Российский Государственный Технологический Университет им. К. Э. Циолковского - МАТИ

Виды атак

Доклад

Фонтанов Д. Ю., Голубев Ю. М., Бузлов И. А.
9/26/2012

Содержание

Виды атак на информационные системы

Мощные, грамотно построенные криптографические системы способны на многое, но нельзя считать их панацеей. Пользователи, уделяющие слишком много внимания алгоритмам шифрования в ущерб другим методам обеспечения безопасности, похожи на людей, которые вместо того, чтобы обнести свои владения высоким забором, перегораживают дорогу массивными воротами, нисколько не задумываясь о том, что злоумышленникам не составит труда сделать шаг в сторону и обойти сей "неприступный бастион".

В популярных журналах классификация продуктов шифрования проводится, как правило, по алгоритму и длине ключа. Обзоры печатаются под броскими заголовками. Описание особенностей того или иного продукта и его сравнение с конкурирующими предложениями легко укладывается буквально в несколько слов. Наверняка каждому из вас доводилось встречать утверждения типа: "128-разрядные ключи обеспечивают надежную защиту, в то время как 40-разрядные вскрываются довольно легко", "алгоритм triple-DES гораздо надежнее в сравнении с обычным алгоритмом DES" или даже "шифрование RSA с 2048-разрядным ключом лучше RSA с 1024-разрядным ключом".

Однако в криптографии не все так просто — более длинные ключи отнюдь не гарантируют повышенной безопасности.

Давайте попробуем сравнить алгоритмы шифрования с замком от входной двери. Во многих дверных замках установлено четыре металлических шипа, каждый из которых может находиться в одном из десяти положений. Если ключ соответствует конфигурации замка, запор открывается. Таким образом, конструкция каждого однотипного замка предусматривает 10 000 различных комбинаций. Следовательно, для того, чтобы проникнуть в дом, взломщик должен перепробовать до 10 000 ключей.

Замки усовершенствованной конструкции имеют уже 10 шипов (10 млрд. комбинаций), но это вовсе не значит, что за безопасность своего жилища теперь можно не беспокоиться. Совершенно очевидно, что взломщики не станут последовательно подбирать все возможные ключи (это было бы слишком примитивно): они достаточно умны, для того чтобы проникнуть в дом иным способом (в данном случае напрашивается аналогия с криптографической атакой). Гораздо проще и эффективнее разбить окно, выломать дверь или переодеться в форму полицейского и приставить ствол пистолета к голове ничего не подозревающего хозяина. Один из грабителей в Калифорнии, недолго думая, распилил стену дома бензопилой. Понятно, что от таких действий не спасут даже самые лучшие замки.

Мощные, грамотно построенные криптографические системы способны на многое, но нельзя считать их панацеей от всех бед. Пользователи, уделяющие слишком много внимания алгоритмам шифрования в ущерб другим методам обеспечения безопасности, похожи на людей, которые вместо того, чтобы обнести свои владения высоким забором, перегораживают дорогу массивными воротами, нисколько не задумываясь о том, что злоумышленникам не составит труда сделать шаг в сторону и обойти сей "неприступный бастион". Квалифицированные взломщики просочатся даже через самую незаметную брешь.

Компания Counterpane Systems вот уже в течение многих лет занимается созданием, анализом и взломом систем шифрования. Мы исследуем алгоритмы или протоколы, спецификации которых опубликованы в открытой печати; большая часть работы связана с изучением особенностей конкретных продуктов. Нам довелось проектировать и анализировать средства, защищающие част-

ную тайну, гарантирующие конфиденциальность, отстаивающие справедливость и обеспечивающие функционирование систем электронной торговли. Мы работали с самыми различными программными пакетами, автономными аппаратными средствами и аппаратно-программными решения. Нам прекрасно известны слабые места алгоритмов шифрования, но почти всегда мы находили более элегантные способы обхода систем безопасности.

Атаки на архитектуру

Криптографическая система не может быть надежнее использованных в ней отдельных алгоритмов шифрования. Иными словами, для того чтобы преодолеть систему защиты, достаточно взломать любой из ее компонентов. Использование хороших строительных материалов еще не является гарантией прочности здания. Так и криптографическая система, построенная на основе мощных алгоритмов и протоколов, тоже может оказаться слабой.

Многие системы "теряют гарантию" безопасности, если используются неправильно. Скажем, проверка допустимости значений переменных не выполняется, "случайные" параметры используются многократно, что совершенно недопустимо. Алгоритмы шифрования необязательно обеспечивают целостность данных. Протоколы обмена ключами необязательно гарантируют, что обе стороны получат один и тот же ключ.

Некоторые системы шифрования, использующие связанные ключи, могут быть взломаны, даже если каждый ключ в отдельности надежен. Чтобы обеспечить безопасность, недостаточно просто реализовать алгоритм и ждать, что все будет работать. Даже наличие квалифицированных инженеров, помощь известных компаний и упорный труд не могут гарантировать абсолютной надежности. Бреши, обнаруженные в алгоритмах шифрования систем сотовой связи стандартов CDMA и GSM, а также в протоколе Microsoft Point-to-Point Tunneling Protocol (PPTP), наглядно это иллюстрируют. К примеру, в достаточно надежном алгоритме RC4, на котором построен протокол PPTP, нам удалось обнаружить режим, который делал защиту абсолютно прозрачной.

Еще одно слабое место криптографических средств - генераторы случайных чисел. Разработать хороший генератор случайных чисел непросто, поскольку он часто зависит от особенностей аппаратного и программного обеспечения. Сама система шифрования может быть выполнена на высоком уровне, но если генератор случайных чисел выдает легко угадываемые ключи, то все оставшиеся барьеры преодолеваются без особого труда. В ряде продуктов используются генераторы случайных чисел, вырабатывающие ключи, в которых прослеживается определенная закономерность. В таких случаях о безопасности говорить не приходится. Интересно, что применение одного и того же генератора в некоторых областях обеспечивает требуемую степень безопасности, а в других - нет.

Еще одно возможное слабое место - взаимодействие между по отдельности безопасными протоколами шифрования. Почти для каждого безопасного протокола, как правило, можно найти другой, не менее надежный, который сведет на нет все достоинства первого, если они оба используют одинаковые ключи на одном и том же устройстве. Если различные стандарты защиты применяются в одной среде, недостаточно четкое взаимодействие между ними зачастую может привести к весьма нежелательным последствиям.

Атаки на конкретные реализации

Многие системы подводят из-за ошибок в реализации. Некоторые продукты не гарантируют, что, зашифровав текст, они уничтожат оригинал. В других для предупреждения потери информации в случае системного сбоя используются временные файлы, а доступная оперативная память расши-

ряется за счет памяти виртуальной; в этом случае на жестком диске могут оставаться отдельные фрагменты незашифрованного текста.

Переполнение буферов, не стертая до конца секретная информация, недостаточно надежная система обнаружения и восстановления после ошибок - все это примеры брешей в конкретных реализациях, через которые очень часто и проникают злоумышленники. В наиболее вопиющих случаях операционная система даже оставляет ключи на жестком диске. В одном из продуктов крупной софтверной компании ввод пароля осуществлялся через специальное окно. При этом пароль сохранялся в буфере окна и после его закрытия. Проводить дальнейшие исследования защищенности системы уже не имело смысла. Мы проникли в нее через пользовательский интерфейс.

Слабые стороны других продуктов не так явно бросались в глаза. Иногда одни и те же данные шифровались при помощи двух ключей: первый из них был надежным, а второй подбирался достаточно легко; но при этом эксперименты с уже подобранным ключом помогал подобрать и другой. В других системах применялись мастер-ключи и ключи "на один сеанс"; причем безопасности главного ключа уделялось недостаточное внимание, а основные надежды возлагались на одноразовые ключи. Для создания по-настоящему надежной системы безопасности необходимо полностью исключить возможность анализа строения ключей, а не ограничиваться лишь самыми очевидными мерами предосторожности.

Создатели систем электронной коммерции часто вынуждены идти на компромисс ради расширения функциональности. И поскольку разработчики предпочитают жертвовать безопасностью, в защите то и дело появляются дыры. Сверка учетных записей, к примеру, может проводиться только раз в день, но за несколько часов взломщик способен нанести поистине колоссальный ущерб! Перегрузка процедуры идентификации может привести к тому, что личность атакующего не будет распознана. Некоторые системы заносят сомнительные ключи в "черные списки"; получение доступа к этим спискам существенно облегчает задачу взломщика. Многие системы защиты преодолеваются после повторных атак и использования старых сообщений или их частей, сбивающих систему с толку.

Потенциальная опасность заложена в возможности восстановления ранее использовавшихся ключей в системах с расщеплением. В хороших криптографических системах срок жизни ключей ограничивается максимально коротким промежутком времени. Процедура восстановления позволяет продлить жизнь ключа уже после того, как от него отказались. Используемые для восстановления ключей базы данных сами по себе являются источником опасности, и их архитектура должна быть выверена с особой тщательностью. В ряде случаев бреши в них позволяли взломщикам маскироваться под легальных пользователей.

Атаки на оборудование

Некоторые системы (чаще всего коммерческого назначения) имеют так называемое "кольцо безопасности", состоящее из аппаратных средств повышенной устойчивостью к взломам (смарт-карт, электронных бумажников, электронных ключей и т.д.). Создатели подобных систем исходят из предположения, что архитектура системы внутри этого кольца надежно защищена от несанкционированного доступа. Надежность оборудования - очень важная составляющая комплексных систем безопасности, но не стоит полностью доверять решениям, защищающим только от воровства и неумелого обращения.

Большинство подобных технологий на практике не работают, а инструменты для их взлома непрерывно совершенствуются. При проектировании подобных систем очень важно не забывать о до-

полнительных механизмах защиты, которые должны срабатывать, если взломщикам удастся преодолеть первые оборонительные редуты. Нужно постараться максимально осложнить задачу противника и сделать ее решение невыгодным с экономической точки зрения. Стоимость защищаемых данных должна быть значительно ниже затрат на разрушение системы безопасности. Ценность электронного проездного не может идти ни в какое сравнение со стоимостью портфеля ценных бумаг. Исходя из этого и следует проектировать средства защиты.

В 1995 году значительно возросло число "атак по расписанию": выяснилось, что секретные ключи RSA можно восстанавливать, измеряя временные интервалы между операциями шифрования. Был зарегистрирован ряд случаев успешного взлома смарт-карт, а также серверов электронной коммерции в Internet. Обнаружилось, что атаки строились на основе измерения потребляемой мощности, анализа электромагнитного излучения и других побочных источников информации. Специалистам по криптографии удалось по этим признакам реконструировать логику многих систем с открытыми ключами, продемонстрировав их ненадежность.

Большую популярность приобрел метод анализа сбоев, позволяющий находить слабые места криптопроцессоров и восстанавливать секретные ключи. Подобные методы по своему духу скорее биологические. Криптографические системы в этом случае рассматриваются как сложные объекты, которые реагируют на внешние раздражители. Их нельзя четко описать с помощью математических уравнений, но последствия таких атак разрушительны.

Атаки на модели доверительных отношений

Многие интересные способы преодоления защитных рубежей связаны с моделями доверительных отношений внутри системы. Прежде всего, следует выявить связи между отдельными компонентами системы, уяснить ограничения и механизм реализации схемы доверительных отношений. Простые системы (средства шифрования телефонных переговоров и информации на жестких дисках) используют элементарные доверительные модели. Комплексные системы (средства электронной торговли или средства защиты многопользовательских пакетов электронной почты) построены на основе сложных (и гораздо более надежных) моделей доверительных отношений, описывающих взаимосвязи множества элементов.

В программе электронной почты может использоваться супернадежный алгоритм шифрования сообщений, но если ключи не сертифицированы источником, заслуживающим доверия, и сертификация эта не может быть подтверждена в реальном времени, безопасность системы остается под вопросом. Некоторые торговые системы могут быть вскрыты по соглашению продавца с покупателем или в результате объединенных усилий нескольких клиентов. В других системах предусмотрено наличие средств обеспечения безопасности, но качество этих средств никто никогда не проверял. Если модель доверительных отношений не документирована, то в процессе развертывания в продукт можно случайно внести какие-либо недопустимые изменения, после чего стройность системы безопасности будет нарушена.

Многие программные пакеты слишком доверяются защищенности аппаратных средств. Предполагается, что компьютер абсолютно безопасен. Рано или поздно в такую программу проникает "троянский конь", который подбирает пароли, считывает незашифрованный текст или каким-то иным образом вмешивается в работу системы защиты. Разработчикам систем, функционирующих в компьютерных сетях, следует побеспокоиться о безопасности сетевых протоколов. Уязвимость компьютеров, подключенных к Internet, многократно возрастает.

Система шифрования, которая преодолевается "со стороны сети", никуда не годится. Не существует программ, безопасность которых выстояла после того, как противнику удалось применить обратное проектирование. Очень часто система проектируется в расчете на одну модель доверительных отношений, а в практической реализации фигурирует совсем другая. Принятые в процессе проектирования решения полностью игнорируются после передачи готового продукта пользователям. Система, которая абсолютно безопасна, когда ее операторы заслуживают доверия, а доступ к компьютерам полностью контролируется, теряет все свои преимущества, если обязанности операторов выполняют низкооплачиваемые работники, нанятые на короткий срок, а физический контроль за компьютерами утрачен.

Впрочем, хорошие модели доверительных отношений продолжают работать даже в том случае, если отдельные компоненты подводят.

Атаки на пользователей

Даже если система гарантирует надежную защиту при правильной эксплуатации, пользователи могут случайно нарушить ее, особенно если система спроектирована недостаточно хорошо. Классическим примером является сотрудник, предоставляющий свой пароль коллегам с тем, чтобы они имели возможность решать неотложные задачи во время его отсутствия. Атака с учетом "человеческого фактора" зачастую оказывается куда более эффективной, чем месяцы кропотливого анализа алгоритмов.

Пользователи могут в течение нескольких дней не сообщать об утере смарт-карты. Они не уделяют требуемого внимания проверке электронной подписи. Секретные пароли порой повторно используются в несекретных системах. Клиенты даже не пытаются ликвидировать слабые места в системе безопасности. Конечно, даже хорошие системы не в состоянии ликвидировать последствия причин социального свойства, но они могут свести их к минимуму.

Многие продукты взламываются потому, что их защита построена на основе паролей, генерируемых пользователями. Предоставленные сами себе люди не задумываются о том, как выбрать необычную последовательность символов. Ведь пароль, который невозможно подобрать, не так просто запомнить. Если в качестве секретного ключа применяется такой пароль, то подобрать его, как правило, удается гораздо проще и быстрее, чем используя метод грубой силы.

Многие пользовательские интерфейсы еще больше облегчают задачу взломщика, ограничивая длину пароля 8 знаками, преобразуя вводимую последовательность в символы нижнего регистра и т.д. Даже пароли-фразы не обеспечивают требуемой степени безопасности. Злоумышленнику намного легче подобрать фразу из 40 букв, чем перебирать все возможные последовательности 64-разрядных случайных ключей. Иногда защита, в которой применяются очень надежный механизм ключи сеансов, разрушается из-за использования слабых паролей, предназначенных для восстановления ключей. Желание облегчить восстановление системы после сбоя фактически открывает перед атакующими черный ход.

Атака на средства восстановления после сбоев

Разработчики надежных систем не в состоянии заделать в заборе безопасности все мельчайшие щели, но по крайней мере зияющие дыры они ликвидируют. Восстановление ключа к одному файлу не позволит взломщику считать всю информацию, находящуюся на жестком диске. Изготовление фальшивых денег - очень серьезное преступление, ведь обладатель технологии печатания денег может уничтожить национальную валюту. Хакер, взламывающий смарт-карту, изучает секреты данного конкретного устройства, а не всех остальных смарт-карт, входящих в систему. В мно-

гопользовательских системах знание секретов одного человека не должно открывать доступа к информации других.

Многие системы по умолчанию устанавливаются в режим с отключенными средствами безопасности. Если система защиты "заедает", пользователь просто отключает ее и продолжает заниматься своим делом. Такое поведение делает особенно эффективными атаки типа denial-of-service ("отказ в обслуживании"). Если онлайновая система авторизации кредитных карт отключена, продавец вынужден довольствоваться значительно менее надежной бумажной технологией.

Иногда у взломщиков появляется возможность воспользоваться обратной совместимостью различных версий программного обеспечения. Как правило, в каждом новом варианте продукта разработчики стараются устранить бреши, имевшиеся в старом. Но требование обратной совместимости позволяет атакующему применять протокол старой, незащищенной версии.

Некоторые системы не имеют средств восстановления. Если защита разрушена, вернуть программу в работоспособное состояние не представляется возможным. Выход из строя системы электронной торговли, к которой обращаются миллионы клиентов, грозит обернуться катастрофой. Поэтому подобные системы должны иметь средства организации противодействия атакующим и поддерживать возможность обновления системы безопасности без остановки программы.

Хорошо продуманная система сама знает, как лучше противостоять атаке и что следует делать для устранения повреждений и оперативного восстановления работоспособности.

Атака на средства шифрования

Иногда слабые места можно найти и непосредственно в системе шифрования. Некоторые продукты создаются на базе не слишком удачных алгоритмов собственной разработки. Как правило, вскрыть известные алгоритмы шифрования удается лишь в исключительных случаях. Если же разработчик делает ставку на собственные методы, шансы взломщиков повышаются многократно. Незнание секрета алгоритма не является особым препятствием. Квалифицированному специалисту достаточно пары дней, чтобы по объектному коду восстановить исходный алгоритм шифрования.

Надежность стандартной для электронной почты архитектуры S/MIME 2 не в состоянии компенсировать слабостей алгоритма шифрования. И без того не слишком надежная защита GSM от слабого алгоритма шифрования проигрывает еще больше. Во многих системах используются слишком короткие ключи.

Можно привести множество других примеров ошибок в системах шифрования: программы повторно генерируют "уникальные" случайные значения, алгоритмы цифровой подписи не в состоянии обеспечить контроль за передаваемыми параметрами, хэш-функции открывают то, что должны защищать. В протоколы шифрования вносятся не предусмотренные разработчиками изменения. Пользователи любят "оптимизировать" имеющиеся средства, доводя их до столь примитивного уровня, что вся система защиты рушится, как карточный домик.

Предупреждение, а не выявление

Средства шифрования снижают вероятность того, что пользователи станут жертвами обмана, мошенничества, некорректных действий и т.д. Но архитектуру безопасности нельзя ограничивать столь узкими рамками.

Надежная система должна самостоятельно обнаруживать несанкционированные операции и ликвидировать нежелательные последствия атаки. Один из основных принципов проектирования подобных систем заключается в знании того, что рано или поздно атаки противника увенчаются успехом. Скорее всего, удар будет нанесен в самом неожиданном направлении, с использованием неизвестных разработчикам методов. Очень важно своевременно распознать такое нападение и принять все необходимые меры к тому, чтобы минимизировать ущерб.

Еще важнее как можно быстрее восстановить работоспособность поврежденной в ходе атаки системы. Необходимо сгенерировать новые пары ключей, заменить протокол, прекратить использование раскрытых противником средств, исключить из системы узлы, к которым взломщику удалось получить доступ, и т.д. К сожалению, многие продукты не занимаются сбором нужной информации, не контролируют ситуацию и не в состоянии надежно защитить данные от изменений.

В журнале регистрации должны отражаться все события, позволяющие установить факт нападения. В случае необходимости должны быть представлены неопровержимые доказательства, способные убедить судей и присяжных в виновности злоумышленника.

Разработчики систем безопасности должны следовать наставлениям таких авторитетов, как прусский генерал Карл фон Клаузевитц, утверждавший, что хорошие оборонительные средства должны отражать любые удары, даже те, о которых на сегодняшний день еще ничего не известно.

Атакующим, напротив, достаточно найти одну единственную брешь, и вся система защиты будет ликвидирована. Они прибегают к самым разнообразным уловкам. Взломщики не гнушаются участвовать в заговорах, тщательно маскируют свои противоправные действия и готовы в течение достаточно продолжительного времени ждать появления необходимых средств. В их арсенале всегда найдется идея, позволяющая нанести неожиданный для разработчиков удар.

Нет ничего проще, чем оградить свою информацию шатким, непрочным барьером с зияющими в нем дырами. Построить же непроницаемую систему защиты очень сложно. К сожалению, многие пользователи не видят разницы. В других областях анализ функциональных возможностей позволяет без труда отличить качественные продукты от наспех построенных систем. Достоинства хорошего кодека видны невооруженным глазом, плохой же выглядит значительно слабее и не поддерживает тех функций, которые доступны его конкурентам.

В криптографии все по-другому. Тот факт, что программы шифрования работают, еще не позволяет говорить о надежной защите. Как создается большинство продуктов? Разработчики читают Applied Cryptography, выбирают приглянувшийся им алгоритм и протокол, тестируют его, и вот уже проект готов. На самом деле все не так просто.

Функциональность и высокое качество не являются синонимами, и даже бесконечное тестирование не позволяет устранить всех брешей в системе защиты. Нужно хорошо разбираться в тонкостях терминологии: даже продукты, обладающие абсолютно надежными средствами шифрования, зачастую не могут гарантировать пользователям полной безопасности.

Виды атак на хеш-функции

Возможные атаки на хеш-функции подразделяются на следующие виды:

1. Атаки, не зависящие от алгоритма
2. Атаки, направленные на методы создания лавинного эффекта
3. Атаки, зависящие от взаимодействия с системой ЭЦП
4. Атаки, зависящие от лежащего в основе блочного алгоритма шифрования
5. Высокоуровневые атаки

Прежде, чем описать атаки подробнее, рассмотрим, какая информация доступна атакующему. В случае MDC вся информация публично доступна и атакующему требуется найти прообраз или сообщение с таким же образом в случае OWHF, или же коллизию в случае CRHF. Различают следующие случаи, в зависимости от того, отличен ли IV от стандартного:

- **Прообраз**: атакующий пытается найти прообраз для данного хеша
- **Второй прообраз**: атакующий пытается найти второй прообраз для данного хеша
- **Псевдо-прообраз**: атакующий пытается найти прообраз для данного хеша, при $IV' \neq IV$
- **Второй псевдо-прообраз**: атакующий пытается найти второй прообраз для данного хеша, при $IV' \neq IV$
- **Коллизия**: атакующий пытается найти коллизию
- **Коллизия для IV, отличного от стандартного**: атакующий пытается найти коллизию при $IV' \neq IV$
- **Псевдо-коллизия**: атакующий пытается найти для некоторых IV' и IV'' пару X', X'', такую, что $h_{IV'}(X') = h_{IV''}(X'')$.

По определениям понятно, что нахождение псевдо-коллизии не сложнее, чем нахождение псевдо-прообраза, на нахождение коллизии не сложнее, чем нахождение (второго) прообраза.

В случае MAC ситуация чуть сложнее. В зависимости от информации, доступной атакующему, различают следующие типы атак:

- **Атака на основе открытых текстов**: у атакующего есть возможность изучать открытые тексты и соотвествующие им MAC
- **Атака на основе подобранного отркрытого текста**: атакующий может выбирать открытые тексты и получать соответствующие им MAC
- **Атака на основе адаптивно подобранного открытого текста**: атакующий может получать MAC для выбранного им текста

Под ``взломом'' хеш-функции обычно имеют в виду следующее:

- **Полный взлом**: атакующий определяет секретный ключ K
- **Общая подделка**: атакующий находит алгоритм, который функционально аналогичен используемому при генерации MAC алгоритму
- **Выборочная подделка**: атакующий вычисляет правильный MAC для выбранного им открытого текста
- **Экзистенциальная подделка**: атакующий определяет MAC для хотя бы одного открытого текста: т.к. атакующий не может вычислить MAC для любого другого открытого текста, это «открытие» может быть случайным и не имеющим особого значения

Атаки, не зависящие от алгоритма

Этот класс атак зависит от длины хеша n и длины секретного ключа K (в случае MAC) и не зависит от вида алгоритма. Предполагается, что хеш равномерно распределен и представляет собой случайную величину, иначе этот класс атак будет еще более эффективен.

Случайная атака

Оппонент выбирает случайное сообщение и надеется, что хеш останется неизменным. Если используемая хеш-функция обеспечивает равномерное распределение получаемого хеша, то вероятность успеха этой атаки равна $1/n^2$, где n это длина хеша в битах. Важное различие между MAC и MDC здесь заключается в том, что для атаки на MAC атакующему нужно иметь непосредственный доступ к системе, в которой сохранен секретный ключ и в этом случае атака зависит от количества попыток T, которыми он располагает.

Ожидаемое значение такой атаки равна $T \cdot V/2^n$, где V это ожидаемая прибыль. Если количество попыток или ожидаемое значение может быть ограничено, как, например, в банковских приложениях, то $n = 32$ будет достаточно, большинству же других приложений следует использовать 64 и более бит.

Для MDC атака может проводиться оффлайн и паралелльно, это значит, что длина хеша должна быть как минимум 64 бита.

Исчерпывающий поиск ключей

Эта атака применима только к MAC. Это атака на основе открытых текстов, где атакующий знает M пар открытый текст-MAC для данного ключа. Он попытается перебрать все возможные значения ключа, чтобы исключить неподходящие. Длина ключа равна k бит, а ожидаемое число оставшихся ключей равно K_{exp}. Если M достаточно большое, то возможно однозначно определить секретный ключ K, или $K_{exp} \approx 1$. Для правильного ключа атакующему будет достаточно вычислить MAC M раз, в то время как для плохого ключа вероятность того, что потребуется совершить i вычислений равна

$$\left(1 - \frac{1}{2^n}\right) 2^{-n(i-1)}$$

Ожидаемое число попыток в таком случае определяется следующим неравенством:

$$\left(1 - \frac{1}{2^n}\right) \sum_{i=1}^{M} \frac{i}{2^{n(i-1)}} < \frac{1}{1 - 2^{-n}}$$

Общее количество попыток для нахождения ключа ограничено сверху выражением

$$M + \frac{2^k - 1}{1 - 2^{-n}}$$

а количество ключей, которые осталось проверить, равно

$$K_{exp} = 1 + \frac{2^k - 1}{2^{Mn}}$$

Это означает, что количество пар открытый текст-MAC намного больше, чем k/n. После описания атаки дней рождения мы обсудим, каким должно быть k для обеспечения достаточной безопасности в течение ближайших десятилетий.

Атака дней рождения

Идея этой атаки основана на том, что для группы из 23 человек вероятность того, что у двоих из них день рождения будет в один день, превышает $1/2$. Т.к. число 23 намного меньше числа, которое предполагает большинство людей, это называется «парадокс дней рождения». Другой пример: если есть две группы по 17 человек в каждой, то вероятность того, что у двух людей в разных группах совпадут дни рождения, также превышает $1/2$. Здесь предполагается, что дни рождений случайно и равномерно распределены по всем дням в году, иначу вероятность совпадений даже выше. Это можно обобщить следующим образом: если взять два объекта размерами r_1 и r_2 из набора с n объектами и если $r_1 r_2 = n$ при $r_1, r_2 = O(\sqrt{n})$, то вероятность совпадения равна $1 - 1/e = 63\%$.

Первая атака, основанная на этом свойстве была предложена Г. Ювалем в 1979 году. Он показал, что проще найти коллизию для односторонней хеш-функции, чем пытаться найти прообраз для элемента из заданного диапазона. Коллизия может быть найдена следующим образом:

- Противник генерирует r_1 поддельных сообщений и r_2 подлинных. Если $r_1 = r_2 = \sqrt{n}$, то вероятность нахождения коллизии равна 63%.
- Поиск совпадений не требует r^2 операций: после сортировки данных, которая потребует $O(r \log r)$ операций, сравнение будет простым.

Алгоритмическое улучшение для подобного поиска коллизий было предложено Дж.-Дж. Квисквотером в 1990 году: оно аннилирует требования к дисковому пространству атакующего, если у него есть возможность применять требуемую хеш-функцию. Идея состоит в том, что происходит итерация по случайному отображению и выходные данные используются в качестве входных. Рано или поздно такая итерация после λ шагов будет повторяться в цикле с периодом μ. В момент, когда итерация начинает повторяться, будет найдено два значения x' и x'' такие, что $f(x') = f(x'')$. Ожидаемое количество вычислений хеш-функции равно $\rho = \lambda + \mu = \sqrt{\pi n/8} + \sqrt{\pi n/8} = \sqrt{\pi n/2}$.

Осуществимость

Можно предположить, что, согласно закону Мура, рано или поздно появится компьютер, который сможет полным перебором взломать любой шифротекст. Но это не совсем так. Существуют фундаментальные ограничения вычислительной мощности, наложенные структурой вселенной: например, скорость передачи любого сигнала не может превышать скорость распространения света в вакууме, а количество атомов во Вселенной (из которых, в конечном счете, состоят компьютеры) огромно, но конечно. Рассмотрим два фундаментальных ограничения:

1. Предел, основанный на выделяемой Солнцем энергии. Все вычисления потребляют энергию. Согласно принципам классической термодинамики и статистической механики, потребление энергии при осуществлении необратимого преобразования (вычисления) имеет порядок kT, где T – температура окружающей среды, а k – постоянная Больцмана. Мощность излучения Солнца составляет приблизительно $3.86 \cdot 10^{26}$Вт; таким образом, за весь свой предполагаемый период существования - 10 млрд. лет, или $3 \cdot 10^{17}$ секунд - Солнце выделит около 10^{44} Дж. Предположим, температура окружающей среды $T = 10^{-6}$ градусов, тогда энергозатраты на одну операцию составляют $1.4 \cdot 10^{-29}$ Дж. Значит, количество вычислительных операций, которые можно осуществить с использованием всей выделяемой солнцем энергии, равно выделяемой мощности, поделенной на

количество энергии, необходимой для осуществления одной операции, т.е. всего 10^{73} операций. Такое количество операций потребовалось бы на взлом ключа из 73 десятичных цифр (или около 250 бит) методом прямого перебора при грубом предположении, что для проверки одного значения ключа необходима всего одна операция (на самом деле - сотни операций). Для справки, количество атомов солнечной системы – около 10^{60}.

2. Предел, основанный на массе Земли. Масса Земли составляет порядка $6 \cdot 10^{24}$ кг. Масса протона $1.6 \cdot 10^{-27}$кг, т.е. Земля содержит приблизительно $4 \cdot 10^{51}$ протонов. Сопоставим каждому протону отдельный компьютер и примем за скорость выполнения операции на таком компьютере время, за которое луч света проходит расстояние, равное диаметру этого протона. Таким образом, каждый компьютер может выполнять $3 \cdot 10^{25}$ операций в секунду. Если все эти компьютеры будут работать параллельно, их суммарное быстродействие составит $4 \cdot 10^{51} + 3 \cdot 10^{25}$ операций в секунду, т.е. 10^{77} операций в секунду. Возраст Вселенной приблизительно 10 млрд. лет, или $3 \cdot 10^{17}$ секунд. За весь период существования Вселенной такие гипотетические компьютеры размером с протон смогли бы выполнить $3 \cdot 10^{94}$ операций. При описанных в п. 1 предположений относительно количества операций, необходимых на проверку значения ключа, такое количество операций позволит взломать ключ длиной 95 десятичных цифр, или 320 бит.

Атаки, направленные на методы создания лавинного эффекта

Этот класс атак основан на некоторых высокоуровневых свойствах элементарной функции f.

Атака «встреча посередине»

Эта атака является разновидностью атаки дней рождения, но вместо хеша в ней сравниваются промежуточные переменные. В отличие от атаки дней рождения, такая атака позволяет создать сообщение с заранее известным хешом, она также применима к OWHF. Атакующий создает r_1 вариаций первой части сообщения и r_2 вариаций последней. Начиная с начального значения IV и двигаясь «назад» от хеша, вероятность совпадения значения промежуточной переменной такая же, как и в атаке дней рождения. Единственное ограничение состоит в том, что точка «встречи» не должна быть на первом или последнем блоке сообщения. Алгоритм Квисквотера может быть легко изменен для подобного класса атаки, при этом требования к диковому пространству атакующего также останется несущественным.

Атака корректирования блока

Эта атака заключается в замене всех блоков сообщения, кроме неторого X_j. Затем этот блок генерируется таким образом, чтобы хеш принял нужное значение. Хеш-функции, основанные на модульной арифметике, особенно уязвимы этому типу атак.

Атака корректирования блока может быть также применена для получения коллизии. Атакующий начинает с двух произвольных сообщений X' и X'' и добавляет к ним один или более корректирующих блоков Y' и Y'' таких, что сообщения $X'||Y'$ и $X''||Y''$ имеют одинаковый хеш.

Атака фиксированной точки

Идея этой атаки заключается в поиске X_i и H_{i-1} таких, что $f(X_i, H_{i-1}) = H_{i-1}$. Если промежуточная переменная равна H_{i-1}, то возможно вставить в сообщение произвольное число произвольных блоков без изменения результирующего хеша. Нахождение коллизии или второго прообраза в результате этой атаке возможно, если промежуточную переменную удастся сделать равной H_{i-1}. Однако, предотвратить эту атаку довольно просто: достаточно добавлять к сообщению его длину до или после хеширования.

Дифференциальные атаки

Дифференциальные атаки основаны на отношении разностей входных и выходных данных и применим как к блочным алгоритмам шифрования, так и к хеш-функциям. При выполнении этих атак находится зависимость между различиями входных и выходных данных. Эти атаки применемы только к алгоритмам шифрования, которые удовлетворяют некоторым условиям, т.н. марковским шифрам. Им подвержены, например, DES, FEAL, LOKI, IDEA и PES. Для таких шифров в результате применения этой атаке будет найден ключ на основании большого количества открытых текстов и соответствующих им шифротекстов. Проще всего провести эту атаку на MDC, где вся информация доступна публично. К таким атакам относятся обнаруженные относительно недавно т.н. *туннели*, появляющиеся, если слова состояний подходят под определенный шаблон.

Атаки, зависящие от взаимодействия с системой ЭЦП

В некоторых случаях возможно подделать ЭЦП, даже если применяемая хеш-функция является стойкой к коллизиям.

Атаки, зависящие от лежащего в основе блочного алгоритма шифрования

Некоторые уязвимости алгоритмов блочного шифрования не важны, если их задача заключается в шифровании документа, но могут быть критичны, если этот алгоритм используется для хеширования. Эти уязвимости могут быть использованы для изменения сообщения без изменения результирующего хеша или для создания сообщений с заранее известным хешем.

Свойство отрицания

Одно из известных свойств DES заключается в симметричности относительно отрицания:

$$\forall\, P, K\colon C = DES(K, P) \Leftrightarrow \bar{C} = DES(\overline{K}, \bar{P})$$

Это свойство может помочь уменьшить время, затраченное на перебор ключей в два раза и также может быть использовано для нахождения простейших коллизий.

Слабые ключи

Другим известным свойством DES является существование 4-х слабых ключей. Для этих ключей выполняется следующее: $E\big(K, E(K, P)\big) = P, \forall P$. Также существует 6 пар частично слабых ключей, для которых $K\big(K_2, E(K_1, P)\big) = P, \forall P$. Это свойство может быть использовано для создания фиксированных точек.

Фиксированные точки

Фиксированные точки это текст, которых после шифрования (хеширования) равен самому себе для некоторых ключей. Т.к. хороший алгоритм шифрования это почти случайное преобразование, то для каждого ключа с вероятностью $1 - 1^{-e}$ существует фиксированная точка. При некоторых обстоятельствах легко найти фиксированные точки:

- В случае DES можно использовать слабые ключи: для каждого слабого ключа K_p существует 2^{32} сообщений P, для которых $E\big(K_p, P\big) = P$. Аналогично для частично слабых ключей $E\big(K_{ap}, P\big) = \bar{P}$.
- Для блочного алгоритма шифрования LOKI существует 256 фиксированных точек, где ключ имеет форму $gggggggghhhhhhhh_x$, а открытый текст $iiiiiiiiiiiiiiii_x$, где $i = g \oplus h$. g, h и i

это 4-хбитные числа в шестнадцатиричной системе счисления. Следовательно, для каждого такого слабого ключа существует 2^{32} фиксированных точек.

Высокоуровневые атаки

Даже если описанные выше атаки неосуществимы, следует применять особые меры, чтобы не пересылать повторяющихся сообщений или создавать новые сообщения путем комбинирования шифрованных (хешированных) ранее.

Для аутентификации сообщений, атаки на этом уровне могут быть предотвращены добавлением нонса – избыточной информации, являющейся уникальной для каждого сообщения. Важно проверять аутентичность нонса так же тщательно, как и аутентичность самого сообщения.

Применяют следующие виды нонсов:

- **Отметка времени (timestamp)**: дата и время отправки сообщения. Если точность времени достаточно высока, то это может служить уникальным идентификатором сообщения.
- **Серийный номер**: уникальный номер, назначенный каждому сообщению.
- **Случайное число**: достаточно длинное случайное число, присоединяемое к сообщению. Чтобы предотвратить атаку дней рождения, это число должно быть больше, чем квадрат числа всех сообщений, которые будут отправлены с данным ключом.