Конкретная реализация атаки может выглядеть, например, так. Хакер перехватывает зашифрованный пакет сниффером, изменяет несколько битов пакета и пересчитывает его CRC, после чего передает пакет серверу, который благополучно "проглатывает" поддельный пакет (ведь CRC рассчитан правильно) и передает его на следующий уровень в стеке протоколов. На уровне 3 (layer 3) происходит конкретный облом. Пакет отвергается и генерируется вполне предсказуемый ответ, который передается "наверх", подвергаясь шифровке. Хакер вылавливает зашифрованный пакет и применяет атаку по прямому тексту, восстанавливая гамму ключа. Все! Теперь остальные пакеты расшифровываются на ура!

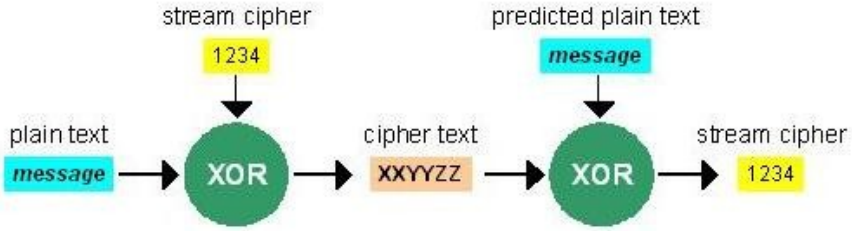

Рисунок 14 атака тика bit-flipping

Атака путем ресинхронизации: если процессе передачи теряется пакет, либо приходит пакет с неверным номером в заголовке MPPE, происходит так называемая "ресинхронизация ключа". Отправитель реинициализирует таблицы RC4 и устанавливает бит "сброшен" (flushed) в заголовке MPPE. Если система обнаруживает в пакете установленный бит "сброшен", она реинициализирует свои таблицы RC4 и устанавливает счетчик пакетов в соответствии с полученным значением. В практическом плане это означает, что шифрование гаммой начинается сначала. Если хакер будет бомбардировать жертву запросами на ресинхронизацию (а они не требуют никакой аутентификации), то все пакеты будут шифроваться одной и той же гаммой, что существенно упрощает атаку по открытому тексту.

Все три описываемых атаки главным образом относятся к MS-CHAP v1, а в MS-CHAP v2 их влияние значительно ослабло, поэтому основным способом взлома стал подбор пароля с помощью программ типа Rainbow Crack.

исследование VPN-сетей

Большинство компаний предпочитают не афишировать наличия VPN-сервера в своей сети. И это правильно, поскольку, практически ни одна реализация не смогла избежать программистских ошибок и за последние несколько лет было обнаружено множество дыр, но использовать их не так-то просто! Во-первых, необходимо выяснить IP-адрес VPN-севера, а во-вторых, как-то определить тип программного обеспечения и версию реализации.

Вот для этого и нужен IKE-scan! Это бесплатно распространяемая утилита, посылающая Control Connection запросы по UDP и анализирующая ответы. Как показала практика, каждая реализация имеет свой уникальный "почерк", на жаргоне называемый "отпечатком" (fingerprint), по которому ее можно определить. Методика снятия отпечатков постоянно совершенствуется и за подробной информацией лучше обратиться непосредственно к самим разработчикам (http://www.nta-monitor.com/ike-scan/overview-old.htm). Оттуда же можно скачать и готовую утилиту с исходными текстами в придачу: http://www.nta-monitor.com/ike-scan/download.htm. Как и большинство остальных программ этого рода, она ориентирована на UNIX, но неплохо чувствует себя и под Windows в среде Cygwin. Некоторые дистрибьютивы (например, DEBIAN) уже включают ее в штатный комплект поставки, поэтому, ничего качать не надо. Кстати говоря, в IKE-Scan'е недавно обнаружилась уязвимость… Забавно, инструмент для поиска дыр, сам оказался большой дырой…

заключение

Так все-таки можно доверять VPN сетям или нет? Ответ неоднозначен. Да, они действительно представляют собой дополнительный уровень защиты поверх традиционных сетей, однако, открывать доступ во внутреннюю корпоративную сеть через VPN очень опасно. Хакер без труда сможет подобрать пароль за столь короткое время, что администратор и глазом моргнуть не успеет. Последствия такого вторжения каждый может домыслить сам. Тут не нужно богатого воображения!

>>> врезка что читать

- Malware FAQ Microsoft PPTP VPN
  - подробный и доходчивый faq по взлому VPN (на английском языке) http://www.sans.org/resources/malwarefaq/pptp-vpn.php;
- Breaking the Secure Safe
  - фрагмент из книги Wireless Hacking: Breaking Through, посвященной взлому беспроводных сетей с кучей практических примеров (на английском языке): http://www.informit.com/articles/article.asp?p=353735&seqNum=8&rl=1;
- [The Crumbling Tunnel]-< A Menagerie of PPTP Vulnerabilities >
  - статья из phrack'a с обстоятельным анализом уязвимостей PPTP-протокола (на английском языке) http://www.phrack.org/phrack/53/P53-12;
- Криптоанализ туннельного протокола типа точка-точка (PPTP) от Microsoft
  - перевод статьи известного криптоаналетика Брюса Шрайера и этим все сказано (на русском языке): http://www.ssl.stu.neva.ru/psw/crypto/pptp.html;