Узбекское агентство связи и информатизации

Ташкентский университет информационных технологий

Кафедра "Информационная безопасность"

Допустить к защите

Зав. кафедрой _____

_____

Выпускная квалификационная работа бакалавра

на тему: Анализ методов обеспечение целостности информации в сетях передачи данных.

Выпускник: Маматкулов Ислом Бекмуратович

Руководитель: Юлдашев Муродуллажон Дадамирзаевич

Рецензент: Ахмедова О.П.

Консультант по БЖД: Агзамова М.

Ташкент 2010

Узбекское агентство связи и информатизации

Ташкентский университет информационных технологий

Факультет_____ кафедра_____

Направление (специальность) _____
_____

"УТВЕРЖДАЮ"
Зав. кафедрой _____

_____

задание

на выпускную квалификационную работу

_____
(фамилия, имя, отчество)

1. Тема работы _____
_____
_____
Утверждена приказом по университету от "__" ____ 20__ г. № _____
2. Срок сдачи законченной работы _____
3. Исходные данные к работе _____
_____
_____
_____
4. Содержание расчетно-пояснительной записки (перечень подлежащих разработке вопросов) _____
_____
_____
_____
5. Перечень графического материала _____
_____
_____

7. Дата выдачи задания _____

Руководитель _____

(подпись)

Задание принял _____

8. Консультанты по отдельным разделам выпускной работы

Раздел	Ф.И.О. руководителя	Подпись, дата	
		Задание выдал	Задание получил

9. График выполнения работы

№	Наименование раздела работы	Срок выполнения	Отметка руководителя о выполнении
1	Угрозы безопасности		
2	Избыточные коды		
3	CRC коды		
4	Хэш функция		
5	HashMAC		
6	Безопастность жизнедеятельности		

Выпускник _____
_____ "____" _____2010 г.
(подпись)

Руководитель _____
_____ "____" _____2010 г.
(подпись)

Аннотация

В выпускной квалификационной работе рассмотрено обеспечение целостности информации в сетях передачи данных. А также анализировано алгоритмы кодирование и хэш функции которые используется для обеспечение целостности информации.

In exhaust work is devoted to ways of providing integrity of information at data send networks . Also analyzed algorithms coding and hash functions which used provides integrity.

Битирув малакавий ишда маълумотларни узатиш тармоқларида узатилаётган ахборотнинг тўлиқлигини таъминлаш усуллари кўриб чиқилган. Тўлиқлиликни таъминлашда ишлатиладиган кодлаштириш ва хэш функция алгоритмлари таҳлил қилинган.

1. Проблемы защиты информации в сетях передачи данных

1.1 Классификации основных угроз

Информатизация является характерной чертой жизни современного общества. Новые информационные технологии активно внедряются во все сферы народного хозяйства. Компьютеры управляют космическими кораблями и самолетами, контролируют работу атомных электростанций, распределяют электроэнергию и обслуживают банковские системы. Компьютеры являются основой множества автоматизированных систем обработки информации (АСОИ), осуществляющих хранение и обработку информации, предоставление ее потребителям, реализуя тем самым современные информационные технологии.

По мере развития и усложнения средств, методов и форм автоматизации процессов обработки информации повышается зависимость общества от степени безопасности используемых им информационных технологий, от которых порой зависит благополучие, а иногда и жизнь многих людей.

Актуальность и важность проблемы обеспечения безопасности информационных технологий обусловлены следующими причинами:

- резкое увеличение вычислительной мощности современных компьютеров при одновременном упрощении их эксплуатации;
- резкое увеличение объемов информации, накапливаемой, хранимой и обрабатываемой с помощью компьютеров и других средств автоматизации;
- сосредоточение в единых базах данных информации различного назначения и различной принадлежности;
- высокие темпы роста парка персональных компьютеров, находящихся в эксплуатации в самых разных сферах деятельности;
- резкое расширение круга пользователей, имеющих непосредственный доступ к вычислительным ресурсам и массивам данных;

- бурное развитие программных средств, не удовлетворяющих даже минимальным требованиям безопасности;

- повсеместное распространение сетевых технологий и объединение локальных сетей в глобальные;

- развитие глобальной сети Internet, практически не препятствующей нарушениям безопасности систем обработки информации во всем мире.

Безопасность АСОИ достигается принятием мер по обеспечению конфиденциальности и целостности обрабатываемой ею информации, а также доступности и целостности компонентов и ресурсов системы.

Конфиденциальность данных - это статус, предоставленный данным и определяющий требуемую степень их защиты. По существу конфиденциальность информации - это свойство информации быть известной только допущенным и прошедшим проверку (авторизированным) субъектам системы (пользователям, процессам, программам). Для остальных субъектов системы эта информация должна быть неизвестной.

Целостность информации обеспечивается в том случае, если данные в системе не отличаются в семантическом отношении от данных в исходных документах, т.е. если не произошло их случайного или преднамеренного искажения или разрушения.

Целостность компонента или ресурса системы - это свойство компонента или ресурса быть неизменными в семантическом смысле при функционировании системы в условиях случайных или преднамеренных искажений или разрушающих воздействий.

Доступность компонента или ресурса системы - это свойство компонента или ресурса быть доступным для авторизованных законных субъектов системы.

По цели воздействия различают три основных типа угроз безопасности АСОИ;

- угрозы нарушения конфиденциальности информации;
- угрозы нарушения целостности информации;
- угрозы нарушения работоспособности системы (отказы в обслуживании)

Угрозы нарушения конфиденциальности направлены на разглашение конфиденциальной или секретной информации. При реализации этих угроз информация становится известной лицам, которые не должны иметь к ней доступ. В терминах компьютерной безопасности угроза нарушения конфиденциальности имеет место всякий раз, когда получен несанкционированный доступ к некоторой закрытой информации, хранящейся в компьютерной системе или передаваемой от одной системы к другой.

Угрозы нарушения целостности информации, хранящейся в компьютерной системе или передаваемой по каналу связи, направлены на ее изменение или искажение, приводящее к нарушению ее качества или полному уничтожению. Целостность информации может быть нарушена умышленно злоумышленником, а также в результате объективных воздействий со стороны среды, окружающей систему. Эта угроза особенно актуальна для систем передачи информации -компьютерных сетей и систем телекоммуникаций. Умышленные нарушения целостности информации не следует путать с ее санкционированным изменением, которое выполняется полномочными лицами с обоснованной целью (например, таким изменением является периодическая коррекция некоторой базы данных).

Угрозы нарушения работоспособности (отказ в обслуживании) направлены на создание таких ситуаций, когда определенные преднамеренные действия либо снижают работоспособность АСОИ, либо

блокируют доступ к некоторым ее ресурсам. Например, если один пользователь системы запрашивает доступ к некоторой службе, а другой предпринимает действия по блокированию этого доступа, то первый пользователь получает отказ в обслуживании. Блокирование доступа к ресурсу может быть постоянным или временным.

Нарушения конфиденциальности и целостности информации, а также доступности и целостности определенных компонентов и ресурсов АСОИ могут быть вызваны различными опасными воздействиями на АСОИ.

Несанкционированный доступ (НСД) является наиболее распространенным и многообразным видом компьютерных нарушений. Суть НСД состоит в получении пользователем (нарушителем) доступа к объекту в нарушение правил разграничения доступа, установленных в соответствии с принятой в организации политикой безопасности. НСД использует любую ошибку в системе защиты и возможен при нерациональном выборе средств защиты, их некорректной установке и настройке. НСД может быть осуществлен как штатными средствами АСОИ, так и специально созданными аппаратными и программными средствами.

Перечислим основные каналы несанкционированного доступа, через которые нарушитель может получить доступ к компонентам АСОИ и осуществить хищение, модификацию и/или разрушение информации:

- все штатные каналы доступа к информации (терминалы пользователей, оператора, администратора системы; средства отображения и документирования информации; каналы связи) при их использовании нарушителями, а также законными пользователями вне пределов их полномочий;

- технологические пульты управления;

- линии связи между аппаратными средствами АСОИ;

- побочные электромагнитные излучения от аппаратуры, линий связи, сетей электропитания и заземления и др.

Из всего разнообразия способов и приемов несанкционированного доступа остановимся на следующих распространенных и связанных между собой нарушениях:

- перехват паролей;
- "маскарад";
- незаконное использование привилегий.

Перехват паролей осуществляется специально разработанными программами. При попытке законного пользователя войти в систему программа-перехватчик имитирует на экране дисплея ввод имени и пароля пользователя, которые сразу пересылаются владельцу программы-перехватчика, после чего на экран выводится сообщение об ошибке и управление возвращается операционной системе. Пользователь предполагает, что допустил ошибку при вводе пароля. Он повторяет ввод и получает доступ в систему. Владелец программы-перехватчика, получивший имя и пароль законного пользователя, может теперь использовать их в своих целях. Существуют и другие способы перехвата паролей.

"Маскарад" - это выполнение каких-либо действий одним пользователем от имени другого пользователя, обладающего соответствующими полномочиями. Целью "маскарада" является приписывание каких-либо действий другому пользователю либо присвоение полномочий и привилегий другого пользователя.

Примерами реализации "маскарада" являются:

- вход в систему под именем и паролем другого пользователя (этому "маскараду" предшествует перехват пароля);

- передача сообщений в сети от имени другого пользователя.

"Маскарад" особенно опасен в банковских системах электронных платежей, где неправильная идентификация клиента из-за "маскарада" злоумышленника может привести к большим убыткам клиента банка.

Незаконное использование привилегий. Большинство систем защиты устанавливают определенные наборы привилегий для выполнения заданных функций. Каждый пользователь получает свой набор привилегий: обычные пользователи - минимальный, администраторы - максимальный. Несанкционированный захват привилегий, например посредством "маскарада", приводит к возможности выполнения нарушителем определенных действий в обход системы защиты. Следует отметить, что незаконный захват привилегий возможен либо при наличии ошибок в системе защиты, либо из-за халатности администратора при управлении системой и назначении привилегий.

Особо следует остановиться на угрозах, которым могут подвергаться компьютерные сети. Основная особенность любой компьютерной сети состоит в том, что ее компоненты распределены в пространстве. Связь между узлами (объектами) сети осуществляется физически с помощью сетевых линий связи и программно с помощью механизма сообщений. При этом управляющие сообщения и данные, пересылаемые между объектами сети, передаются в виде пакетов обмена. При вторжении в компьютерную сеть злоумышленник может использовать как пассивные, так и активные методы вторжения [55].

При пассивном вторжении (перехвате информации) нарушитель только наблюдает за прохождением информации по каналу связи, не вторгаясь ни в информационный поток, ни в содержание передаваемой информации. Как правило, злоумышленник может определить пункты назначения и идентификаторы либо только факт прохождения сообщения,

его длину и частоту обмена, если содержимое сообщения не распознаваемо, т.е. выполнить анализ трафика (потока сообщений) в данном канале.

При активном вторжении нарушитель стремится подменить информацию, передаваемую в сообщении. Он может выборочно модифицировать, изменить или добавить правильное или ложное сообщение, удалить, задержать или изменить порядок следования сообщений. Злоумышленник может также аннулировать и задержать все сообщения, передаваемые по каналу. Подобные действия можно квалифицировать как отказ в передаче сообщений.

Компьютерные сети характерны тем, что кроме обычных локальных атак, осуществляемых в пределах одной системы, против объектов сетей предпринимают так называемые удаленные атаки, что обусловлено распределенностью сетевых ресурсов и информации. Злоумышленник может находиться за тысячи километров от атакуемого объекта, при этом нападению может подвергаться не только конкретный компьютер, но и информация, передающаяся по сетевым каналам связи. Под удаленной атакой понимают информационное разрушающее воздействие на распределенную компьютерную сеть, программно осуществленное по каналам связи .

В табл. 1.1 показаны основные пути реализации угроз безопасности АСОИ при воздействии на ее компоненты. Конечно, табл. 1.1 дает самую общую картину того, что может произойти с системой.

Таблица 1

Пути реализации угроз безопасности АСОИ

Объекты воздействия	Нарушение конфиденциальности информации	Нарушение целостности информации	Нарушение работоспособности системы
Аппаратные средства	НСД - подключение; использование ресурсов; хищение носителей	НСД -подключение; использование ресурсов; модификация, изменение режимов	НСД -изменение режимов; вывод из строя; разрушение
Программное обеспечение	НСД -копирование; хищение; перехват	НСД, внедрение "троянского коня", "вирусов", "червей"	НСД -искажение; удаление; подмена
Данные	НСД- копирование; хищение; перехват	НСД -искажение; модификация	НСД -искажение; удаление; подмена
Персонал	Разглашение; передача сведений о защите; халатность	"Маскарад"; вербовка; подкуп персонала	Уход с рабочего места; физическое устранение

Для защиты от указанных вредоносных программ необходимо применение ряда мер:

- исключение несанкционированного доступа к исполняемым файлам;
- тестирование приобретаемых программных средств;

- контроль целостности исполняемых файлов и системных областей;
- создание замкнутой среды исполнения программ.

1.2 Основный подход и методы защити информации

Управление персоналом

С одной стороны, общепризнанно, что это одна из важнейших сторон обеспечения информационной безопасности предприятия. С другой — именно эта сторона может вызвать у вас чувство определенного дискомфорта, связанного с кажущимся недоверием к вам со стороны администрации организации или предприятия. В солидных организациях управление персоналом начинается еще до приема нового сотрудника на работу. Чем критичнее должность с точки зрения обеспечения информационной безопасности, тем тщательнее вас будут проверять, наводя справки самым различным образом (возможно через бывших сослуживцев, знакомых и пр., вплоть до правоохранительных органов). При приеме на работу от вас могут потребовать, например, подписать различные соглашения о неразглашении конфиденциальной информации. В ходе работы может проводиться проверка данных, записанных на вашем компьютере, за вами может вестись контроль с помощью видеокамер внутреннего наблюдения и т.д.

В общем случае ко всему этому нужно относиться с определенной долей понимания и здравого смысла, однако не стоит забывать и о собственных интересах.

С точки зрения информационной безопасности существует два общих принципа, используемых при управлении кадрами: разделение обязанностей и минимизация привилегий. Принцип разделения обязанностей состоит в таком распределении ролей и ответственности, чтобы один человек не смог нарушить критически важный для организации процесс. Принцип

минимизации привилегий служит развитием предыдущего и предписывает выделять пользователям только те права доступа к информации, которые необходимы им для выполнения служебных обязанностей.

Смысл этих принципов очевиден — уменьшить ущерб от случайных или умышленных некорректных действий пользователей. Важно, однако, как конкретно реализованы эти принципы в организации или на предприятии и не окажетесь ли вы «крайним», когда будут искать виновника того или иного нарушения информационной безопасности.

Что можно сделать, чтобы хоть как-то защитить свои интересы? Формальных правил здесь, к сожалению нет, но можно дать несколько практических советов.

Во-первых, следует обязательно познакомиться с должностной инструкцией или другим документом, определяющим ваши права, обязанности и ответственность как сотрудника конкретной организации, в том числе и с точки зрения информационной безопасности. Если такого документа нет, желательно, чтобы эти вопросы явно были отражены в контракте или аналогичном документе. Тем самым вы сможете в какой-то мере избежать необоснованных претензий.

Во-вторых, можно потребовать, чтобы вас ознакомили (в пределах ваших полномочий) с основами общей политики безопасности, принятой на предприятии.

В-третьих, имеет смысл позаботиться о формальном протоколировании всех действий, связанных с предоставлением вам тех или иных полномочий доступа к информации и, что особенно важно, с передачей этих полномочий другим лицам (на время вашего отпуска, командировки, болезни и пр.). В последнем случае следует обязательно удостовериться, что переданные полномочия изъяты у другого лица после вашего возвращения.

И, наконец, стоит обратить внимание на то, могут ли получить доступ *к* конфиденциальной информации, с которой вы работаете, сотрудники внешних организаций, проводящих, например, обновление версий программного обеспечения, установку новых систем, профилактическое обслуживание, ремонт и т.п. В любом случае в ваших интересах, чтобы время проведения их работ протоколировалось тем или иным образом.

Физическая защита

Очевидно, что информационная безопасность компьютерной системы самым непосредственным образом зависит от окружения, в котором она работает. В частности, важное значение здесь имеют меры так называемой физической защиты, в состав которых входят:

- физическое управление доступом;
- противопожарные меры;
- защита поддерживающей инфраструктуры;
- защита от перехвата данных;
- защита мобильных систем.

С какой точки зрения эти меры могут интересовать вас как пользователя, работающего с конфиденциальной информацией? Ответ простой — без соблюдения администрацией этих мер вы просто не можете нести ответственность за сохранность информации, с которой работаете.

Задача мер физического управления доступом — контроль и при необходимости ограничение входа и выхода сотрудников и посетителей как на общую территорию организации, так и в ее отдельные режимные помещения, например те, где расположены серверы, коммуникационная аппаратура, хранилища машинных носителей и т.п. Средства физического управления доступом известны давно — это охрана, двери с замками, перегородки, телекамеры, датчики движения и многое другое.

Ваша задача — убедиться в соответствии мер физической защиты предъявляемым к вам требованиям по работе с конфиденциальной информацией. Приведем несколько самых простых примеров. Может ли экран монитора, на котором вы работаете с конфиденциальной информацией, просматриваться посетителями или сотрудниками, не имеющими соответствующего допуска? Как хранятся конфиденциальные документы во время вашего отсутствия, могут ли получить к ним доступ посторонние лица? Таких вопросов можно задать еще очень много. Важны их последствия. Например, если в организации, где вы работаете, принято хранить конфиденциальные документы просто в ящике стола, установленного в комнате, где работают еще несколько человек, и приходят посетители, то брать на себя ответственность за их сохранность вряд ли будет разумным. Во избежание возможных неприятностей эти вопросы необходимо поставить перед непосредственным руководителем или службой безопасности.

Противопожарные меры и меры по защите поддерживающей инфраструктуры призваны свести к минимуму ущерб, вызванный огнем или авариями электропитания, водопровода, отопления и т.п. Здесь можно руководствоваться определенными соображениями здравого смысла. Есть инструкция по противопожарным мерам? Выполняйте ее неукоснительно, о нарушениях незамедлительно сообщайте руководству, иначе виновным можете оказаться именно вы. «Гуляет» или часто пропадает напряжение в сети — обязательно ставьте вопрос о приобретении источников бесперебойного питания, иначе можете потерять ценную информацию или нарушить ее целостность. Ветхие трубы отопления или водопровода (в том числе и над вашей комнатой) — бейте тревогу.

Как мы уже говорили во второй главе, перехват данных может осуществляться самыми различными, в том числе и очень изощренными способами. Самостоятельно решить проблему защиты от перехвата вы,

разумеется, не сможете. Однако, если руководство постоянно подчеркивает конфиденциальность данных, с которыми вы работаете, имеет смысл явно поставить перед ним эту проблему. В принципе здесь может применяться экранирование помещения, использоваться различные зачумляющие устройства и т.п., выбор которых уж совсем не входит в ваши обязанности.

Если конфиденциальная информация записана на ноутбуке, который вы берете на деловые встречи и домой, то вполне возможный способ ее утечки — кража ноутбука. Здесь нужно обязательно обратить внимание, во-первых, на обеспечение сохранности ноутбука, а во-вторых — на защиту хранящейся на нем информации программно-техническими методами.

Поддержание работоспособности

Из самых общих соображений понятно, что эти меры имеют важное значение для нормального функционирования информационно-вычислительной системы. Организация их выполнения — задача соответствующей специализированной службы. На что вы должны обратить внимание, как пользователь? Здесь можно выделить следующие группы вопросов.

Во-первых, важно, как организовано (да и организовано ли вообще) на вашем предприятии так называемое резервное копирование. Оно необходимо для восстановления программ и данных после возможных умышленных нарушений информационной безопасности или после аварий. Восстановлением программ должны заниматься специализированные службы, а вот меры по обеспечению возможности восстановления вашей информации зачастую придется принимать непосредственно вам. Хорошо, если регулярность резервного копирования определяется должностной инструкцией или еще каким-то директивным документом. Тогда стоит лишь посмотреть, соответствуют ли эти требования реальному положению вещей. Например, документ может устареть или быть ориентированным на другой

вид информации, в результате резервное копирование будет проводиться слишком часто, слишком редко или в нерациональном составе. А вот если такого документа нет, следует обязательно согласовать правила регулярного резервного копирования с руководством и службой информатизации. Практика показывает, что, несмотря на некоторую обременительность резервного копирования, пренебрегать им крайне опасно, можно полностью потерять всю информацию, накопленную за длительное время.

Во-вторых, важно, как организована на предприятии поддержка пользователей. По ходу работы даже у опытного пользователя могут возникать вопросы к системным администраторам и другим специалистам по работе конкретных программ и систем. В разрешении этих вопросов и состоит поддержка пользователей. В контексте данной книги очень важно, чтобы в общем потоке этих вопросов выявлялись проблемы, потенциально связанные с информационной безопасностью.

Работоспособность системы может быть нарушена непреднамеренными действиями пользователя. В этом плане мы настоятельно не рекомендуем вам самостоятельно устанавливать на компьютер новые программы или проводить обновление текущих версий. Пусть этим занимаются специалисты. Дело не только в том, что при этом вы можете занести в систему компьютерный вирус. (Это достаточно известный факт, хотя пользователи с упорством, достойным лучшего применения, продолжают наступать здесь на одни и те же грабли). При установке новой программы вы можете незаметно для себя сменить набор драйверов, настройки системных программ и т.п., что в принципе может повлечь нарушение работоспособности системы в целом или нарушение работы ее механизмов информационной безопасности.

Планирование восстановительных работ

Общее планирование восстановительных работ, как и реализация всех рассмотренных выше составляющих организационных мер обеспечения информационной безопасности — задача специализированной службы. Тем не менее и вы лично должны быть готовы к проведению работ по восстановлению вашей информации после аварии или умышленного нарушения целостности данных.

Прежде всего отметим формальный момент: нужно в общих чертах представлять себе весь план восстановительных работ организации и детально — свое место в этих работах.

С практической точки зрения не лишним будет составить и свой план восстановления, рассмотрев основные возможные проблемы, например, что вы будете делать, если выйдет из строя компьютер, на котором вы обычно работаете, если вирус разрушит информацию на вашем винчестере, если выйдет из строя сервер сети или нарушится передача информации по линиям связи. Важность наличия такого плана объясняется и тем обстоятельством, что восстановление часто проводится в условиях катастрофического дефицита времени, повышенной занятости системного персонала, зачастую в нервной обстановке, когда получить консультацию будет просто некогда или не у кого.

Следует иметь в виду, что восстановление работы с информацией сразу в полном объеме может оказаться практически невозможным. Поэтому имеет смысл подумать о том, как восстановить основные критические для работы организации функции хотя бы временным путем (использование другого компьютера или другого сегмента сети, обходных или временных каналов связи, частичное восстановление информации по резервным копиям, ручная обработка и пр.).

Программно-технические методы обеспечения информационной безопасности

Идентификация и аутентификация

Эти, на первый взгляд, достаточно формальные термины фактически являются основой всех программно-технических средств информационной безопасности, поскольку остальные средства, рассматриваемые ниже, рассчитаны на работу с уже идентифицированными пользователями. Часто процессы идентификации и аутентификации взаимосвязаны, поэтому возможна некоторая путаница этих понятий. Давайте остановимся на их смысле подробнее, опуская пока технические детали.

Идентификация — процесс, позволяющий установить имя пользователя. Хорошим примером здесь, в частности, может служить вручение визитной карточки, где указаны имя, должность и другие атрибуты конкретного лица. Но как убедиться, что визитная карточка принадлежит действительно тому человеку, который называет ее своей?

Здесь потребуется уже процедура аутентификации. Аутентификация предполагает выполнение процесса проверки подлинности введенного в систему имени пользователя**.** На бытовом уровне аутентификация может, например, осуществляться с помощью фотографии. Еще одним примером аутентификации может служить узнавание голоса при звонках по телефону — вряд ли вы будете продолжать беседу с человеком, назвавшимся по телефону знакомой фамилией, но говорящим незнакомым голосом и с другими интонациями.

Средства идентификации и аутентификации могут и объединяться. Всем известным примером здесь может быть служебное удостоверение, где приведены и данные для идентификации (фамилия, должность и пр.) и данные для аутентификации (фотография). Важно отметить, что и сами средства идентификации и аутентификации могут иметь некоторые признаки, подтверждающие их подлинность. На удостоверении, например, это печати, подписи и, при необходимости, другие признаки защиты от подделок.

Предыдущий пример приведен неслучайно. В информационных технологиях способы идентификации и аутентификации являются своеобразным служебным удостоверением пользователя, обеспечивающим его доступ в информационное пространство организации в целом или отдельные разделы этого пространства.

Весьма значительное число используемых в настоящее время способов идентификации и аутентификации пользователя информационно-вычислительных систем можно разделить на следующие основные группы:

- парольные методы;
- методы с применением специализированных аппаратных средств;
- методы, основанные на анализе биометрических характеристик пользователя.

Наибольшее распространение получили парольные методы, что объясняется относительной простотой их реализации. Смысл этих методов заключается в том, что для входа в систему пользователь вводит два кода: свое условное имя (идентификация) и уникальный, известный только ему одному код-пароль для аутентификации. При правильном использовании парольные схемы могут обеспечить приемлемый для многих организаций уровень безопасности. Тем не менее по совокупности характеристик они считаются сегодня самым слабым средством аутентификации. Дело в том, что надежность паролей очень сильно зависит от «человеческого фактора». Она изначально основана на способности людей помнить пароль и хранить его в тайне. Однако, чтобы пароль был запоминающимся, его зачастую делают простым, а простой пароль нетрудно угадать, особенно если знать пристрастия данного пользователя. Сложный пароль часто записывают на бумажке, которую не так уж трудно найти злоумышленнику. Кроме того, пароль можно подсмотреть при вводе, подобрать с помощью специальных программ и т.п.

Вторая из выделенных схем идентификации и аутентификации предполагает использование специальных устройств — магнитных карт, смарт-карт, так называемых таблеток, токенов и др., на которых записана уникальная информация. Эти методы отличаются большей устойчивостью, однако потребуют от вас в рабочее время постоянного ношения соответствующего блока. В принципе это необременительно, поскольку сейчас такие устройства по размеру и весу не больше обычного брелка для ключей (да, как правило, они и носятся таким же образом). Однако существует вполне реальная возможность потери или кражи аппаратного идентификатора, его можно просто забыть дома и т.п. Поэтому в ряде организаций практикуется получение аппаратных идентификаторов утром перед началом работы и возврат их на хранение перед уходом с предприятия с соответствующим документированием этой процедуры.

Наиболее перспективным в настоящее время считается использование средств идентификации пользователя по биометрическим признакам — отпечаток пальца, рисунок радужной оболочки глаз, отпечаток ладони и др. Эти методы обладают достаточно высокой надежностью и в то же время не требуют от пользователя запоминания и хранения в тайне сложных паролей или заботы о сохранности аппаратного идентификатора. Многим из вас, возможно, эти методы знакомы пока лишь по приключенческим кинофильмам, однако развитие информационных технологий ведет к тому, что стоимость этих средств становится доступной для большинства организаций. Правда, и эти средства не лишены недостатков. Не говоря уже о возможности использования «мертвой руки» или «мертвого пальца», здесь можно упомянуть о проблемах с идентификацией, возникающих из-за изменения радужной оболочки под воздействием некоторых лекарств или связанных с изменениями в кожном покрове под воздействием высокой или низкой температуры воздуха. Например, если вы вошли в помещение с

замерзшими руками, система аутентификации по отпечаткам пальцев может вас не опознать.

С какими из рассмотренных методов придется иметь дело на практике? В принципе вопрос о применимости того или иного средства решается в зависимости от выявленных угроз и технических характеристик защищаемого объекта. Здесь обычно выбирается компромисс между надежностью, доступностью по цене, удобством использования и администрирования средств идентификации и аутентификации. Наверняка большинство из наших читателей уже тем или иным способом использовали парольную защиту. Вполне возможно, что в будущем вы столкнетесь и с аппаратными идентификаторами и с идентификацией по биометрическим характеристикам. Некоторые примеры реализации этих методов мы рассмотрим во второй части книги.

Разграничение доступа

Итак, с помощью средств идентификации и аутентификации вы получили доступ в систему. Теперь в дело вступают средства логического управления доступом. В принципе их задача примерно та же, что и у средств физического управления доступом, которые мы рассмотрели выше. Средства логического управления доступом тоже контролируют возможность попадания пользователя в тот или иной раздел информации, хранящейся в системе, только они реализуются программным путем. Логическое управление доступом — это основной механизм многопользовательских систем, призванный обеспечить конфиденциальность и целостность информации.

Нужно сказать, что тема логического управления доступом — одна из сложнейших в области информационной безопасности. Поскольку вам наверняка не придется разбираться в ней «изнутри», мы отметим лишь следующее. Схему управления доступом принято характеризовать так

называемой матрицей доступа, в строках которой перечислены субъекты, в столбцах — объекты, а в клетках, расположенных на пересечении строк и столбцов, записаны разрешенные виды доступа и дополнительные условия (например, время и место действия).

Удобной надстройкой над средствами логического управления доступом является ограничивающий интерфейс, когда пользователя лишают самой возможности попытаться совершить несанкционированные действия, включив в число видимых ему объектов только те, к которым он имеет доступ. Подобный подход обычно реализуют в рамках системы меню и пользователю показывают лишь допустимые варианты выбора.

Протоколирование и аудит

Эти термины невольно вызывают ассоциации с правоохранительными органами и финансовой деятельностью. В информационной безопасности они имеют свою специфику.

Под протоколированием понимается сбор и накопление информации о событиях, происходящих в информационно-вычислительной системе. У каждой программы есть свой набор возможных событий, но в любом случае их можно подразделить на внешние — вызванные действиями других программ или оборудования, внутренние — вызванные действиями самой программы, и клиентские — вызванные действиями пользователей и администраторов.

Аудит — это анализ накопленной информации, проводимый оперативно, почти в реальном времени, или периодически.

Реализация протоколирования и аудита преследует следующие главные цели:

- обеспечение подотчетности пользователей и администраторов;

- обеспечение возможности реконструкции последовательности событий;

- обнаружение попыток нарушений информационной безопасности;

- предоставление информации для выявления и анализа проблем.

Вы, как пользователь, не в состоянии вмешаться в процесс протоколирования, а в процессе аудита будете участвовать скорее всего только тогда, когда по результатам аудита к вам будут претензии. Тем не менее не стоит забывать, что в хорошо сделанной системе фиксируются все ваши попытки доступа к информации и практически все виды действий, которые вы над этой информацией производите.

В принципе обеспечение подобной подотчетности считается одним из средств сдерживания попыток нарушения информационной безопасности — в этих условиях труднее замести следы, поэтому злоумышленнику нужно принимать какие-то дополнительные меры, а просто любопытные побоятся предпринимать несанкционированные действия. Если есть основания подозревать какого-то конкретного пользователя, его работу можно рассмотреть «под микроскопом» — регистрировать его действия особенно детально, например, до каждого нажатия клавиши. Последующая реконструкция событий позволяет выявить слабости в защите, найти виновника, определить способ устранения проблемы и вернуться к нормальной работе. Тем самым в определенной степени обеспечивается целостность информации.

Можно, однако, упомянуть и о другой стороне медали — если подобными средствами воспользуется злоумышленник, который тем или иным образом получил соответствующие полномочия, то работа системы будет перед ним как на ладони.

Криптографическое преобразование данных

Криптография, или шифрование — одна из самых наукоемких и до настоящего времени одна из самых закрытых областей информационной безопасности. Во многих отношениях она занимает центральное место среди программно-технических средств безопасности, являясь основой реализации многих из них и, если можно так выразиться, последним барьером, предотвращающим несанкционированный доступ *к* информации.

Поскольку разработка криптографических средств наверняка не входит в ваши обязанности, но использовать их вам скорее всего придется, мы ограничимся здесь самыми общими представлениями.

В современной криптографии используются два основных метода шифрования — симметричное и асимметричное.

В симметричном шифровании один и тот же ключ используется и для шифровки, и для расшифровки сообщений. Существуют весьма эффективные методы симметричного шифрования. Основным недостатком симметричного шифрования является то, что секретный ключ должен быть известен и отправителю, и получателю. С одной стороны, это ставит проблему безопасной пересылки ключей при обмене сообщениями.

С другой — получатель, имеющий шифрованное и расшифрованное сообщение, не может доказать, что он получил его от конкретного отправителя, поскольку такое же сообщение он мог сгенерировать и сам.

В асимметричных методах применяются два ключа. Один из них, несекретный, используется для шифровки и может без всяких опасений передаваться по открытым каналам, другой — секретный, применяется для расшифровки и известен только получателю.

Асимметричные методы шифрования позволяют реализовать так называемую электронную подпись, или электронное заверение сообщения. Идея состоит в том, что отправитель посылает два экземпляра сообщения —

открытое и дешифрованное его секретным ключом (здесь следует учитывать, что дешифровка незашифрованного сообщения на самом деле есть форма шифрования). Получатель может зашифровать с помощью открытого ключа отправителя дешифрованный экземпляр и сравнить с открытым. Если они совпадут, личность и подпись отправителя можно считать установленными.

Существенным недостатком асимметричных методов является их низкое быстродействие, поэтому их приходится сочетать с симметричными. Так, для решения задачи рассылки ключей сообщение сначала симметрично шифруют случайным ключом, затем этот ключ шифруют открытым асимметричным ключом получателя, после чего сообщение и ключ отправляются по сети.

Обратим внимание на то, что при использовании асимметричных методов необходимо иметь гарантию подлинности пары (имя, открытый ключ) адресата. Для решения этой задачи вводится понятие сертификационного центра, который заверяет справочник имен/ключей своей подписью.

В любом случае вам нужно иметь в виду, что ахиллесовой пятой любого метода шифрования является секретный ключ. При его случайном или намеренном раскрытии все усилия по шифрованию пропадут даром. Поэтому определенную проблему здесь составляет надежное хранение закрытых ключей.

Экранирование

С развитием сетевых технологий все большую актуальность приобретает защита от случайных или намеренных воздействий из внешних сетей (например, Интернет), с которыми взаимодействует сеть предприятия. Для этой цели используются различные разновидности так называемых

межсетевых экранов, а сам процесс защиты получил название экранирования. Если опустить технические подробности, то межсетевой экран — это специализированная программная система, ограничивающая возможность передачи информации как из внешней сети в сеть предприятия, так и из сети предприятия во внешнюю среду. Помимо функций разграничения доступа, экраны осуществляют также протоколирование информационных обменов.

Говоря об межсетевых экранах, нельзя не остановиться на одном из общих прагматических аспектов защиты информации. Любое подключение к сети в потенциале предоставляет «лазейку» для несанкционированного доступа к информации. Поэтому для надежного хранения конфиденциальных данных используются автономные, не подключаемые к сети системы. Средства защиты информации в сети имеет смысл использовать, только если эта информация как раз и предназначена для сетевого применения — например, в электронной коммерции, на информационных серверах Интернет и т.п.

2. ОБЕСПЕЧЕНИЕ ЦЕЛОСТНОСТИ ИНФОРМАЦИИ С ИСПОЛЬЗОВАНИЕМ ИЗБЫТОЧНЫХ КОДОВ

2.1 Классификации избыточных кодов и их параметры

1-рис. Классификации избыточных кодов

Циклические коды являются подклассом в классе линейных кодов, удовлетворяющим дополнительному сильному структурному требованию. В силу этой структуры поиск хороших кодов, контролирующих ошибки, в классе циклических кодов оказался наиболее успешным. При этом в качестве математического аппарата, облегчающего поиск хороших кодов, была использована теория полей Галуа. Вне класса циклических кодов теория полей Галуа помогает мало; большинство завершённых построений, использующих идеи этой теории, относятся к циклическим кодам.

Важность циклических кодов обусловлена также, тем, что заложенные в основу их определения идеи теории полей Галуа приводят к процедурам кодирования и декодирования, эффективным как с алгоритмической, так и с вычислительной точки зрения. Поэтому нужно знать свойства полей Галуа.

Полем называется множество с двумя определёнными на нём операциями: сложением и умножением.

Существуют следующие примеры полей:

1. R: множество вещественных чисел;
2. C: множество комплексных чисел;
3. Q: множество рациональных чисел.

Все эти поля содержат бесконечное число элементов. Но нас интересуют поля с конечным числом элементов. Конечное поле, названное в честь французского математика Галуа обозначается GF(q), где q-это конечное множество элементов обладающих свойством поля.

Множество p(x) и элементы поля GF(p^m) обладают целом рядом свойств, используемых при построении и описании циклических кодов.

Свойство 1: Элементы поля GF(p^m), отличные от нуля образуют мультипликативную группу порядка p^m–1. Поэтому для любого элемента поля имеет место равенство $a^{p\ -1}$=1.

Это свойство выполняется и для нулевого элемента поля. Очевидно, что ненулевые элементы поля являются корнями многочлена x^{p-1}, а все элементы поля являются корнями многочлена x^p–x.

Свойство 2: Всегда в поле GF(p^m) существует первообразный элемент □, элемент порядка p^m–1. Любой ненулевой элемент поля может быть представлен как степень одного и того же первообразного элемента □, т.е. мультипликативная группа поля Галуа циклична.

Важную роль при кодировании циклическими кодами имеют следующие свойства:

Свойство 3: Любой приводимый по модулю p многочлен p(x) степени m, если он существует, есть делитель по этому модулю двучлена x^p–x.

Следующие свойства позволяют определить элементы поля, являющиеся корнями многочлена p(x), если известно, что один из этих корней □.

Свойство 4: В простом поле GF(p) имеет место равенство $(a+b)^p=a^p+b^p$.

Свойство 5: Для простого модуля p справедливо сравнение

$[p(x)]^p=p(x^p)(\text{mod } p)$,

где p(x)–произвольный многочлен, коэффициенты которого принадлежат простому полю GF(p).

Свойство 6: Если элемент □ поля $GF(p^m)$ есть корень неприводимого по модулю p многочлена p(x) степени m, то остальными корнями p(x) будут элементы $□^p$, $□^{p\,□}$, ….,$□^{p□}$.

Следующие свойства имеют важную роль в теории циклических кодов.

Свойство 7: Если k-делитель n, то многочлен x^k–1 является делителем многочлена x^{n-1}.

Свойство 8: Для того чтобы неприводимый по модулю многочлен $p_1(x)$ степени k является делителем двучлена $x^{p\ -1}$, степень k должна быть делителем числа m, и наоборот.

Свойство 9: Поля Галуа $GF(p^m)$, образованные различными неприводимыми примитивными многочленами степени m изоморфны, т.е. для любого простого числа p и любого примитивного многочлена p(x) степени m существует только одно поле Галуа $GF(p^m)$.

Выше мы рассмотрели все необходимые свойства полей Галуа, необходимых для изучения циклических кодов. Приступим теперь к изучению самих циклических кодов.

Циклическим кодом называется линейный блоковый (n,k)–код, который характеризуется свойством цикличности, т.е. сдвиг влево на один шаг любого разрешённого кодового слова даёт также разрешенное кодовое слово, принадлежащее этому же коду и у которого, множество кодовых слов представляется совокупностью многочленов степени (n–1) и менее, делящихся на некоторый многочлен g(x) степени r= n-k , являющийся

сомножителем двучлена $x^{x}+1$. Многочлен g(x) называется порождающим. Как следует из определения, в циклическом коде кодовые слова представляются в виде многочленов: a(x)=a+…+a $_{x}x^{x}$где, n–длина кода, a $_{x}$-коэффициент из поля GF(p).

2.3 CRC коды, алгоритмы и их использование

Алгоритм CRC

Алгоритм CRC базируется на свойствах деления с остатком двоичных многочленов, то есть многочленов над конечным полем GF(2). Значение CRC является по сути остатком от деления многочлена, соответствующего входным данным, на некий фиксированный порождающий многочлен (полином).

Каждой конечной последовательности битов $a_0, a_1, \ldots, a_{N-1}$ взаимнооднозначно сопоставляется двоичный многочлен $\sum_{n=0}^{N-1} a_n x^n$, последовательность коэффициентов которого представляет собой исходную последовательность. Например, последовательность битов 0,1,0,1,1,0,1 соответствует многочлену

$$P(x) = 1 \cdot x^6 + 0 \cdot x^5 + 1 \cdot x^4 + 1 \cdot x^3 + 0 \cdot x^2 + 1 \cdot x^1 + 0 \cdot x^0 = x^6 + x^4 + x^3 + x^1$$

Нетрудно видеть, что количество различных многочленов степени меньшей N равно 2^N, что совпадает с числом всех двоичных последовательностей длины N.

Значение CRC с порождающим многочленом $G(x)$ степени N определяется как битовая последовательность длины N, представляющая многочлен $R(x)$, получившийся в остатке при делении многочлена $P(x)$, представляющего входной поток бит, на многочлен $G(x)$:

$$R(x) = P(x) \cdot x^N \bmod G(x)$$

где

$R(x)$ — многочлен, представляющий значение CRC.

$P(x)$ — многочлен, коэффициенты которого представляют входные данные.

$G(x)$ — порождающий многочлен.

$N = \deg G(x)$ — степень порождающего многочлена.

Умножение x^N осуществляется приписыванием N нулевых битов к входной последовательности, что улучшает качество хеширования для коротких входных последовательностей.

При делении с остатком степень многочлена-остатка строго меньше степени многочлена-делителя, то есть при делении на многочлен $G(x)$ степени N можно получить 2^N различных остатков от деления. При «правильном» выборе порождающего многочлена $G(x)$, остатки от деления на него будут обладать нужными свойствами хеширования — хорошей перемешиваемостью и быстрым алгоритмом вычисления. Второе обеспечивается тем, что степень порождающего многочлена обычно пропорциональна длине байта или машинного слова (например 8, 16 или 32).

Операция деления на примитивный полином также эквивалентна следующей схеме:

Пусть выбран примитивный полином, задающий цикл де Брейна 0010111001011100… и блок данных 0111110, построена таблица, верхняя строка заполнена блоком данных, а нижние строки — смещения на 0,1,2 бит цикла де Брейна

$$\begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 \end{pmatrix}$$

Тогда контрольная сумма будет равна операции XOR тех столбцов, над которыми в верхней строке расположена 1. В этом случае, 010 xor 101 xor 011 xor 111 xor 110 = 101 (CRC).

Ниже представлены реализации получения некоторых CRC для многочленов степени 8 (CRC-8), 16 (CRC-16) и 32 (CRC-32).

Классификация реализаций алгоритмов CRC

Для точного указания, как именно рассчитывается CRC, чаще всего приходится полностью приводить алгоритм её расчёта.

На самом деле достаточно указать ряд параметров, точно описывающих конкретный частный алгоритм CRC (если это на самом деле CRC).

В модели алгоритма CRC Rocksoft, получившей некоторое хождение, используются следующие параметры:

Name: Это имя, присвоенное данному алгоритму.

Width: Степень алгоритма, выраженная в битах. Она всегда на единицу меньше длины полинома, но равна его степени.

Poly: Собственно полином. Это битовая величина, которая для удобства может быть представлена шестнадцатеричным числом. Старший бит при этом опускается (он всегда 1). Например, если используется полином 10110, то он обозначается числом «06h». Важной особенностью данного параметра является то, что он всегда представляет собой необращенный полином, младшая часть этого параметра во время вычислений всегда является наименее значащими битами делителя.

Init: Этот параметр определяет исходное содержимое регистра на момент запуска вычислений. Данный параметр указывается шестнадцатеричным числом.

RefIn(Revert): Логический параметр. Если он имеет значение False, байты сообщения обрабатываются, начиная с 7 бита, который считается наиболее значащим, а наименее значащим считается бит 0 (сдвиг налево). Если параметр имеет значение True («Истина»), то каждый байт перед обработкой обращается (сдвиг направо).

RefOut: Логический параметр. Если он имеет значение False, то конечное содержимое регистра сразу передается на стадию XorOut, в противном случае, когда параметр имеет значение True, содержимое регистра обращается перед передачей на следующую стадию вычислений. в приведённых алгоритмах, по-видимому, False).

XorOut: W битное значение, обозначаемое шестнадцатеричным числом. Оно комбинируется с конечным содержимым регистра (после стадии RefOut), прежде чем будет получено окончательное значение контрольной суммы.

Check: Это поле, собственно, не является частью определения алгоритма, данное поле служит контрольным значением, которое может быть использовано для слабой проверки правильности реализации алгоритма. Поле содержит контрольную сумму, рассчитанную для ASCII строки «123456789» (шестнадцатеричные значение «313233…»).

После определения всех этих параметров, можно точно описать особенности применённого CRC алгоритма.

Примеры спецификаций некоторых алгоритмов CRC:

Таблица 2.

Name : CRC 16 Width : 16 Poly : 8005 Init : 0000 RefIn : True RefOut : True XorOut : 0000 Check : BB3D	Name : CRC 16/CITT Width : 16 Poly : 1021 Init : FFFF RefIn : False RefOut : False XorOut : 0000	Name : XMODEM Width : 16 Poly : 8408 Init : 0000 RefIn : True RefOut : True XorOut : 0000
Name : ARC Width : 16 Poly : 8005 Init : 0000 RefIn : True RefOut : True XorOut : 0000	Name : CRC 32 Width : 32 Poly : 04C11DB7 Init : FFFFFFFF RefIn : True RefOut : True XorOut : FFFFFFFF Check : CBF43926	

Наиболее используемые и стандартизованные CRC

В то время, как циклические избыточные коды являются частью стандартов, сами они не стандартизированы в плане адаптации одного алгоритма для конкретной степени полинома: существуют три описания полинома для CRC-12[1], десять противоречивых определений CRC-16 и четыре - CRC-32[2]. Наиболее широко используемые полиномы не являются наиболее эффективными из всех возможных. Между 1993 и 2004, Koopman, Castagnoli и другие исследовали пространство полиномов до 16 бит включительно[1], и 24 и 32 бит,[3][4] найдя примеры, которые дают гораздо большую производительность (в смысле Hamming distance для данных заданного размера), чем полиномы предшествовавших протоколов, и опубликовав лучшие из них с целью улучшения способности к выявлению ошибок будущих стандартов. В настоящее время в протоколе ISCSI используется один из результатов этого исследования.

Самый популярный, рекомендуемый IEEE полином для CRC-32, используемый в Ethernet, FDDI и др., является генератором кода Хемминга и

был выбран, основываясь на его производительности и способности выявления ошибок передачи данных.

Тип	Полиноминал	Представление нормальный/полностью измененный/перемена взаимных
CRC-1	$x + 1$ (most hardware; also known as *parity bit*)	0x1 / 0x1 / 0x1
CRC-4-ITU	$x^4 + x + 1$ (ITU G.704[6])	0x3 / 0xC / 0x9
CRC-5-EPC	$x^5 + x^3 + 1$ (Gen 2 RFID[7])	0x09 / 0x12 / 0x14
CRC-5-ITU	$x^5 + x^4 + x^2 + 1$ (ITU G.704[8])	0x15 / 0x15 / 0x1A
CRC-5-USB	$x^5 + x^2 + 1$ (USB token packets)	0x05 / 0x14 / 0x12
CRC-6-ITU	$x^6 + x + 1$ (ITU G.704[9])	0x03 / 0x30 / 0x21
CRC-7	$x^7 + x^3 + 1$ (telecom systems, ITU-T G.707[10], ITU-T G.832[11], MMC, SD)	0x09 / 0x48 / 0x44
CRC-8-ATM	$x^8 + x^2 + x + 1$ (ATM HEC)	0x07 / 0xE0 / 0x83
CRC-8-CCITT	$x^8 + x^7 + x^3 + x^2 + 1$ (1-Wire bus)	0x8D / 0xB1 / 0xC6
CRC-8-Dallas/Maxim	$x^8 + x^5 + x^4 + 1$ (1-Wire bus)	0x31 / 0x8C / 0x98
CRC-8	$x^8 + x^7 + x^6 + x^4 + x^2 + 1$	0xD5 / 0xAB / 0xEA[1]
CRC-8-SAE J1850	$x^8 + x^4 + x^3 + x^2 + 1$	0x1D / 0xB8 / 0x8E

CRC-10	$x^{10} + x^9 + x^5 + x^4 + x + 1$	0x233 / 0x331 / 0x319
CRC-11	$x^{11} + x^9 + x^8 + x^7 + x^2 + 1$ (FlexRay[12])	0x385 / 0x50E / 0x5C2
CRC-12	$x^{12} + x^{11} + x^3 + x^2 + x + 1$ (telecom systems[13][14])	0x80F / 0xF01 / 0xC07
CRC-15-CAN	$x^{15} + x^{14} + x^{10} + x^8 + x^7 + x^4 + x^3 + 1$	0x4599 / 0x4CD1 / 0x62CC
CRC-16-IBM	$x^{16} + x^{15} + x^2 + 1$ (Bisync, Modbus, USB, ANSI X3.28[15], many others; also known as *CRC-16* and *CRC-16-ANSI*)	0x8005 / 0xA001 / 0xC002
CRC-16-CCITT	$x^{16} + x^{12} + x^5 + 1$ (X.25, HDLC, XMODEM, Bluetooth, SD, many others; known as *CRC-CCITT*)	0x1021 / 0x8408 / 0x8810[1]
CRC-16-T10-DIF	$x^{16} + x^{15} + x^{11} + x^9 + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$ (SCSI DIF)	0x8BB7[16] / 0xEDD1 / 0xC5DB
CRC-16-DNP	$x^{16} + x^{13} + x^{12} + x^{11} + x^{10} + x^8 + x^6 + x^5 + x^2 + 1$ (DNP, IEC 870, M-Bus)	0x3D65 / 0xA6BC / 0x9EB2
CRC-16-Fletcher	Not a CRC; see Fletcher's checksum	Used in Adler-32 A & B CRCs
CRC-24	$x^{24} + x^{22} + x^{20} + x^{19} + x^{18} + x^{16} + x^{14} + x^{13} + x^{11} + x^{10} + x^8 + x^7 + x^6 + x^3 + x + 1$ (FlexRay[12])	0x5D6DCB / 0xD3B6BA / 0xAEB6E5
CRC-24-Radix-64	$x^{24} + x^{23} + x^{18} + x^{17} + x^{14} + x^{11} + x^{10} + x^7 + x^6 + x^5 + x^4 + x^3 + x + 1$ (OpenPGP)	0x864CFB / 0xDF3261 / 0xC3267D
CRC-30	$x^{30} + x^{29} + x^{21} + x^{20} + x^{15} + x^{13} + x^{12} + x^{11} + x^8 + x^7 + x^6 + x^2 + x + 1$ (CDMA)	0x2030B9C7 / 0x38E74301 / 0x30185CE3
CRC-32-Adler	Not a CRC; see Adler-32	See Adler-32

CRC-32-IEEE 802.3	$x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$ (V.42, MPEG-2, PNG[17], POSIX cksum)	0x04C11DB7 / 0xEDB88320 / 0x82608EDB[4]
CRC-32C (Castagnoli)	$x^{32} + x^{28} + x^{27} + x^{26} + x^{25} + x^{23} + x^{22} + x^{20} + x^{19} + x^{18} + x^{14} + x^{13} + x^{11} + x^{10} + x^9 + x^8 + x^6 + 1$ (iSCSI, G.hn payload)	0x1EDC6F41 / 0x82F63B78 / 0x8F6E37A0[4]
CRC-32K (Koopman)	$x^{32} + x^{30} + x^{29} + x^{28} + x^{26} + x^{20} + x^{19} + x^{17} + x^{16} + x^{15} + x^{11} + x^{10} + x^7 + x^6 + x^4 + x^2 + x + 1$	0x741B8CD7 / 0xEB31D82E / 0xBA0DC66B[4]
CRC-32Q	$x^{32} + x^{31} + x^{24} + x^{22} + x^{16} + x^{14} + x^8 + x^7 + x^5 + x^3 + x + 1$ (aviation; AIXM[18])	0x814141AB / 0xD5828281 / 0xC0A0A0D5
CRC-64-ISO	$x^{64} + x^4 + x^3 + x + 1$ (HDLC — ISO 3309)	0x000000000000001B / 0xD800000000000000 / 0x8000000000000000D
CRC-64-ECMA-182	$x^{64} + x^{62} + x^{57} + x^{55} + x^{54} + x^{53} + x^{52} + x^{47} + x^{46} + x^{45} + x^{40} + x^{39} + x^{38} + x^{37} + x^{35} + x^{33} + x^{32} + x^{31} + x^{29} + x^{27} + x^{24} + x^{23} + x^{22} + x^{21} + x^{19} + x^{17} + x^{13} + x^{12} + x^{10} + x^9 + x^7 + x^4 + x + 1$[19]	0x42F0E1EBA9EA3693 / 0xC96C5795D7870F42 / 0xA17870F5D4F51B49

Таблица 3.

Циклический избыточный код CRC-4

Из всей совокупности методов контроля с использованием циклической структуры группового сигнала наибольшее распространение получил метод контроля первичных цифровых трактов CRC-4.

При передаче потока E1 по сети связи, включающей в себя ряд систем передачи плезиохронной и синхронной цифровых иерархий, прозрачных для его прохождения, возникает необходимость проверки качественных показателей первичного цифрового канала на всём его протяжении.

Из 265 бит, образующих стандартный цикл 2-мегабитного сигнала, сигналы с 1-ого по 31-й КИ (канальный интервал) являются случайными. Поэтому из всего группового сигнала можно идентифицировать только 7 бит циклового синхросигнала, что позволяет обнаруживать битовые ошибки без остановки связи, однако указанные 7 бит составляют только 1,4 % от общего объёма передаваемой информации. Проблема обеспечения контроля ошибок в остальных 31 каналах оптимально решается путём использования метода контроля с использованием избыточности циклического сигнала CRC-4.

CRC-4 был определён в 1980-х годах рекомендацией МСЭ-Т, но широкое распространение получил только в последнее время вследствие трудностей схемотехнической реализации, которую можно осуществить только методами микросхемотехники.

Сущность метода состоит в том, что цифровой сигнал разбивается на группы, получившие название блоков или субсверхциклов. На передающем конце тракта производится подсчёт суммы символов блока, эта информация в составе группового сигнала передаётся на приёмный конец тракта, где подсчёт суммы символов повторяется, и результаты подсчёта на передающем и приёмном концах сравниваются. Совпадение результатов интерпретируется как отсутствие ошибок при передаче блока. Расхождение результатов говорит о наличии ошибок при передаче данного блока. Для передачи результатов сравнения в обратном направлении, на передающий конец тракта, используются свободные биты служебных канальных интервалов группового сигнала обратного направления.

16 следующих друг за другом циклов образуют сверхцикл, который, в свою очередь, делится на два субсверхцикла (1-й и 2-й) по 8 циклов каждый. Таким образом, временной интервал CRC-4 равен 16 × 125 мкс = 2 мс.

Для формирования сигнала CRC-4 сумма бинарных символов каждого субсверхцикла делится на полином четвертой степени ($x^4 + x + 1$).

Результат деления, представляющий собой 4 бинарных символа, вводится в групповой сигнал в позициях от С1 до С4. Приёмная сторона использует аналогичный метод для того, чтобы затем сравнить кодовое слово, поступающее от передатчика, с результатом, полученным на приёмном конце. Если указанные слова различны, значит субсверхцикл, равный 2048 битам, был передан с ошибками.

Преимуществом этого метода является то, что с его помощью можно контролировать цифровой поток без остановки связи и независимо от его содержания.

Вместе с тем, при использовании указанного метода невозможно точно указать, какие биты из тысяч переданных были приняты с ошибками.

Сверхцикловый сигнал CRC-4 служит для того, чтобы можно было обеспечить синхронизацию по битам от С1 до С4. Биты Е (Е1 и Е2 для 1-го и 2-го субсверхциклов) инвертируются для сохранения структуры сверхцикла в моменты обнаружения ошибок. Таким образом, приёмная сторона может информировать передающую сторону об обнаружении ошибок передачи. После того, как установится цикловая синхронизация, сигналы CRC-4 будут передаваться непрерывно. Потеря синхронизации CRC-4 происходит тогда, когда более чем 914 сигналов CRC-4, передаваемые в течение 1 секунды, не будут соответствовать нормированным.

Аналогично организуется и проверка цифровых трактов других ступеней иерархии. Меняется только величина блоков и степень полинома: 6-я степень для CRC-6, используемого для контроля ИКМ-120, 8-я — для CRC-8, используемого для контроля ИКМ-480.

Формализованный алгоритм расчёта CRC16

Для получения контрольной суммы, необходимо сгенерировать полином. Основное требование к полиному: его степень должна быть равна длине контрольной суммы в битах. При этом старший бит полинома обязательно должен быть равен «1».Из файла берется первое слово. Если старший бит в слове "1", то слово сдвигается влево на один разряд с последующим выполнением операции XOR. Соответственно если старший бит в слове "0", то после сдвига операция XOR не выполняется. После сдвига (умножения) теряется старый старший бит, а младший бит освобождается (обнуляется). На место младшего бита загружается очередной бит из файла. Операция повторяется до тех пор, пока не загрузится последний бит файла.

3. РОЛЬ ХЭШ-ФУНКЦИИ В ОБЕСПЕЧЕНИЯ ЦЕЛОСТНОСТИ ИНФОРМАЦИИ

3.1 Хэш-функции

Хэш-функция предназначена для сжатия подписываемого документа M до нескольких десятков или сотен бит. Хэш-функция h( ) принимает в качестве аргумента сообщение (документ) M произвольной длины и возвращает хэш-значение h(M)-H фиксированной длины. Обычно хэшированная информация является сжатым двоичным представлением основного сообщения произвольной длины. Следует отметить, что значение хэш-функции h(M) сложным образом зависит от документа M и не позволяет восстановить сам документ M.

Хэш-функция должна удовлетворять целому ряду условий:

- хэш-функция должна быть чувствительна к всевозможным изменениям в тексте M, таким как вставки, выбросы, перестановки и т.п.;

- хэш-функция должна обладать свойством необратимости, то есть задача подбора документа M', который обладал бы требуемым значением хэш-функции, должна быть вычислительно неразрешима;

- вероятность того, что значения хэш-функции двух различных документов (вне зависимости от их длин) совпадут, должна быть ничтожно мала [123].

Большинство хэш-функций строится на основе однонаправленной функции f(-), которая образует выходное значение длиной n при задании двух входных значений длиной n. Этими входами являются блок исходного текста Mi и хэш-значение Hм предыдущего блока текста (рис. 6.1):

$Hi = f(Mi, H_{i-1})$.

Хэш-значение, вычисляемое при вводе последнего блока текста, становится хэш-значением всего сообщения M.

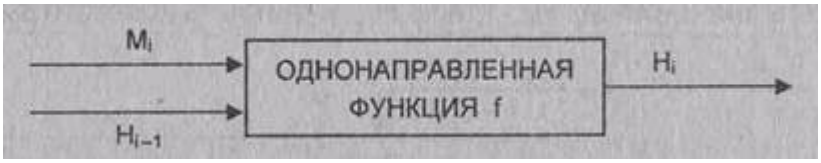

Рис. 2.-Построение однонаправленной хэш-функции

В результате однонаправленная хэш-функция всегда формирует выход фиксированной длины n (независимо от длины входного текста).

MD 5

Криптографическая хеш-функция	
Название	MD5
Создан	1991
Опубликован	Апрель 1992
Размер хеша	128 бит
Число раундов	4
Тип	хеш-функция

Таблица 4.

MD5 (англ. *Message Digest 5*) — 128-битный алгоритм хеширования, разработанный профессором Рональдом Л. Ривестом из Массачусетского технологического института (Massachusetts Institute of Technology, MIT) в 1991 году. Предназначен для создания «отпечатков» или «дайджестов» сообщений произвольной длины. Является улучшенной в плане безопасности версией MD4. Зная MD5-образ (называемый также MD5-хеш или MD5-дайджест), невозможно восстановить входное сообщение, так как одному MD5-образу могут соответствовать разные сообщения. Используется для

проверки подлинности опубликованных сообщений путём сравнения дайджеста сообщения с опубликованным. Эту операцию называют «проверка хеша» (hashcheck).

MD5 — один из серии алгоритмов по построению дайджеста сообщения, разработанный профессором Рональдом Л. Ривестом из Массачусетского технологического института. Разработан в 1991 году, как более надёжный вариант предыдущего алгоритма MD4. Позже Гансом Доббертином были найдены недостатки алгоритма MD4.

В 1993 году Берт ден Боер (Bert den Boer) и Антон Босселарис (Antoon Bosselaers) показали, что в алгоритме возможны псевдоколлизии, когда разным инициализирующим векторам соответствуют одинаковые дайджесты для входного сообщения.

В 1996 году Ганс Доббертин (Hans Dobbertin) объявил о коллизии в алгоритме и уже в то время было предложено использовать другие алгоритмы хеширования, такие как Whirlpool, SHA-1 или RIPEMD-160.

Из-за небольшого размера хеша в 128 бит, можно рассматривать birthday атаки. В марте 2004 года был запущен проект MD5CRK с целью обнаружения уязвимостей алгоритма, используя birthday атаки.

Проект MD5CRK закончился после 17 августа 2004, когда Ван Сяоюнь (Wang Xiaoyun), Фен Дэнгуо (Feng Dengguo), Лай Сюэцзя (Lai Xuejia) и Юй Хунбо (Yu Hongbo) обнаружили уязвимости в алгоритме.

1 марта 2005, Arjen Lenstra, Xiaoyun Wang, и Benne de Weger продемонстрировали построение двух X.509 документов с различными открытыми ключами и одинаковым хешем MD5.

Алгоритм MD5

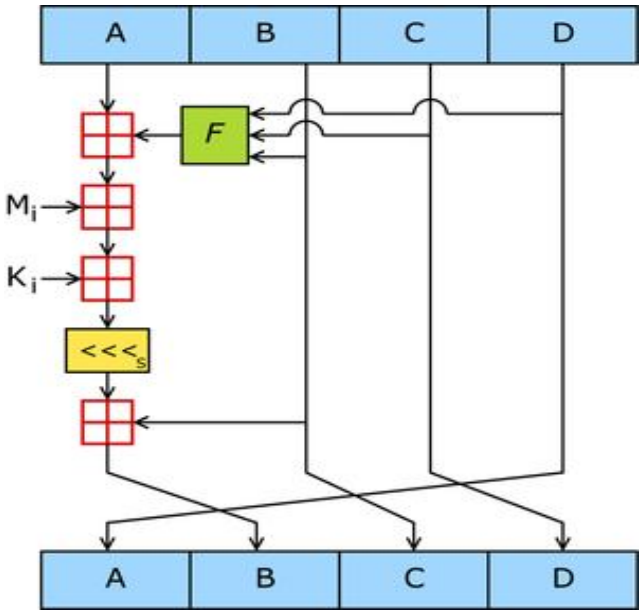

Рис. 3.- Схема работы алгоритма MD5

На вход алгоритма поступает входной поток данных, хеш которого необходимо найти. Длина сообщения может быть любой (в том числе нулевой). Запишем длину сообщения в L. Это число целое и неотрицательное. Кратность каким-либо числам необязательна. После поступления данных идёт процесс подготовки потока к вычислениям.

Ниже приведены 5 шагов алгоритма:

Шаг 1. Выравнивание потока

Входные данные выравниваются так, чтобы их размер был сравним с 448 по модулю 512 (L’ = 512 × N + 448). Сначала дописывают единичный бит в конец потока, затем необходимое число нулевых бит (выравнивание происходит, даже если длина уже конгруэнтна — сравнима с 448).

Шаг 2. Добавление длины сообщения

В оставшиеся 64 бита дописывают 64-битное представление длины данных (количество бит в сообщении) до выравнивания. Если длина превосходит $2^{64} - 1$, то дописывают только младшие биты. После этого длина

потока станет кратной 512. Вычисления будут основываться на представлении этого потока данных в виде массива слов по 512 бит.

Шаг 3. Инициализация буфера

Для вычислений инициализируются 4 переменных размером по 32 бита и задаются начальные значения шестнадцатеричными числами:

A = 01 23 45 67;

B = 89 AB CD EF;

C = FE DC BA 98;

D = 76 54 32 10.

В этих переменных будут храниться результаты промежуточных вычислений. Начальное состояние ABCD называется инициализирующим вектором.

Определим ещё функции и константы, которые нам понадобятся для вычислений.

- Потребуются 4 функции для четырёх раундов. Введём функции от трёх параметров — слов, результатом также будет слово.

  1 раунд $FunF(X,Y,Z) = (X \wedge Y) \vee (\neg X \wedge Z)$.

  2 раунд $FunG(X,Y,Z) = (X \wedge Z) \vee (\neg Z \wedge Y)$.

  3 раунд $FunH(X,Y,Z) = X \oplus Y \oplus Z$.

  4 раунд $FunI(X,Y,Z) = Y \oplus (\neg Z \vee X)$.

- Определим таблицу констант T[1..64] — 64-элементная таблица данных, построенная следующим образом: $T[i] = int(4294967296 * |sin(i)|)$, где $4294967296 = 2^{32}$.[3]
- Выровненные данные разбиваются на блоки (слова) по 32 бита, и каждый блок проходит 4 раунда из 16 операторов. Все операторы однотипны и имеют вид: [abcd k s i], определяемый как $a = b + ((a + Fun(b,c,d) + X[k] + T[i]) <<< s)$, где X — блок данных. X[k] = M [n * 16 + k], где k — номер 32-битного слова из n-го 512-битного блока сообщения, и s — циклический сдвиг влево на s бит полученого 32-битного аргумента.

Шаг 4. Вычисление в цикле

Заносим в блок данных элемент n из массива. Сохраняются значения A, B, C и D, оставшиеся после операций над предыдущими блоками (или их начальные значения, если блок первый).

AA = A

BB = B

CC = C

DD = D

Раунд 1

/*[abcd k s i] a = b + ((a + F(b,c,d) + X[k] + T[i]) <<< s). */

[ABCD 0 7 1][DABC 1 12 2][CDAB 2 17 3][BCDA 3 22 4]

[ABCD 4 7 5][DABC 5 12 6][CDAB 6 17 7][BCDA 7 22 8]

[ABCD 8 7 9][DABC 9 12 10][CDAB 10 17 11][BCDA 11 22 12]

[ABCD 12 7 13][DABC 13 12 14][CDAB 14 17 15][BCDA 15 22 16]

Раунд 2

/*[abcd k s i] a = b + ((a + G(b,c,d) + X[k] + T[i]) <<< s). */

[ABCD 1 5 17][DABC 6 9 18][CDAB 11 14 19][BCDA 0 20 20]

[ABCD 5 5 21][DABC 10 9 22][CDAB 15 14 23][BCDA 4 20 24]

[ABCD 9 5 25][DABC 14 9 26][CDAB 3 14 27][BCDA 8 20 28]

[ABCD 13 5 29][DABC 2 9 30][CDAB 7 14 31][BCDA 12 20 32]

Раунд 3

/*[abcd k s i] a = b + ((a + H(b,c,d) + X[k] + T[i]) <<< s). */

[ABCD 5 4 33][DABC 8 11 34][CDAB 11 16 35][BCDA 14 23 36]

[ABCD 1 4 37][DABC 4 11 38][CDAB 7 16 39][BCDA 10 23 40]

[ABCD 13 4 41][DABC 0 11 42][CDAB 3 16 43][BCDA 6 23 44]

[ABCD 9 4 45][DABC 12 11 46][CDAB 15 16 47][BCDA 2 23 48]

Раунд 4

/*[abcd k s i] a = b + ((a + I(b,c,d) + X[k] + T[i]) <<< s). */

[ABCD 0 6 49][DABC 7 10 50][CDAB 14 15 51][BCDA 5 21 52]

[ABCD 12 6 53][DABC 3 10 54][CDAB 10 15 55][BCDA 1 21 56]

[ABCD 8 6 57][DABC 15 10 58][CDAB 6 15 59][BCDA 13 21 60]

[ABCD 4 6 61][DABC 11 10 62][CDAB 2 15 63][BCDA 9 21 64]

Суммируем с результатом предыдущего цикла:

A = AA + A

B = BB + B

C = CC + C

D = DD + D

После окончания цикла необходимо проверить, есть ли ещё блоки для вычислений. Если да, то изменяем номер элемента массива (n++) и переходим в начало цикла.

Шаг 5. Результат вычислений

Результат вычислений находится в буфере ABCD, это и есть хеш. Если вывести слова в обратном порядке DCBA, то мы получим наш MD5 хеш.

MD5-хеши

Хеш содержит 128 бит (16 байт) и обычно представляется как последовательность из 32 шестнадцатеричных цифр.

Несколько примеров хеша:

MD5("md5") = 1bc29b36f623ba82aaf6724fd3b16718

Даже небольшое изменение входного сообщения (в нашем случае на один бит: ASCII символ «5» с кодом $0x35_{16}$ = 00011010**1**$_2$ заменяется на символ «4» с кодом $0x34_{16}$ = 00011010**0**$_2$) приводит к полному изменению хеша. Такое свойство алгоритма называется лавинным эффектом.

MD5("md4") = c93d3bf7a7c4afe94b64e30c2ce39f4f

Пример MD5-хеша для «нулевой» строки:

MD5("") = d41d8cd98f00b204e9800998ecf8427e

Криптоанализ

На данный момент существуют несколько видов «взлома» хешей MD5 — подбора сообщения с заданным хешем:

- Перебор по словарю
- Brute-force
- RainbowCrack

Атаки переборного типа

Для полного перебора или перебора по словарю можно использовать программы PasswordsPro[4], MD5BFCPF[5], John the Ripper. Для перебора по словарю существуют готовые словари.[6]

RainbowCrack — еще один метод взлома хеша. Он основан на генерировании большого количества хешей из набора символов, чтобы по получившейся базе вести поиск заданного хеша. Хотя генерация хешей занимает много времени, зато последующий взлом производится очень быстро. Rainbow-таблицы можно найти как готовые[7] [8], так и сгенерировать самостоятельно.

Важно отметить, что наличие у MD5 коллизий упрощает (но не усложняет) взлом многих приложений MD5, когда по заданной хеш-сумме достаточно найти *любое* значение входных данных ей соответствующее.

Коллизии MD5

Коллизия хеш-функции — это получение одинакового значения функции для разных сообщений и идентичного начального буфера. В отличие от коллизий, *псевдоколлизии* определяются как равные значения хеша для разных значений начального буфера, причём сами сообщения могут совпадать или отличаться. В 1996 году Ганс Доббертин нашёл псевдоколлизии в MD5, используя определённые *инициализирующие* векторы, отличные от стандартных. Оказалось, что можно для известного сообщения построить второе, такое, что оно будет иметь такой же хеш, как и исходное. С точки зрения математики это означает: MD5(IV,L1) = MD5(IV,L2), где IV — начальное значение буфера, а L1 и L2 — различные сообщения. Например, если взять начальное значение буфера:

A = 0x12AC2375

B = 0x3B341042

C = 0x5F62B97C

D = 0x4BA763ED

и задать входное сообщение

AA1DDABE	D97ABFF5	BBF0E1C1	32774244
1006363E	7218209D	E01C136D	9DA64D0E
98A1FB19	1FAE44B0	236BB992	6B7A779B
1326ED65	D93E0972	D458C868	6B72746A

то, добавляя число 2^9 к определённому 32-разрядному слову в блочном буфере, можно получить второе сообщение с таким же хешем. Ханс Доббертин представил такую формулу:

$$L2_i = \begin{cases} L1_i, & i \neq 14; \\ L1_i + 2^9, & i = 14. \end{cases}$$

Тогда MD5(IV, L1) = MD5(IV, L2) = BF90E670752AF92B9CE4E3E1B12CF8DE.

В 2004 году китайские исследователи Ван Сяоюнь (Wang Xiaoyun), Фен Дэнгуо (Feng Dengguo), Лай Сюэцзя (Lai Xuejia) и Юй Хунбо (Yu Hongbo) объявили об обнаруженной ими уязвимости в алгоритме, позволяющей за небольшое время (1 час на кластере en:IBM_p690) находить коллизии.[9][10]

В 2005 году Ван Сяоюнь и Юй Хунбо из университета Шаньдун в Китае опубликовали алгоритм, который может найти две различные последовательности в 128 байт, которые дают одинаковый MD5-хеш. Одна из таких пар (отличающиеся разряды выделены):

d131dd02c5e6eec4693d9a0698aff95c	2fcab5**8**712467eab4004583eb8fb7f89
55ad340609f4b30283e488832571415a	085125e8f7cdc99fd91dbd**f**280373c5b
d8823e3156348f5bae6dacd436c919c6	dd53e2**b**487da03fd02396306d248cda0

e99f33420f577ee8ce54b67080**a**80d1e	c69821bcb6a8839396f965**2**b6ff72a70

и

d131dd02c5e6eec4693d9a0698aff95c	2fcab5**0**712467eab4004583eb8fb7f89
55ad340609f4b30283e4888325**f**1415a	085125e8f7cdc99fd91dbd**7**280373c5b
d8823e3156348f5bae6dacd436c919c6	dd53e2**3**487da03fd02396306d248cda0
e99f33420f577ee8ce54b67080**2**80d1e	c69821bcb6a8839396f965**a**b6ff72a70

Каждый из этих блоков даёт MD5-хеш, равный 79054025255fb1a26e4bc422aef54eb4.

Метод Вана Сяоюня и Юя Хунбо

Метод Вана Сяоюня и Юя Хунбо использует тот факт, что MD5 построен на итерационном методе Меркле-Дамгарда. Поданный на вход файл, сначала дополняется, так чтобы его длина была кратна 64 байтам, после этого он делится на блоки по 64 байта каждый $M_0,M_1,\ldots,M_{n-1}$. Далее вычисляется последовательность 16-ти байтных состояний $s_0,\ldots,s_n$ по правилу $s_{i+1}=f(s_i,M_i)$, где f некоторая фиксированная функция. Начальное состояние s_0 называется инициализирующим вектором.

Метод позволяет для заданного инициализирующего вектора найти две пары M,M' и N,N', такие что $f(f(s,M),M') = f(f(s,N),N')$. Важно отметить, что этот метод работает для любого инициализирующего вектора, а не только для вектора используемого по стандарту.

Эта атака является разновидностью *дифференциальной* атаки, которая, в отличие от других атак этого типа, использует целочисленное вычитание а не XOR в качестве меры разности. При поиске коллизий используется метод модификации сообщений: сначала выбирается произвольное сообщение M_0, далее оно модифицируется по некоторым правилам, сформулированным в статье, после чего вычисляется дифференциал хеш-функции, причём $M'_0 = M_0 + dM_0$ с вероятностью 2^{-37}. К M_0 и M'_0 применяется функция сжатия для проверки условий коллизии; далее выбирается произвольное M_1, модифицируется, вычисляется новый дифференциал, равный нулю с

вероятностью 2^{-30}, а равенство нулю дифференциала хеш-функции как раз означает наличие коллизии. Оказалось, что найдя одну пару M_0 и M'_0, можно менять лишь два последних слова в M_0, тогда для нахождения новой пары M_1 и M'_1 требуется всего около 2^{39} операций хеширования.

Примеры использования

MD5 позволяет получать относительно надёжный идентификатор для блока данных. Такое свойство алгоритма широко применяется в разных областях. Оно позволяет искать дублирующиеся файлы на компьютере, сравнивая MD5 файлов, а не их содержимое. Как пример, dupliFinder — графическая программа под Windows и Linux. Такой же поиск может работать и в интернете.

С помощью MD5 проверяют целостность скачанных файлов — так, некоторые программы идут вместе со значением хеша. Например, диски для инсталляции.

MD5 используется для хеширования паролей. В системе UNIX каждый пользователь имеет свой пароль и его знает только пользователь. Для защиты паролей используется хеширование. Получить настоящий пароль можно только полным перебором. При появлении UNIX единственным способом хеширования был DES (Data Encryption Standard), но им могли пользоваться только жители США, потому что исходные коды DES нельзя было вывозить из страны. Во FreeBSD решили эту проблему. Пользователи США могли использовать библиотеку DES, а остальные пользователи имеют метод, разрешённый для экспорта. Поэтому в FreeBSD стали использовать MD5 по умолчанию.[13]

Многие системы используют базу данных для хранения паролей и существует несколько способов для хранения паролей.

- Пароли хранятся как есть. При взломе такой базы все пароли станут известны.
- Хранятся только хеши паролей (с помощью MD5, SHA). Найти пароли можно только полным перебором. Но сейчас такая задача решается за доли секунды. Пароль из таблицы был найден всего за 0,036059 сек.[14]
- Хранятся хеши паролей и несколько случайных символов. К каждому паролю добавляется несколько случайных символов (их ещё называют «salt» или «соль») и результат ещё раз хешируется. Например, md5(md5(pass)+word). Найти пароль с помощью таблиц таким методом не получится.

Существует несколько надстроек над MD5.

- MD5 (HMAC) — HMAC — Keyed-Hashing for Message Authentication (хеширование с ключом для аутентификации сообщения) — алгоритм позволяет хешировать входное сообщение L с некоторым ключом K, такое хеширование позволяет аутентифицировать подпись.
- MD5 (Base64) — здесь полученный MD5 хеш кодируется алгоритмом Base64.
- MD5 (Unix) — алгоритм вызывает тысячу раз стандартный MD5.

SHA-1

Криптографическая хеш-функция	
Название	SHA-1
Создан	1995
Опубликован	1995
Размер хеша	160 бит
Число раундов	80
Тип	хеш-функция

Таблица 5.

Secure Hash Algorithm 1 — алгоритм криптографического хеширования. Описан в RFC 3174. Для входного сообщения произвольной длины (максимум $2^{64} - 1$ бит) алгоритм генерирует 160-битное хеш-значение, называемое также дайджестом сообщения. Используется во многих криптографических приложениях и протоколах. Также рекомендован в качестве основного для государственных учреждений в США. Принципы, положенные в основу SHA-1, аналогичны тем, которые использовались Рональдом Ривестом при проектировании MD4.

История

В 1993 году NSA совместно с NIST разработали алгоритм безопасного хеширования (сейчас известный как SHA-0) (опубликован в документе *FIPS* PUB 180) для стандарта безопасного хеширования. Однако вскоре NSA отозвало данную версию, сославшись на обнаруженную ими ошибку, которая так и не была раскрыта. И заменило его исправленной версией, опубликованной в 1995 году в документе *FIPS* PUB 180-1. Эта версия и считается тем, что называют SHA-1. Немного спустя, на конференции CRYPTO в 1998 году два французских исследователя представили атаку на алгоритм SHA-0, которая не работала на алгоритме SHA-1 Возможно, это и была ошибка, открытая NSA.

Описание алгоритма

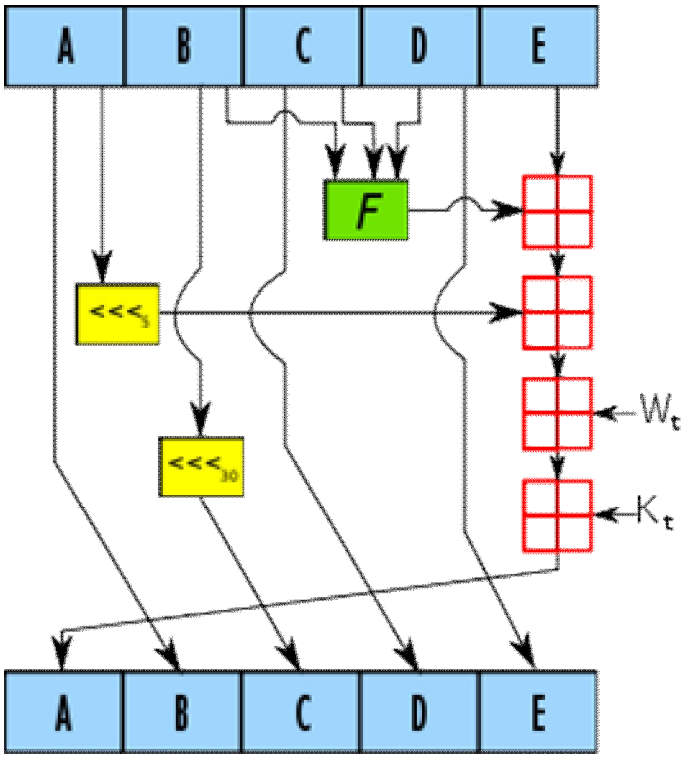

Рис.4. - Одна итерация алгоритма SHA1

SHA-1 реализует хеш-функцию, построенную на идее функции сжатия. Входами функции сжатия являются блок сообщения длиной 512 бит и выход предыдущего блока сообщения. Выход представляет собой значение всех

хеш-блоков до этого момента. Иными словами хеш блока M_i равен $h_i = f(M_i, h_{i-1})$. Хеш-значением всего сообщения является выход последнего блока.

Инициализация

Исходное сообщение разбивается на блоки по 512 бит в каждом. Последний блок дополняется до длины, кратной 512 бит. Сначала добавляется 1 а потом нули, чтобы длина блока стала равной (512 - 64 = 448) бит. В оставшиеся 64 бита записывается длина исходного сообщения в битах. Если последний блок имеет длину более 448, но менее 512 бит, дополнение выполняется следующим образом: сначала добавляется 1, затем нули вплоть до конца 512-битного блока; после этого создается ещё один 512-битный блок, который заполняется вплоть до 448 бит нулями, после чего в оставшиеся 64 бита записывается длина исходного сообщения в битах. Дополнение последнего блока осуществляется всегда, даже если сообщение уже имеет нужную длину.

Инициализируются пять 32-битовых переменных.

A = a = 0x67452301

B = b = 0xEFCDAB89

C = c = 0x98BADCFE

D = d = 0x10325476

E = e = 0xC3D2E1F0

Определяются четыре нелинейные операции и четыре константы.

$F_t(m,l,k) = (m \wedge l) \vee (\neg m \wedge k)$	K_t = 0x5A827999	0≤t≤19
$F_t(m,l,k) = m \oplus l \oplus k$	K_t = 0x6ED9EBA1	20≤t≤39

$F_t(m,l,k) = (m \wedge l) \vee (m \wedge k) \vee (l \wedge k)$	K_t = 0x8F1BBCDC	40≤t≤59
$F_t(m,l,k) = m \oplus l \oplus k$	K_t = 0xCA62C1D6	60≤t≤79

Главный цикл

Главный цикл итеративно обрабатывает каждый 512-битный блок. Итерация состоит из четырех этапов по двадцать операций в каждом. Блок сообщения преобразуется из 16 32-битовых слов M_i в 80 32-битовых слов W_j по следующему правилу:

$W_t = M_t$ при 0≤t≤15
$W_t = (W_{t-3} \oplus W_{t-8} \oplus W_{t-14} \oplus W_{t-16}) << 1$ при 16≤t≤79

здесь << — это циклический сдвиг влево

для *t* **от** 0 **до** 79
temp = (a<<5) + F_t(b,c,d) + e + W_t + K_t
e = d
d = c
c = b<<30
b = a
a = temp

После этого a, b, c, d, e прибавляются к A, B, C , D , E соответственно. Начинается следующая итерация.

Итоговым значением будет объединение пяти 32-битовых слов в одно 160-битное хеш-значение.

Псевдокод SHA-1

Псевдокод алгоритма SHA-1 следующий:

Замечание: Все используемые переменные 32 бита.

Инициализация переменных:

h0 = 0x67452301

h1 = 0xEFCDAB89

h2 = 0x98BADCFE

h3 = 0x10325476

h4 = 0xC3D2E1F0

Предварительная обработка:

Присоединяем бит '1' к сообщению

Присоединяем k битов '0', где k наименьшее число ≥ 0 такое, что длина получившегося сообщения (в битах) сравнима по модулю 512 с 448 (length mod 512 == 448).

Добавляем длину исходного сообщения (до предварительной обработки) как целое 64-битное Big-endian число, в *битах*.

В процессе сообщение разбивается последовательно по 512 бит:

for перебираем все такие части

разбиваем этот кусок на 16 частей, слов по 32-бита w[i], 0 <= i <= 15

16 слов по 32-бита дополняются до 80 32-битовых слов:

for i **from** 16 to 79

w[i] = (w[i-3] **xor** w[i-8] **xor** w[i-14] **xor** w[i-16]) **циклический сдвиг** 1

Инициализация хеш-значений этой части:

a = h0

b = h1

c = h2

d = h3

e = h4

Основной цикл:

for i **from** 0 to 79

if 0 ≤ i ≤ 19 **then**

f = (b **and** c) **or** ((**not** b) **and** d)

k = 0x5A827999

else if 20 ≤ i ≤ 39

f = b **xor** c **xor** d

k = 0x6ED9EBA1

else if 40 ≤ i ≤ 59

f = (b **and** c) **or** (b **and** d) **or** (c **and** d)

k = 0x8F1BBCDC

else if 60 ≤ i ≤ 79

f = b **xor** c **xor** d

k = 0xCA62C1D6

temp = (a **leftrotate** 5) + f + e + k + w[i]

e = d

d = c

c = b **leftrotate** 30

b = a

a = temp

Добавляем хеш-значение этой части к результату:

h0 = h0 + a

h1 = h1 + b

h2 = h2 + c

h3 = h3 + d

h4 = h4 + e

Итоговое хеш-значение:

digest = hash = h0 **append** h1 **append** h2 **append** h3 **append** h4

Вместо оригинальной формулировки FIPS PUB 180-1 приведены следующие эквивалентные выражения и могут быть использованы на компьютере f в главном цикле:

(0 ≤ i ≤ 19): f = d **xor** (b **and** (c **xor** d)) *(альтернатива 1)*

(0 ≤ i ≤ 19): f = (b **and** c) **xor** ((**not** b) **and** d) *(альтернатива 2)*

(0 ≤ i ≤ 19): f = (b **and** c) + ((**not** b) **and** d) *(альтернатива 3)*

(40 ≤ i ≤ 59): f = (b **and** c) **or** (d **and** (b **or** c)) *(альтернатива 1)*

(40 ≤ i ≤ 59): f = (b **and** c) **or** (d **and** (b **xor** c)) *(альтернатива 2)*

(40 ≤ i ≤ 59): f = (b **and** c) + (d **and** (b **xor** c)) *(альтернатива 3)*

(40 ≤ i ≤ 59): f = (b **and** c) **xor** (b **and** d) **xor** (c **and** d) *(альтернатива 4)*

Примеры

Ниже приведены примеры хешей SHA-1. Для всех сообщений подразумевается использование кодировки ASCII.

Хеш панграммы на русском:

SHA-1("В чащах юга жил бы цитрус? Да, но фальшивый экземпляр!")

= 9e32295f 8225803b b6d5fdfc c0674616 a4413c1b

Хеш панграммы на английском:

SHA-1("The quick brown fox jumps over the lazy dog")
= 2fd4e1c6 7a2d28fc ed849ee1 bb76e739 1b93eb12
SHA-1("sha")
= d8f45903 20e1343a 915b6394 170650a8 f35d6926

Небольшое изменение исходного текста (одна буква в верхнем регистре) приводит к сильному изменению самого хеша. Это происходит вследствие лавинного эффекта.

SHA-1("Sha") = ba79baeb 9f10896a 46ae7471 5271b7f5 86e74640

Даже для пустой строки вычисляется нетривиальное хеш-значение.

SHA-1("") = da39a3ee 5e6b4b0d 3255bfef 95601890 afd80709

Криптоанализ

Криптоанализ хеш-функций направлен на исследование уязвимости к различного вида атакам. Основные из них:

- нахождение коллизий — ситуация, когда двум различным исходным сообщениям соответствует одно и то же хеш-значение.
- нахождение прообраза — исходного сообщения — по его хешу.

При решении методом «грубой силы»:

- вторая задача требует 2^{160} операций.
- первая же требует в среднем $2^{160/2} = 2^{80}$ операций, если использовать парадокс дней рождения.

От устойчивости хеш-функции к нахождению коллизий зависит безопасность электронной цифровой подписи с использованием данного хеш-алгоритма. От устойчивости к нахождению прообраза зависит безопасность хранения хешей паролей для целей аутентификации.

В январе 2005 года Vincent Rijmen и Elisabeth Oswald опубликовали сообщение об атаке на усеченную версию SHA-1 (53 раунда вместо 80), которая позволяет находить коллизии меньше, чем за 2^{80} операций.

В феврале 2005 года Сяоюнь Ван, Йицунь Лиза Йинь и Хунбо Ю представили атаку на полноценный SHA-1, которая требует менее 2^{69} операций.

О методе авторы пишут:

Мы представляем набор стратегий и соответствующих методик, которые могут быть использованы для устранения некоторых важных препятствий в поиске коллизий в SHA-1. Сначала мы ищем близкие к коллизии дифференциальные пути, которые имеют небольшой «вес Хэмминга» в «векторе помех», где каждый 1-бит представляет 6-шаговую локальную коллизию. Потом, мы соответствующим образом корректируем дифференциальный путь из первого этапа до другого приемлемого дифференциального пути, чтобы избежать неприемлемых последовательных и усеченных коллизий. В конце концов мы преобразуем два одноблоковых близких к коллизии дифференциальных пути в один двухблоковый коллизионный путь с удвоенной вычислительной сложностью.

Также они заявляют:

В частности, наш анализ основан на оригинальной дифференциальной атаке на SHA-0, «near-collision» атаке на SHA-0, мультиблоковой методике, а также методикам модификации исходного сообщения, использованных при атаках поиска коллизий на HAVAL-128, MD4, RIPEMD и MD5.

Статья с описанием алгоритма была опубликована в августе 2005 года на конференции CRYPTO.

В этой же статье авторы опубликовали атаку на усеченный SHA-1 (58 раундов), которая позволяет находить коллизии за 2^{33} операций.

В августе 2005 года на CRYPTO 2005 эти же специалисты представили улучшенную версию атаки на полноценный SHA-1, с вычислительной сложностью в 2^{63} операций. В декабре 2007 года детали этого улучшения были проверены Мартином Кохраном.

Кристоф де Каньер и Кристиан Рехберг позже представили усовершенствованную версию атаки на SHA-1, за что были удостоены награды за лучшую статью на конференции ASIACRYPT 2006. Ими была представлена двух-блоковая коллизия на 64-раундовый алгоритм с вычислительной сложностью около 2^{35} операций.

Хотя теоретически SHA-1 считается взломанным (количество вычислительных операций сокращено в 2^{80-63} = 131 000 раз), на практике подобный взлом неосуществим, так как займет пять миллиардов лет.

Бурт Калински, глава исследовательского отдела в «лаборатории RSA» предсказывает, что первая атака по нахождению прообраза будет успешно осуществлена в ближайшие 5-10 лет.

Ввиду того, что теоретические атаки на SHA-1 оказались успешными, NIST планирует полностью отказаться от использования SHA-1 в цифровых подписях.

2 ноября 2007 года NIST анонсировало конкурс по разработке нового алгоритма SHA-3, который продлится до 2012 года.

Сравнение с MD5

И MD5, и SHA-1 являются, по сути, улучшенными продолжениями MD4.

Сходства:

- Четыре этапа.
- Каждое действие прибавляется к ранее полученному результату.
- Размер блока обработки равный 512 бит.
- Оба алгоритма выполняют сложение по модулю 2^{32}, они рассчитаны на 32-битную архитектуру.

Различия:

- В SHA-1 на четвертом этапе используется та же функция f, что и на втором этапе.
- В MD5 в каждом действии используется уникальная прибавляемая константа. В SHA-1 константы используются повторно для каждой из четырех групп.
- В SHA-1 добавлена пятая переменная.
- SHA-1 использует циклический код исправления ошибок.
- В MD5 четыре сдвига, используемые на каждом этапе отличаются от значений, используемых на предыдущих этапах. В SHA на каждом этапе используется постоянное значение сдвига.
- В MD5 четыре различных элементарных логических функции, в SHA-1 — три.
- В MD5 длина дайджеста составляет 128 бит, в SHA-1 — 160 бит.
- SHA-1 содержит больше раундов (80 вместо 64) и выполняется на 160-битном буфере по сравнению со 128-битным буфером MD5. Таким образом, SHA-1 должен выполняться приблизительно на 25 % медленнее, чем MD5 на той же аппаратуре.

Брюс Шнайер делает следующий вывод : «SHA-1 — это MD4 с добавлением расширяющего преобразования, дополнительного этапа и улучшенным лавинным эффектом. MD5 — это MD4 с улучшенным битовым хешированием, дополнительным этапом и улучшенным лавинным эффектом.»

Использование

Хеш-функции используются в системах контроля версий, системах электронной подписи, а также для построения кодов аутентификации.

SHA-1 является наиболее распространенным из всего семейства SHA и применяется в различных широко распространенных криптографических приложениях и алгоритмах.

SHA-1 используется в следующих приложениях:

- S/MIME — дайджесты сообщений.
- SSL — дайджесты сообщений.
- IPSec — для алгоритма проверки целостности в соединении «точка-точка».
- SSH — для проверки целостности переданных данных.
- PGP — для создания электронной цифровой подписи.
- Git — для идентификации каждого объекта по SHA-1-хешу от хранимой в объекте информации.

Схемы использование хэш функции в ИКС

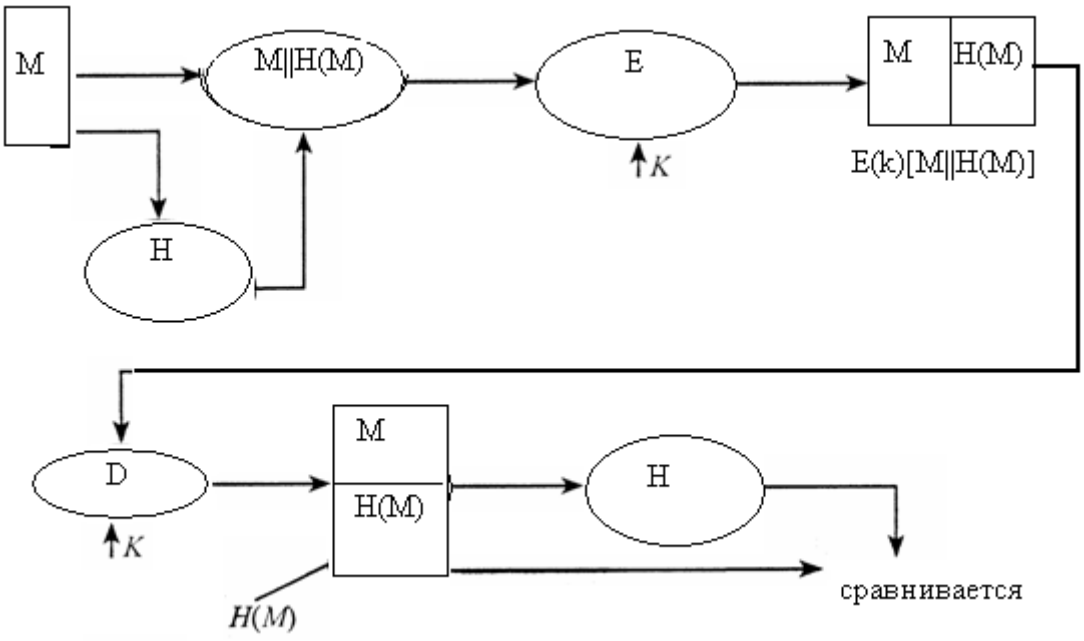

Рис. 5.

a) A→B : E(k)[M||H(M)] (**Рис. 5.**)

- обеспечивает секретности

- обеспечивает целостности (H(M) – криптографический защищен).

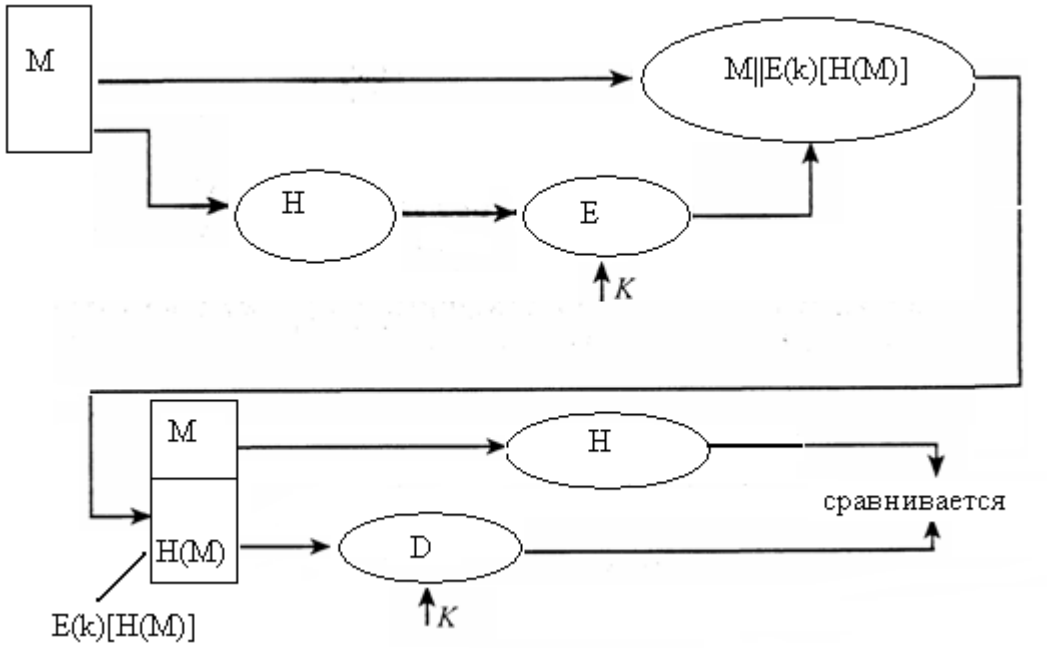

Рис. 6.

а) A→B:M||E(k)[H(M)] (**Рис. 6.**)

- обеспечивает целостности (H(M) – криптографический защищен).

3.2 HMAC

HMAC (сокращение от англ. *hash-based message authentication code*, хеш-код идентификации сообщений) — алгоритм усиления криптостойкости других криптоалгоритмов .

Алгоритм

Хеш-функция разделяет сообщения на блоки фиксированного размера и применяет к ним функцию сжатия. (MD5 или SHA-1 используют блоки 512 бит). После применения HMAC размер результата не меняется (128 или 160 бит для MD5 и SHA-1).

Функция HMAC определяется следующим образом:

$$\mathrm{HMAC}_K(m) = h\Big((K \oplus \mathrm{opad}) \| h\big((K \oplus \mathrm{ipad}) \| m\big)\Big)$$, где:

- h — хеш-функция
- K — секретный ключ, дополненный нулями до размера блока
- m — сообщение для идентификации
- $\|$ — конкатенация
- $\oplus$ — xor
- opad — 0x5c5c..5c (длина равна размеру блока)
- ipad — 0x3636..36 (длина равна размеру блока)

Если длина секретного ключа превышает размер блока, то ключ необходимо укоротить, преобразовав его с помощью функции h, и дополнить нулями до размера блока.

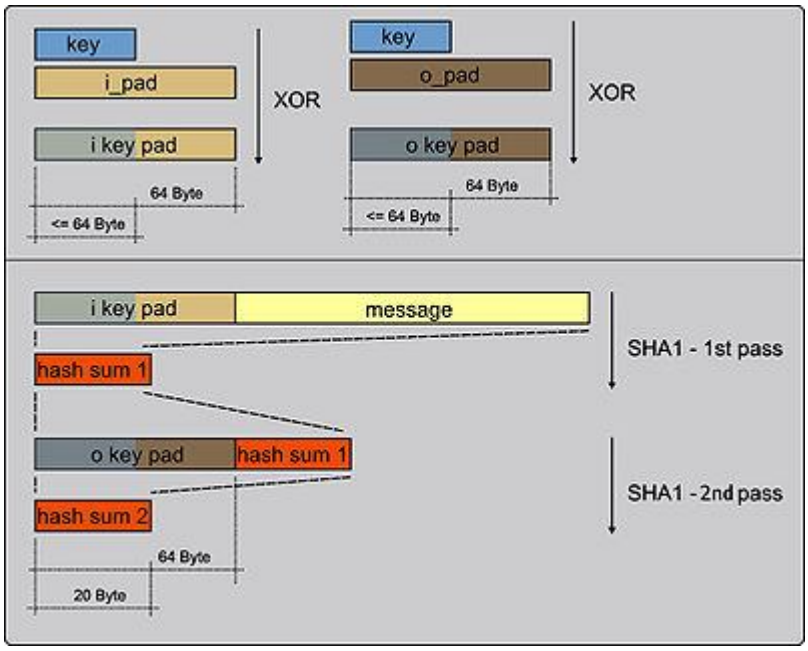

Рис. 7.

В криптографии HMAC (Хэш основе Message Authentication Code), является специфической конструкции для расчета кода аутентификации сообщений (MAC) с участием криптографической хэш-функция в сочетании с секретным ключом. Как и в любой MAC, она может быть использована одновременно проверить и целостность данных и подлинность сообщения. Любое итерационных криптографической хэш-функции, как например MD5 или SHA-1, могут быть использованы при расчете HMAC; в результате MAC алгоритм называется HMAC-MD5 и HMAC-SHA1 соответственно. Криптографической стойкости HMAC зависит от криптографической стойкости основных хэш-функция, размер хэш длиной выход в битах и от размера и качества криптографического ключа.

Итерационных хэш-функция разбивает сообщение на блоки фиксированного размера и перебирает их функции сжатия. Так, например, MD5 и SHA-1 работает на 512-битные блоки. Размер выходного КСОМ то же, что и основные функции хеширования (128 или 160 бит в случае MD5

или SHA-1, соответственно), хотя он может быть усечен по желанию.

Определение и анализ строительства HMAC был впервые опубликован в 1996 году Mihir Bellare, Ran Канетти, и Хьюго Кравчик, [1], который также написал RFC 2104. В этом документе также определены вариант называется NMAC, что редко, если когда-либо использовал. FIPS PUB 198 обобщает и стандартизирует использование HMACs. HMAC-SHA-1 и HMAC-MD5 используются в IPsec и TLS протоколы.

HMAC SHA 1

HMACSHA1 — это хэш-алгоритм с ключом, созданный на основе хэш-функции SHA1 и используемый для вычисления хэш-кода проверки подлинности сообщения (HMAC).Процесс вычисления кода HMAC заключается в смешивании секретного ключа с данными сообщения, вычислении хэш-функции результата, повторном смешивании хэш-значения с секретным ключом и повторном применении хэш-функции.Длина выходного хэша составляет 160 бит.

Код HMAC может использоваться для выявления факта подделки сообщения, передаваемого по незащищенному каналу, при условии, что секретный ключ известен отправителю и получателю.Отправитель вычисляет хэш-значение исходных данных, а затем передает исходные данные и хэш-значение в одном сообщении.Получатель повторно вычисляет значение хэша полученных данных и проверяет, совпадает ли оно с полученным кодом HMAC.

Любое изменение данных или значения хэша вызовет несовпадение, поскольку для изменения сообщения и повторного создания правильного кода HMAC необходимо знать секретный ключ.Таким образом, если исходное значение хэша совпадает с вычисленным, считается, что подлинность сообщения установлена.

SHA-1 (Secure Hash Algorithm; другое название — Secure Hash Standard, SHS) является криптографическим хэш-алгоритмом, опубликованным правительством США.Он используется для формирования 160-разрядных хэш-значений из строк произвольной длины.

4. БЕЗОПАСНОСТЬ ЖИЗНЕДЕЯТЕЛЬНОСТИ

4.1. Оздоровления воздушной среды в производственном помещении.

Одним из необходимых условий здорового и высокопроизводительного труда является обеспечение чистоты воздуха и нормальных метеорологических условий в рабочей зоне помещений, т. е. пространстве высотой до 2 м над уровнем пола или площадки, где находятся рабочие места.

Мероприятия по оздоровлению воздушной среды

Требуемое состояние воздуха рабочей зоны может быть обеспечено выполнением определенных мероприятий, к основным из которых относятся:

1. Механизация и автоматизация производственных процессов, дистанционное управление ими. Эти мероприятия имеют большое значение для защиты от воздействия вредных веществ, теплового излучения, особенно при выполнении тяжелых работ. Автоматизация процессов, сопровождающихся выделением вредных веществ, не только повышает производительность, но и улучшает условия труда, поскольку рабочие выводятся из опасной зоны. Например, внедрение автоматической сварки с дистанционным управлением вместо ручной дает возможность резко оздоровить условия труда сварщика, применение роботов-манипуляторов позволяет устранить тяжелый ручной труд.

2. Применение технологических процессов и оборудования, исключающих образование вредных веществ или попадание их в рабочую зону. При проектировании новых технологических процессов и оборудования необходимо добиваться исключения или резкого уменьшения выделения вредных веществ в воздух производственных помещений. Этого можно достичь, например, заменой токсичных веществ нетоксичными, переходом с твердого и жидкого топлива на газообразное, электрический

высокочастотный нагрев; применением пылеподавления водой (увлажнение, мокрый помол) при измельчении и транспортировке материалов и т. д.

Большое значение для оздоровления воздушной среды имеет надежная герметизация, оборудования, в котором находятся вредные вещества, в частности, нагревательных печей, газопроводов, насосов, компрессоров, конвейеров и т. д. Через неплотности в соединениях, а также вследствие газопроницаемости материалов происходит истечение находящихся под давлением газов. Количество вытекающего газа зависит от его физических свойств, площади неплотностей и разницы давлений снаружи и внутри оборудования.

3. Защита от источников тепловых излучений. Это важно для снижения температуры воздуха в помещении и теплового облучения работающих.

4. Устройство вентиляции и отопления, что имеет большое значение для оздоровления воздушной среды в производственных помещениях.

5. Применение средств индивидуальной защиты.

Вентиляция как средство защиты воздушной среды производственных помещений

Задачей вентиляции является обеспечение чистоты воздуха и заданных метеорологических условий в производственных помещениях. Вентиляция достигается удалением загрязненного или нагретого воздуха из помещения и подачей в него свежего воздуха.

По способу перемещения воздуха вентиляция бывает с естественным побуждением (естественной) и с механическим (механической). Возможно также сочетание естественной и механической вентиляции (смешанная вентиляция).

Вентиляция бывает приточной, вытяжной или приточно-вытяжной в зависимости от того, для чего служит система вентиляции, – для подачи

(притока) или удаления воздуха из помещения или (и) для того и другого одновременно.

По месту действия вентиляция бывает общеобменной и местной.

Действие общеобменной вентиляции основано на разбавлении загрязненного, нагретого, влажного воздуха помещения свежим воздухом до предельно допустимых норм. Эту систему вентиляции наиболее часто применяют в случаях, когда вредные вещества, теплота, влага выделяются равномерно по всему помещению. При такой вентиляции обеспечивается поддержание необходимых параметров воздушной Среды во всем объеме помещения.

Воздухообмен в помещении можно значительно сократить, если улавливать вредные вещества в местах их выделения. С этой целью технологическое оборудование, являющееся источником выделения вредных веществ, снабжают специальными устройствами, от которых производится отсос загрязненного воздуха. Такая вентиляция называется местной вытяжкой.

Местная вентиляция по сравнению с общеобменной требует значительно меньших затрат на устройство и эксплуатацию.

В производственных помещениях, в которых возможно внезапное поступление в воздух рабочей зоны больших количеств вредных паров и газов, наряду с рабочей предусматривается устройство аварийной вентиляции.

На производстве часто устраивают комбинированные системы вентиляции (общеобменную с местной, общеобменную с аварийной и т.п.).

Для эффективной работы системы вентиляции важно, чтобы еще на стадии проектирования были выполнены следующие технические и санитарно-гигиенические требования.

1. Количество приточного воздуха должно соответствовать количеству удаляемого (вытяжки); разница между ними должна быть минимальной.

В ряде случаев необходимо так организовать воздухообмен, чтобы одно количество воздуха обязательно было больше другого. Например, при проектировании вентиляции двух смежных помещений, в одном из которых выделяются вредные вещества. Количество удаляемого воздуха из этого помещения должно быть больше количества приточного воздуха, в результате чего в помещении создается небольшое разрежение.

Возможны такие схемы воздухообмена, когда во всем помещении поддерживается избыточное по отношению к атмосферному давление. Например, в цехах электровакуумного производства, для которого особенно важно отсутствие пыли.

2. Приточные и вытяжные системы в помещении должны быть правильно размещены. Свежий воздух необходимо подавать в те части помещения, где количество вредных веществ минимально, а удалять, где выделения максимальны.

Приток воздуха должен производиться, как правило, в рабочую зону, а вытяжка – из верхней зоны помещения.

3. Система вентиляции не должна вызывать переохлаждения или перегрева работающих.

4. Система вентиляции не должна создавать шум на рабочих местах, превышающий предельно допустимые уровни.

5. Система вентиляции должна быть Электра, пожара и взрывобезопасна, проста по устройству, надежна в эксплуатации и эффективна.

Механическая вентиляция. В системах механической вентиляции движение воздуха осуществляется вентиляторами и в некоторых случаях эжекторами.

Приточная вентиляция. Установки приточной вентиляции обычно состоят из следующих элементов (рис. 4.1,*а*): воздухозаборное устройство 1 для забора чистого воздуха; воздуховоды 2, по которым воздух подается в помещение; фильтры 3 для очистки воздуха от пыли; калориферы 4 для нагрева воздуха; вентилятор 5; приточные насадки 6; регулирующие устройства, которые устанавливаются в воздухоприемном устройстве и на ответвлениях воздуховодов.

Вытяжная вентиляция. Установки вытяжной вентиляции включают в себя (рис. 4.1,*б*): вытяжные отверстия или насадки 7; вентилятор 5; воздуховоды 2; устройство для очистки воздуха от пыли и газов 8; устройство для выброса воздуха 9, которое должно быть расположено на 1–1,5 м выше конька крыши.

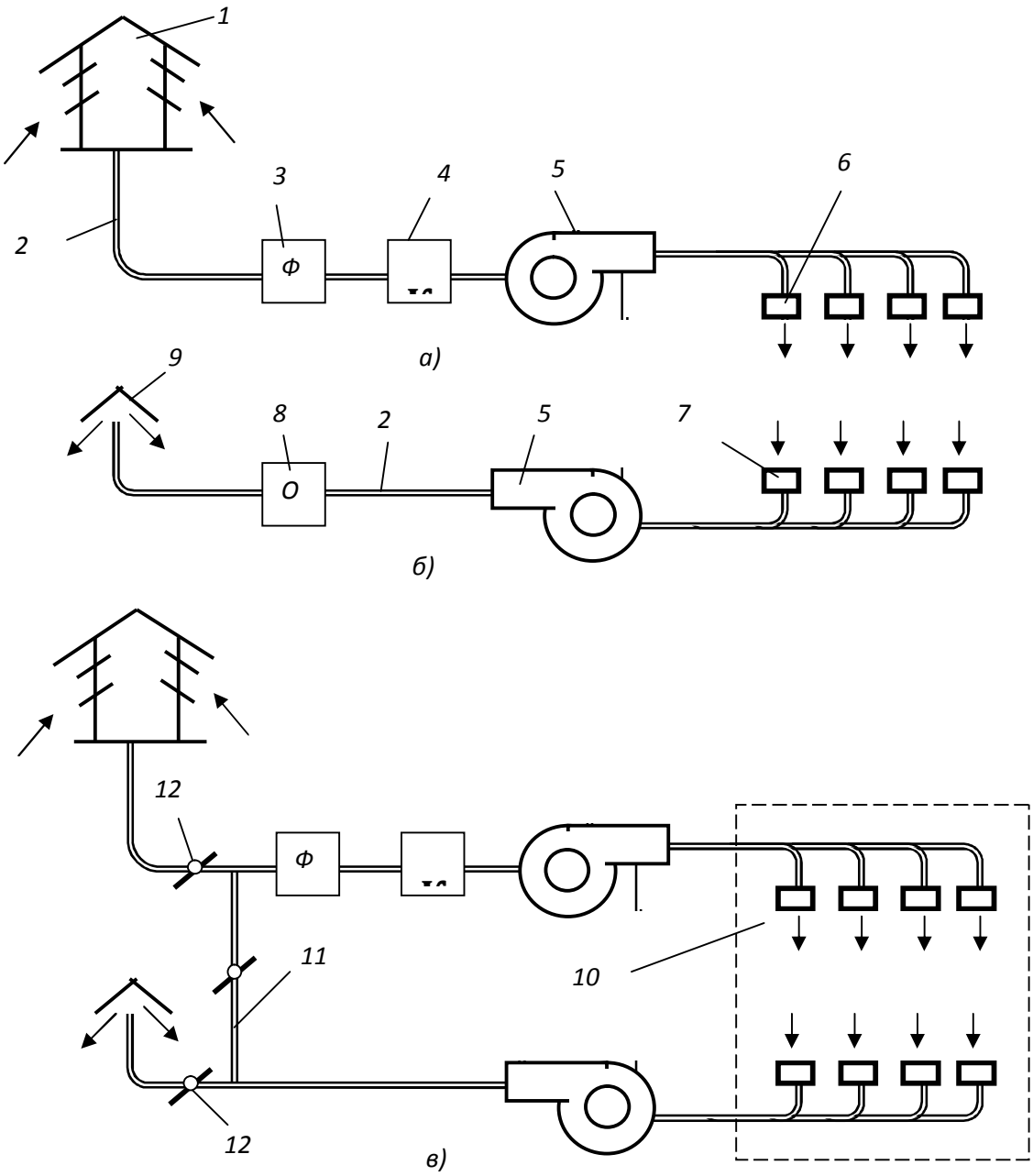

Рис. 8. Механическая вентиляция:

а) – приточная; *б*) – вытяжная; *в*) – приточно-вытяжная.

При работе вытяжной системы чистый воздух поступает в помещение через неплотности в ограждающих конструкциях. В ряде случаев это обстоятельство является серьезным недостатком данной системы вентиляции, так как неорганизованный приток холодного воздуха (сквозняки) может вызвать простудные заболевания.

Приточно-вытяжная вентиляция. В этой системе воздух подается в помещение приточной вентиляцией, а удаляется вытяжной вентиляцией (рис.4.1,*а* и *б*), работающими одновременно.

Приточно-вытяжная вентиляция с рециркуляцией (рис.4.1,*в*) характерна тем, что воздух, отсасываемый из помещения 10 вытяжной системой, частично повторно подают в это помещение через приточную систему, соединенную с вытяжной системой воздуховодом 11. Регулировка количества свежего, вторичного и выбрасываемого воздуха производится клапанами 12. В результате использования такой системы достигается экономия расходуемой теплоты на нагрев воздуха в холодное время года и на его очистку.

Для рециркуляции разрешается использовать воздух помещений, в которых отсутствуют выделения вредных веществ или выделяющиеся вещества относятся к 4-му классу опасности, причем концентрация этих веществ в подаваемом в помещение воздухе не превышает 0,3 концентрации ПДК.

Оборудование для вентиляционных систем.

Вентиляторы – это воздуходувные машины, создающие определенное давление и служащие для перемещения воздуха при потерях давления в вентиляционной сети не более 12 кПа. Наиболее распространенными являются осевые и радиальные (центробежные) вентиляторы.

Осевой вентилятор представляет собой лопаточное колесо, расположенное в цилиндрическом кожухе. При вращении колеса воздух под действием лопаток перемещается в осевом направлении. Преимуществами осевых вентиляторов являются простота конструкции, возможность эффективного регулирования производительности посредством поворота лопаток, большая производительность, реверсивность работы. К недостаткам относятся относительно малая величина давления и повышенный шум.

Радиальный (центробежный) вентилятор состоит из спирального корпуса с размещенным внутри лопаточным колесом. При вращении колеса воздух поступает через входное отверстие в корпусе, попадает между лопатками и под действием центробежной силы перемещается по каналам между лопатками и выбрасывается через выпускное отверстие.

В зависимости от состава перемещаемого воздуха вентиляторы изготовляют из определенных материалов и различной конструкции:

1) обычного исполнения для перемещения чистого воздуха, изготавливаются из обычных сортов стали;

2) антикоррозионного исполнения – для перемещения агрессивных сред, хромистые и хромоникелевые стали винипласт и т.д.;

3) искрозащитного исполнения – для перемещения взрывоопасных смесей (содержащих водород, ацетилен и т.п.), основные детали изготавливаются из алюминия и дюралюминия, устанавливается сальниковое уплотнение на валу;

4) пылевые – для перемещения пыльного воздуха, рабочие колеса изготавливают из материалов повышенной прочности, они имеют мало (4–8) лопаток.

Эжекторы применяют в вытяжных системах в тех случаях, когда необходимо удалить очень агрессивную среду, пыль, способную к взрыву не только от удара, но и от трения, или легко воспламеняющиеся взрывоопасные газы (ацетилен, эфир и т.д.).

Принцип действия эжектора следующий (рис.4.2). Воздух, нагнетаемый расположенным вне помещения компрессором, подводится по трубе 1 (рис.4.2) к соплу 2 и, выходя из него с большой скоростью, создает за счет эжекции разрежение в камере 3, куда подсасывается воздух из помещения. В конфузоре 4 и горловине 5 происходит перемешивание эжектируемого

(из помещения) и эжектирующего воздуха. Диффузор 6 служит для преобразования динамического давления в статическое. Недостатком эжектора является низкий к.п.д, не превышающий 0,25.

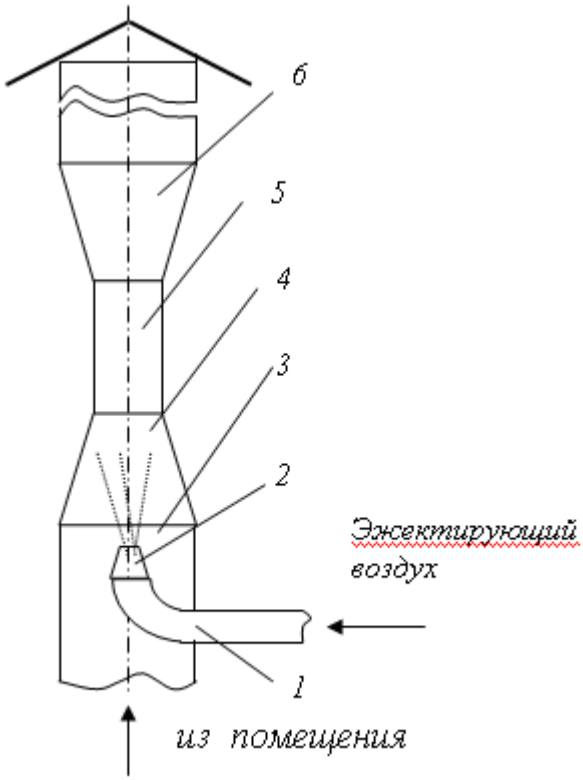

Рис. 9. Эжектор

4.2. Пожарная безопасность.

Пожар - это горение вне специального очага, которое не контролируется и может привести к массовому поражению и гибели людей, а также к нанесению экологического, материального и другого вреда.

Горение - это химическая реакция окисления, сопровождающаяся выделением теплоты и света. Для возникновения горения требуется наличие трех факторов: горючего вещества, окислителя и источника загорания. Окислителями могут быть кислород, хлор, фтор, бром, йод, окиси азота и другие. Кроме того, необходимо чтобы горючее вещество было нагрето до определенной температуры и находилось в определенном количественном соотношении с окислителем, а источник загорания имел определенную энергию.

Наибольшая скорость горения наблюдается в чистом кислороде. При уменьшении содержания кислорода в воздухе горение прекращается. Горение при достаточной и надмерной концентрации окислителя называется полным, а при его нехватке - неполным.

Выделяют три основных вида самоускорения химической реакции при горении: тепловой, цепной и цепочно-тепловой. Тепловой механизм связан с экзотермичностью процесса окисления и возрастанием скорости химической реакции с повышением температуры. Цепное ускорение реакции связано с катализом превращений, которое осуществляют промежуточные продукты превращений. Реальные процессы горения осуществляются, как правило, по комбинированному (цепочно-тепловой) механизму.

Процесс возникновения горения подразделяется на несколько видов.

Вспышка - быстрое сгорание горючей смеси, не сопровождающееся образованием сжатых газов.

Возгорание - возникновение горения под воздействием источника зажигания.

Воспламенение - возгорание, сопровождающееся появлением пламени.

Самовозгорание - явление резкого увеличения скорости экзотермических реакций, приводящее к возникновению горения вещества при отсутствии источника зажигания.

Самовоспламенение - самовозгорание, сопровождается появлением пламени.

Взрыв - чрезвычайно быстрое (взрывчатое) превращение, сопровождающееся выделением энергии с образованием сжатых газов.

Основными показателями пожарной опасности являются температура самовоспламенения и концентрационные пределы воспламенения.

Температура самовоспламенения характеризует минимальную температуру вещества, при которой происходит резкое увеличение скорости экзотермических реакций, заканчивающееся возникновением пламенного горения.

Температура вспышки - самая низкая (в условиях специальных испытаний) температура горючего вещества, при которой над поверхностью

образуются пары и газы, способные вспыхивать в воздухе от источника зажигания, но скорость их образования еще недостаточна для последующего горения.

Горючими называются вещества, способные самостоятельно гореть после изъятия источника загорания.

По степени горючести вещества делятся на: горючие (сгораемые), трудногорючие (трудносгораемые) и негорючие (несгораемые).

К трудногорючим относятся такие вещества, которые не способны распространять пламя и горят лишь в месте воздействия источника зажигания.

Негорючими являются вещества, не воспламеняющиеся даже при воздействии достаточно мощных источников зажигания (импульсов).

Горючие вещества могут быть в трех агрегатных состояниях: жидком, твердом и газообразном. Большинство горючих веществ независимо от агрегатного состояния при нагревании образует газообразные продукты, которые при смешении с воздухом, содержащим определенное количество кислорода, образуют горючую среду. Горючая среда может образоваться при тонкодисперсном распылении твердых и жидких веществ.

Из горючих газов и пыли образуются горючие смеси при любой температуре, в то время как твердые вещества и жидкости могут образовать горючие смеси только при определенных температурах.

В производственных условиях может иметь место образование смесей горючих газов или паров в любых количественных соотношениях. Однако взрывоопасными эти смеси могут быть только тогда, когда концентрация горючего газа или пара находится между границами воспламеняемых концентраций.

Минимальная концентрация горючих газов и паров в воздухе, при которой они способны загораться и распространять пламя, называющееся *нижним концентрационным пределом воспламенения* .

Максимальная концентрация горючих газов и паров, при которой еще возможно распространение пламени, называется *верхним концентрационным пределом воспламенения.*

Указанные пределы зависят от температуры газов и паров: при увеличении температуры на 100°С величины нижних пределов воспламенения уменьшаются на 8 -10 %, верхних - увеличиваются на 12 - 15 %.

Пожарная опасность вещества тем больше, чем ниже нижний и выше верхний пределы воспламенения и чем ниже температура самовоспламенения.

Пыли горючих и некоторых не горючих веществ (например алюминий, цинк) могут в смеси с воздухом образовать горючие концентрации.

Наибольшую опасность по взрыву представляет взвешенная в воздухе пыль. Однако и осевшая на конструкциях пыль представляет опасность не только с точки зрения возникновения пожара, но и вторичного взрыва, вызываемого в результате взвихривания пыли при первичном взрыве.

Минимальная концентрация пыли в воздухе, при которой происходит ее загорание, называется *нижним пределом воспламенения пыли* .

Поскольку достижение очень больших концентраций пыли во взвешенном состоянии практически нереально, термин "верхний предел воспламенения" к пылям не применяется.

Воспламенение жидкости может произойти только в том случае, если над ее поверхностью имеется смесь паров с воздухом в определенном количественном соотношении, соответствующим нижнему температурному пределу воспламенения.

Меры по пожарной профилактике.

Мероприятия по пожарной профилактике разделяются на организационные, технические, режимные и эксплуатационные.

Организационные мероприятия: предусматривают правильную эксплуатацию машин и внутризаводского транспорта, правильное содержание зданий, территории, противопожарный инструктаж и тому подобное.

Технические мероприятия: соблюдение противопожарных правил и норм при проектировании зданий, при устройстве электропроводов и оборудования, отопления, вентиляции, освещения, правильное размещение оборудования.

Режимные мероприятия - запрещение курения в неустановленных местах, запрещение сварочных и других огневых работ в пожароопасных помещениях и тому подобное.

Эксплуатационные мероприятия – своевременная профилактика, осмотры, ремонты и практика тушения пожаров наибольшее распространение получили следующие принципы прекращения горения:

- изоляция очага горения от воздуха или снижение концентрации кислорода путем разбавления воздуха негорючими газами (углеводы CO $i <$ 12 - 14 %).

- охлаждение очага горения ниже определенных температур;

- интенсивное торможение (ингибирование) скорости химической реакции в пламени;

- механический срыв пламени струей газа или воды;

- создание условий огнепреграждения (условий, когда пламя распространяется через узкие каналы).

Литература

Брюс Шнайер - Секреты и ложь. Безопасность данных в цифровом мире 2003 — 373с.

Ю.В.Романец, П.А.Тимофейев. Защита информации в компьютерных системах и сетях. 2 изд. “ Радио и Связ ” 2001 г

Желников В. Криптография от папируса до компьютера. Москва. ABF. 1997.

А. А. Журабаев, З. Г. Рахимов, Организационно-правовое обеспечение информационной безопасности Республики Узбекистан г. Ташкент, Узбекистан

Чмора . Современная прикладная криптография

Домарев В.В. Безопасность информационных технологий. Методология создания систем защиты. — К.: ООО “ДС”, 2001. — 688 с.

Rivest RL. The MD5 message digest algorithm // Advances in Cryptology – CRYPTO’90. LNCS. Springer-Verlag. 1991. V.537. P.303-311.

Коллизии хеш-функций MD5, SHA-0,1 (Collisions in hash functions MD5, SHA-0,1)