Вестник РУДН, Серия *Прикладная и компьютерная математика*. Т. 2, № 2. 2003. с. 123–151

Разработка программного обеспечения и системное администрирование

УДК 681.322

Инфраструктура открытых ключей

Д. С. Кулябов, А. В. Королькова

Кафедра систем телекоммуникаций
Российский университет дружбы народов
Россия, 117198, Москва, ул. Миклухо-Маклая, 6

В статье дан обзор теоретических основ PKI. Рассмотрены её основные понятия и компоненты. Даны практические рекомендации по развёртыванию PKI в рамках организации.

Ключевые слова: сертификат, открытый ключ, закрытый ключ, ЭЦП, PKI, CA, SSL, openssl.

1. Введение

На данном этапе развития информационных технологий на первый план всё сильнее выходит проблема безопасности. В частности, необходима защита каналов передачи данных, проверка целостности сообщений и т. д.

К сожалению, при администрировании систем безопасности администраторы применяют либо чисто эмпирический, либо инженерный подходы. Что подразумевает поверхностное отношение к предмету, бездумное использование «готовых рецептов».

Инфраструктура открытых ключей не достаточно широко освещена в литературе, что препятствует грамотному использованию данной технологии. Поэтому в данной статье вопросы практического применения PKI предварены теоретическим обзором.

В разделе 2 приведён краткий обзор криптографических примитивов, необходимых для понимания основ работы инфраструктуры открытых ключей.

В разделе 3 дан обзор инфраструктуры открытых ключей. В частности, даны основные определения, описаны компоненты, механизмы работы, дано описание стандартов PKI и стандартов, использующих PKI. Кроме того, подробно рассмотрен формат сертификата.

В разделе 4 даны рекомендации по развёртыванию инфраструктуры открытых ключей.

2. Криптографические примитивы и механизмы

Криптография является одним из основных инструментов, обеспечивающих конфиденциальность, доверие, авторизацию, электронные платежи, корпоративную безопасность и множество других важных вещей [1, 2].

Криптографические методы могут применяться для решения следующих проблем безопасности:

— конфидециальности передаваемых или хранимых данных;
— аутентификации;

— целостности передаваемых и хранимых данных;
— обеспечения подлинности документов.

Базовые методы преобразования информации:

— шифрование (симметричное и несимметричное);
— вычисление хэш-функций;
— генерация электронной цифровой подписи (ЭЦП);
— генерация последовательности псевдо-случайных чисел.

Шифрование — обратимое преобразование данных с целью их сокрытия от посторонних.

Почти все методы шифрования используют ключ шифрования. *Ключ шифрования* — секретная кодовая последовательность, используемая в процессе преобразования информации.

2.1. Ассимметричные алгоритмы

Криптование использует пару — закрытый и открытый ключи, обеспечивая тем самым то, что данные могут быть зашифрованы одним ключом, а расшифрованы могут быть только другим ключом. В действительности ключи похожи и могут использоваться альтернативно: один ключ пары шифрует, другой — расшифровывает.

Ключи основаны на больших простых числах, длина которых в битах является достаточной для того, чтобы было невозможно расшифровать сообщение, не зная второго ключа пары.

Открытый ключ может быть доступен каждому, в то время как второй (секретный) ключ должен храниться в надежном месте. Любой может отправить Вам зашифрованное Вашим открытым ключом сообщение, но только Вы можете его открыть (расшифровать и прочитать). И наоборот, Вы можете засертифицировать сообщение так, чтобы было ясно, что именно Вы зашифровали его Вашим секретным ключом и только ассоциированный открытый ключ может корректно расшифровать сообщение.

Криптография с открытыми ключами обеспечивает все требования, предъявляемые к криптографическим системам. Но реализация алгоритмов требует больших затрат процессорного времени. Поэтому в чистом виде криптография с открытыми ключами в мировой практике обычно не применяется. Для шифрования данных используются симметричные (сеансовые) ключи, которые в свою очередь шифруются с использованием открытых ключей для передачи сеансовых ключей по сети.

Алгоритм RSA Безопасность алгоритма RSA основана на трудоемкости разложения на множители (факторизации) больших чисел. Открытый и закрытый ключи являются функциями двух больших простых чисел, разрядностью 100–200 десятичных цифр. Предполагается, что восстановление открытого текста по шифрованному тексту и открытому ключу равносильно разложению числа на два больших простых множителя.

2.2. Симметричные алгоритмы

Симметричным ключом называется ключ, используемый как для шифрования, так и для расшифровки данных. Симметричный ключ является потенциально небезопасным. Если злоумышленник сможет перехватить ключ, то информация не будет более являться секретной. Нельзя передавать симметричный ключ в открытом виде. Симметричный ключ вкладывается внутрь сообщения, зашифрованного асимметричным ключом.

Преимущества криптографии с симметричными ключами:

— производительность
  Производительность алгоритмов с симметричными ключами очень велика.
— стойкость
  Криптография с симметричными ключами очень стойкая, что делает практически невозможным процесс дешифрования. При прочих равных условиях

(общий алгоритм) стойкость определяется длиной ключа. При длине ключа 256 бит необходимо произвести 10^{77} переборов для определения ключа.

Недостатки криптографии с симметричными ключами:

— распределение ключей
Так как для шифрования и расшифрования употребляется один и тот же ключ, при использовании криптографии с симметричными ключами требуются очень надежные механизмы для распределения ключей.

— масштабируемость
Так как используется единый ключ между отправителем и каждым из получателей, количество необходимых ключей возрастает в геометрической прогрессии. Для 10 пользователей нужно 45 ключей, а для 1000 — уже 499500.

— ограниченное использование
Так как криптография с симметричными ключами используется только для шифрования данных и ограничивает доступ к ним, при ее использовании невозможно обеспечить аутентификацию и неотрекаемость.

Таблица 1

Симметричные блочные шифры

Алгоритм	Длина ключа (в битах)	Длина блока (в битах)
DES	64	64
Blowfish	переменная, до 448	64
IDEA	128	64
ГОСТ 28147-89	256	64
RC5	переменная	переменная

Алгоритм DES DES предназначен для шифрования данных 64-битовыми блоками. Алгоритм представляет собой комбинацию двух методов шифрования: рассеивания и перемешивания [1]. Длина ключа — 56 бит (8 байт, в каждом октете один бит — бит чётности).

Алгоритм RC2 RC2 предназначен для замены DES и представляет собой шифр с 64-битовым блоком и имеет переменную длину ключа. Скорость шифрования не зависит от размера ключа.

Алгоритм IDEA International Data Encription Algorithm (IDEA) — симметричный блочный шифр, работающий с 64-битовыми блоками открытого текста и ключами длиной 128 бит.

Алгоритм ГОСТ 28147-89 ГОСТ представляет собой 64-битовый алгоритм с 356-битовым ключом. По своей структуре он очень близок к алгоритму DES.

Алгоритм Blowfish Blowfish представляет собой 64-битовый блочный алгоритм шифрования с ключом переменной длины. Алгоритм состоит из двух частей: расширения ключа и шифрования данных. Расширение ключа преобразует ключ длиной 448 бит в несколько массивов подключей общим размером 4168 байт.

В алгоритме используется множество подключей, которые должны быть вычислены до начала шифрования и расшифрования данных.

Алгоритм RC5 RC5 представляет собой блочный шифр с множеством параметров: размером блока, размером ключа, и числом раундов. В RC5 предусмотрены три операции: XOR, сложение и циклические сдвиги [1]. Переменные циклические сдвиги представляют собой нелинейную функцию.

2.3. Хэш-функция

Криптографическими методами можно обеспечить не только конфидециальность, но и проконтролировать целостность передаваемых или хранимых данных. Контроль целостности в основном производится путем расчета некоторой «контрольной суммы» данных.

Для многих приложений простой контрольной суммы оказывается достаточно, особенно если важна скорость обработки данных и заранее не известен объём данных.

Проблема простых алторитмов вычисления контрольной суммы в том, что достаточно легко подобрать несколько массивов данных, имеющих одинаковую контрольную сумму. Криптографически стойкие контрольный суммы вычисляются как результат применения к исходному тексту хэш-функции.

Все используемые в настоящее время хэш-функции являются так называемыми кандидатами в односторонние функции.

Под *односторонней функцией* понимается функция, определенная, например, на множестве натуральных чисел и не требующая для вычисления своего значения больших вычислительных ресурсов. Но вычисление обратной функции (т.е. по известному значению функции восстановить значение аргумента) оказывается невозможно теоретически или невозможно вычислительно. Строгое существование односторонних функций пока не доказано.

Основными свойствами криптографически-«хорошей» хэш-функции являются свойство рассеивания, свойство стойкости к коллизиям и свойство необратимости.

О необратимости уже было сказано. *Коллизией хэш-функции* H называется ситуация, при которой существуют два рзличных текста T_1 и T_2, но $H(T_1) = H(T_2)$. Значение хэш-функции всегда имеет фиксированную длину, а на длину исходного текста не накладывается никаких ограничений. Из этого следует, что коллизии существуют. Требование стойкости к коллизиям означает, что для криптографически-«хорошей» хэш-функции для заданного текста T_1 вычислительно невозможно найти текст T_2, вызывающий коллизию.

Свойство рассеивания требует, чтобы минимальные изменения текста, подлежащего хэшированию, вызывали максимальные изменения в значении хэш-функции.

Основными применяемыми на сегодняшний день алгоритмами, реализующими хэш-функции, являются MD2, MD4, MD5, SHA и его вариант SHA1, российский алгоритм, описанный стандартом ГОСТ Р 34.11–94. Наиболее часто используются MD5, SHA1 и в России ГОСТ Р 34.11–94.

Алгоритмы MD2, MD4, MD5 Хэш-функции в алгоритмах MD2, MD4, MD5 преобразуют входное сообщение произвольной длины в сжатый 128-битный образ.

Основные операции: сложение по модулю 2^{32}, циклический сдвиг, логические операции $\oplus, \wedge, \vee$.

Алгоритм MD2 отличается от MD4, MD5 способом дополнения сообщения (с применением 16-байтной контрольной суммы), значением стартового вектора хэширования и использованием 256-байтной перестановки, а обработка одного сообщения в нем выполняется за 18 раундов.

Алгоритм SHA-1 Алгоритм SHA-1 является модифицированной версией алгоритма SHA (Secure Hash Algorithm), который используется в стандарте на хэш-функцию SHD (Secure Hash Standart).

Алгоритм SHA-1 сопоставляет сообщению длиной до 2^{64} бит сжатый 160-битный образ, используемый, в частности, для генерации и проверки цифровой подписи в стандарте DSS. Кроме того, алгоритм SHA-1 создан по образцу MD4.

Алгоритм ГОСТ Р34.11-94 Алгоритм вычисления хэш-функции ГОСТ Р34.11-94 сопоставляет сообщению произвольной длины 256-битный сжатый образ, который используется, в частности, для генерации и проверки цифровой подписи в стандарте ГОСТ Р34.10-94.

3. Инфраструктура открытых ключей

3.1. Концепция инфраструктуры открытых ключей

Инфраструктура открытых ключей — Public Key Infrastructure (PKI) — это набор служб, которые призваны обеспечить работу криптографических методов с открытыми ключами.

Технология PKI заключается в использовании двух математически-связанных цифровых ключей, имеющих следующие свойства:

— один ключ может использоваться для шифрования сообщения, которое может быть расшифровано только с помощью второго ключа;
— даже если известен один ключ, с помощью вычислений абсолютно невозможно определить второй. Один из ключей открыт для всех, второй же имеет частный характер и хранится в защищенном месте. Эти ключи могут использоваться для аутентификации, шифрования или цифровой подписи электронных данных.

PKI служит не только для создания цифровых сертификатов, но и для хранения огромного количества сертификатов и ключей, обеспечения резервирования и восстановления ключей, взаимной сертификации, ведения списков аннулированных сертификатов и автоматического обновления ключей и сертификатов после истечения срока их действия.

3.2. Хранение ключей

Согласно правилу Кирхгоффа стойкость шифра и, как следствие, стойкость ЭЦП, должна определяться только секретностью ключа, используемого для шифрования или подписи сообщения. Как следствие, секретные ключи никогда не должны храниться в явном виде на носителях, которые могут быть скопированы. Во многих существующих разработках этим условием пренебрегают, оставляя защиту секретных ключей на совести пользователя системы ЭЦП. В некоторых случаях разработчики предлагают варианты хранения на носителях, которые, по их словам, трудно копируются. Например, «таблетки» Touch Memory, смарт-карты или бесконтактные карты Proximity. Однако в последнее время за рубежом участились случаи «удачных» атак на такие носители. Эти случаи преимущественно зафиксированы за рубежом, но российские «умельцы» ни в чем не уступают своим западным коллегам, и можно предсказать, что скоро случаи попыток «взлома» аппаратных носителей будут зафиксированы и в России.

Для повышения надежности таких схем хранения рекомендуется секретные ключи шифровать на других ключах, которые, в свою очередь, могут быть тоже зашифрованы. Самый последний ключ в этой иерархии называется главным или мастер-ключом и не должен шифроваться. Однако к нему предъявляются очень жесткие требования к хранению в защищенной части компьютера или аппаратуры, реализующей функции цифровой подписи.

3.3. Распределение ключей

Очень важный вопрос при выборе системы ЭЦП — это распределение ключей между абонентами, участвующими в обмене защищаемыми документами. Такое распределение может осуществляться двумя способами:

— путем создания центра генерации и распределения ключей

  Недостаток такого подхода очевиден. Центр обладает полной информацией о том, кто и какой ключ использует. Компрометация центра распределения приводит к компрометации всей передаваемой между абонентами этого центра информации. Кроме того, знание секретных ключей абонентов позволяет нечистым на руку сотрудникам центра фальсифицировать определенные документы, передаваемые в системе обмена информацией.
— путем прямого обмена ключами между абонентами, которые хотят обмениваться подписанными сообщениями

В этом случае основная задача — подтверждение подлинности каждого абонента, участвующего в обмене.

Подтверждение подлинности абонентов в последнем случае может осуществляться следующим образом:

— непосредственно между абонентами

Данный метод применяется в том случае, если абонентов всего двое. Для обмена ключами в данном случае может быть использован алгоритм распределения ключей, например, разработанный в 1976 году криптографами Диффи и Хеллманом. Существуют и другие варианты обмена ключами. Например, при помощи симметричных криптосистем или фельдъегерской службы. Однако, в распределенных сетях, насчитывающих не один десяток абонентов, такие варианты не применимы из-за возникающих сложностей.

— с использованием посредника (арбитра)

Данный метод может применяться в корпоративных сетях, в которых существует так называемый центр верификации или сертификации ключей. Данный центр удостоверяет ключи, используемые для проверки подписи. Подтверждение подлинности ключей может реализовываться или путем формирования справочника открытых ключей, или путем выдачи сертификатов, которые передаются вместе с сообщением, требующим проверки. Данный сертификат представляет собой ключ для проверки подписи и некоторую аутентифицирующую информацию, скрепленные подписью Центра сертификации. В данном случае достаточно проверить подпись Центра, указанную в сертификате, чтобы удостовериться в подлинности ключа абонента.

— с использованием двух и более посредников

Этот метод, являющийся комбинацией двух предыдущих, может применяться в том случае, когда необходимо обеспечить обмен подписанными сообщениями между несколькими корпоративными сетями, в каждой из которых существует свой центр сертификации.

3.4. Подходы к реализации PKI

В настоящее время можно выделить несколько подходов к реализации инфраструктуры открытых ключей [3]:

— защищённая DNS;
— PKI стандарта OpenPGP;
— PKIX — PKI, основанная на сертификатах формата X.509 (PKI for X.509 certificates).

3.4.1. Защищенная DNS

Предложено и реализовано расширение DNS, выполняющее аутентификацию данных путём применения цифровых подписей. Расширение предусматривает хранение открытых ключей в DNS.

Некоторые DNS позволяют включать помимо записей, содержащих информацию об IP-адресе и о сервере имен, записи *KEY* и *SIG*, которые обеспечивают распространение ключа и аутентификацию исходных данных.

Запись *KEY* позволяет связывать открытые ключи с именами DNS. Из-за относительно простого формата имён DNS запись *KEY* содержит флаговые биты, которые указывают вид имени DNS и ограничения использования ключа. Предполагается, что некоторые ключи защищённой DNS будут использоваться совместно с другими протоколами интернет. Байт протокола указывает, что данный ключ может использоваться в других протоколах. Байт алгоритма определяет алгоритм шифрования, для которого предназначен данный ключ.

Запись *SIG* используется для аутентификации других записей, связывая их с доменным именем. Запись *SIG* должна содержать следующие поля: тип записей, защищаемых данной записью, алгоритм, счётчик меток, время жизни, срок прекращения действия подписи, время подписания, имя подписавшего, подпись.

К сожалению, организационная часть, связанная с выделением сертификатов доменам, не выполнена и вряд ли в ближайшей перспективе сертификаты будут выданы. Поэтому защищённая DNS имеет исключительно теоретический интерес.

3.4.2. PGP

PGP используется для защиты сообщений и файлов с помощью их шифрования и цифровой подписи. PGP позволяет шифровать, подписывать, расшифровывать и проверять сообщения. В PGP используются следующие алгоритмы: CAST, 3DES, IDEA; для выработки сеансового ключа используются алгоритмы RSA и Диффи–Хэллмана; для подписи используются алгоритмы RSA и DSS [1].

PGP обеспечивает интегрированную поддержку распространения и поиска открытых ключей на серверах ключей.

Перед использованием PGP генерятся открытый и закрытый ключи. Открытый ключ передаётся непосредственно абоненту либо помещается на сервер открытых ключей. Получив копию чьего-либо открытого ключа, пользователь может добавить его на свою связку открытых ключей. Пользователь должен сам проверить валидность полученного ключа, после чего он подписывает его. Кроме того, пользователь должен указать степень доверия к владельцу данного ключа в смысле его способности ручаться за подлинность ключей третьих лиц. Таким образом, формируется сеть поручительства абонентов сети за достоверность распространяемых ключей, так называемая *сеть доверия*.

3.4.3. PKIX

Данный вариант PKI [4, 5] стал фактически индустриальным стандартом. Статья, собственно, и посвящена его описанию.

3.5. Компоненты инфраструктуры открытых ключей и их функции

Эффективная PKI должна включать следующие элементы:

— доверенный центр;
— архив сертификатов;
— систему аннулирования сертификатов;
— систему создания резервных копий и восстановления ключей;
— систему поддержки невозможности отказа от цифровых подписей;
— систему автоматической корректировки пар ключей и сертификатов;
— систему управления «историей» ключей;
— систему поддержки взаимной сертификации;
— клиентское программное обеспечение, взаимодействующее со всеми этими подсистемами безопасным, согласованным и надежным способом.

Сертификат (Certificate) и соответствующий ему секретный ключ позволяют идентифицировать их владельца. Универсальное применение сертификатов обеспечивает стандарт Международного Телекоммуникационного Союза X.509, который является базовым и поддерживается целым рядом протоколов безопасности.

Стандарт X.509 задает формат цифрового сертификата. Основными атрибутами сертификата являются имя и идентификатор субъекта, информация об открытом ключе субъекта, имя, идентификатор и цифровая подпись уполномоченного по выдаче сертификатов, серийный номер, версия и срок действия сертификата, информация об алгоритме подписи и др. Важно, что цифровой сертификат включает в себя цифровую подпись на основе секретного ключа доверенного центра.

Центр сертификации (ЦС, Certificate Authority—CA) *или Доверенный центр* [1] — объект, который авторизован создавать, подписывать и публиковать сертификаты. CA имеет полномочия идентифицировать пользователей. Основными операциями, которые выполняет CA, являются:

— издание сертификата;
— обновление сертификата;
— аннулирование сертификата.

Действия CA ограничены политикой сертификации, которая диктует ему, какую информацию он должен помещать в сертификат. CA публикует свою политику сертификации таким способом, чтобы пользователи могли проверить соответствие сертификатов этой политике.

Список аннулированных сертификатов (Certificate Revocation List—CRL) — список сертификатов, признанных недействительными в период их действия в случае компрометации секретного ключа или изменения атрибутов сертификата с момента его выпуска.

Хранилище сертификатов — специальный объект PKI, где хранятся выпущенные сертификаты и списки отозванных сертификатов. Оно не является обязательным компонентом PKI, но оно значительно упрощает доступ к ресурсам и управление системой. К хранилищу предъявляют следующие требования:

— простота доступа;
— доступ должен быть стандартным;
— обновление информации;
— встроенная защищенность;
— простое управление;
— совместимость с другими хранилищами (необязательное требование).

Хранилище упрощает систему распространения CRL.

Фактически действующим стандартом доступа к хранилищам является LDAP (Lightweight Directory Access Protocol, упрощенный протокол доступа к каталогу). Он наиболее адекватен в качестве стандарта для сохранения и извлечения сертификатов после их генерации, поддерживается большинством серверных операционных систем и баз данных и достаточно открыт для того, чтобы его могли поддерживать практически любые инфраструктуры с открытыми ключами.

Центр регистрации (Registration Authority—RA) является дополнительным компонентом системы PKI, который авторизован CA аутентифицировать пользователей и проверять информацию, которая заносится в сертификат. В его функции может входить генерация и архивирование ключей, сообщение об аннулировании сертификатов и др. В некоторых системах CA выполняет функции RA. CA выдаёт сертификат RA (если он присутствует в системе), и RA выступает как объект, подчинённый CA. Но RA не может выпускать сертификаты и CRL.

Конечный пользователь (End Entity—EE) — пользователь сертификата PKI и/или владелец сертификата. То есть конечный пользователь — это объект, который использует некоторые сервисы и функции системы PKI. Конечный пользователь может быть владельцем сертификата (человек, организация или иная сущность) или объектом, запрашивающим сертификат или CRL.

[1] В настоящее время не существует общепризнанного русского аналога термина, который берет начало в области шифрования с открытыми ключами, — Certificate Authority. Это понятие получило множество совершенно разных названий: служба сертификации, уполномоченный по выпуску сертификатов, распорядитель сертификатов, орган выдачи сертификатов, доверенный центр, центр сертификации, сертификатор и т. д. В случае отсутствия общепринятого русского аналога в данной статье будут использоваться англоязычные аббревиатуры.

3.6. Сервисы PKI

3.6.1. Сервисы управления сертификатами

Сервисы управления сертификатами — это сервисы, образующие ядро инфраструктуры с открытыми ключами. К ним относятся:

— выпуск сертификата

Сертификаты выпускаются для пользователей (физических и юридических лиц), для доверенных центров, находящихся на более низких уровнях иерархии доверия, а также для других доверенных центров в случае их взаимной сертификации.

— аннулирование сертификата

Если пользователь теряет свой секретный ключ, если ключ похищается или компрометируется, или есть вероятность наступления таких событий, действие сертификата должно быть прекращено. После получения подтверждения запроса пользователя об аннулировании сертификата CA уведомляет об аннулировании все заинтересованные стороны, используя CRL.

Аналогично аннулированию осуществляется приостановление действия сертификата. Оно заключается в однократной отмене сертификата на определенный период времени в течение срока его действия. После этого действие сертификата возобновляется автоматически или же сертификат аннулируется. Приостановление действия сертификата осуществляется в тех ситуациях, когда невозможно установить подлинность лица, обращающегося с запросом об аннулировании.

— публикация сертификата

Выпущенный однократно сертификат включается в каталог (в соответствии со спецификациями стандарта X.500 или иными требованиями), чтобы третьи стороны могли иметь к нему доступ. В одних случаях каталог контролируется доверенным центром, в других — третьей стороной.

Доступ к каталогу может быть ограничен. Если необходимо соблюдение прав приватности абонентов, применяются профилактические меры для защиты от лиц, не имеющих полномочий доступа.

— хранение сертификата в архиве

Выпускаемые сертификаты и списки аннулированных сертификатов хранятся в архиве длительное время. Это делается потому, что заверенный цифровой подписью документ продолжает свое существование и после истечения срока действия сертификата. Следовательно, сертификаты с истекшим сроком действия должны быть по-прежнему доступны.

— выработка политики CA

Для реализации операций сертификации формируется политика операционной работы CA, работы с персоналом и оборудованием и политика выпуска сертификатов на основе критериев контроля за созданием сертификатов для пользователей и других доверенных центров.

3.6.2. Вспомогательные сервисы

В инфраструктуре с открытыми ключами могут поддерживаться также различные дополнительные сервисы.

— регистрация

Регистрационные сервисы обеспечивают регистрацию и контроль индивидуальной информации, а также аутентификацию, необходимую для выпуска или аннулирования сертификатов (от имени доверенного центра). Фактический выпуск сертификатов осуществляет CA.

— хранение информации в архиве

Сервисы хранения информации в архиве предназначены для долговременного хранения и управления цифровыми документами и другой информацией. Сервисы обеспечивают создание резервных копий и восстановление информации в случае уничтожения или старения среды хранения.

— нотариальная аутентификация

Нотариальная аутентификация включает аутентификацию отправителя сообщения, подтверждение целостности и юридической силы цифровых документов.

— создание резервных копий и восстановление ключей

CA должен иметь возможность восстановить зашифрованную информацию в случае потери пользователями их ключей шифрования. Это означает, что доверенному центру, к которому относится пользователь, необходима система создания резервных копий и восстановления этих ключей. Этот процесс известен как коммерческое создание резервных копий и восстановление ключей, и он отличается от принудительного депонирования ключей третьей стороной (обычно правоохранительными органами), которая получает доступ к ключам для расшифровки необходимой информации. Коммерческие сервисы восстановления ключей обеспечивают заблаговременное засекречивание копии ключа на случай утери ключа пользователем, его ухода с работы, забывания пароля, необходимого для доступа к ключу, и восстановление ключа в ответ на запрос пользователя или его работодателя. В одних случаях ключ является секретным ключом из алгоритма с открытыми ключами, в других — это распределяемый ключ.

— каталог

Сервисы каталога осуществляют всестороннее управление и обеспечение информацией, имеющей отношение к пользователю. К атрибутам пользователя относится не только сертификат, но и номер телефона, адрес электронной почты абонента и т. д.

— поддержка невозможности отказа от цифровых подписей

При бумажной технологии подписи лиц законно связывают их с документами, что не позволяет в дальнейшем отказаться от подписания документа. При электронных технологиях обычная подпись заменяется цифровой. Самое главное требование для невозможности отказа от цифровой подписи состоит в том, что ключ, используемый для выработки цифровых подписей — ключ подписи, должен создаваться и безопасно храниться все время исключительно под контролем пользователя. Когда пользователи забывают свои пароли или теряют свои ключи подписи, на резервирование или восстановление предыдущей пары ключей подписи не накладывается никаких технических ограничений (в отличие от аналогичной ситуации с парами ключей шифрования сообщений). В таких случаях допускается генерация и дальнейшее использование пользователями новых пар ключей подписи.

Параллельное функционирование систем резервного копирования и восстановления ключей и системы поддержки невозможности отказа от цифровых подписей вызывает определенные проблемы. При резервном копировании и восстановлении ключей должны создаваться копии секретных ключей пользователя. Для обеспечения невозможности отказа от цифровой подписи не должны создаваться резервные копии секретных ключей пользователя, используемых для выработки цифровой подписи. Для соблюдения этих требований в инфраструктуре с открытыми ключами должны поддерживаться две пары ключей для каждого пользователя. В любой момент времени пользователь должен иметь одну пару ключей для шифрования и дешифрования, а другую пару — для выработки или проверки цифровой подписи.

— корректировка ключей и управление историями ключей

В ближайшем будущем в распоряжении пользователей будет находиться огромное количество пар ключей, которые должны будут поддерживаться как криптографические ключи, даже если никогда не будут использоваться. Ключи шифрования должны со временем обновляться и должна поддерживаться история всех ключей, использованных ранее.

Процесс корректировки пар ключей должен быть «прозрачен» для пользователя. Это означает, что пользователи не должны понимать, что осуществляется обновление ключей, и никогда не должны получать отказ сервиса из-за недействительности своих ключей. Для удовлетворения этого требования пары ключей пользователя должны автоматически обновляться до истечения срока

их действия. При обновлении пары ключей подписи предыдущий ключ подписи безопасно уничтожается. Тем самым предотвращается получение несанкционированного доступа к ключу подписи и устраняется необходимость хранения предыдущих ключей подписи.

— другие сервисы

В ряде случаев необходимы и другие сервисы, например, сервисы генерации пар ключей и записи их на смарт-карты.

3.7. Форматы сертификатов

Наиболее распространен формат сертификата, установленный Международным Телекоммуникационным Союзом (ITU Rec. X.509 | ISO/IEC 9594-8). Сертификат содержит элементы данных, определенные в приведенной ниже табл. 2, сопровождаемые цифровой подписью.

Цифровая подпись для всех элементов вырабатывается при помощи ключа «authority key identifier». Элементы, включенные в версию v1, являются обязательными; элементы, включенные в последующие версии — необязательные. Имя субъекта сначала было обязательным элементом и должно было быть уникальным. Начиная с версии v3, оно стало необязательным. В формат уникального имени включается такая информация, как название страны, региона и данного имени, которые объединяются так, чтобы образовать уникальный идентификатор. Использование одинаковых идентификаторов запрещено.

Таблица 2

Формат сертификата X.509

Версия	Элемент	Описание
v1	version	Версия (0 означает v1, 2 означает v3)
	serialNumber	Серийный номер сертификата
	signature.algorithmIdentifier algorithm parameters	Тип алгоритма подписи
	issuer	Уникальное название центра, выпускающего сертификат
	Validity	Период действия
	NotBefore	Дата и время начала действия
	notAfter	Дата и время конца действия
	subject	Уникальное имя субъекта
	SubjectPublicKeyInfo	Информация об открытом ключе субъекта
	Algorithm	Криптографический алгоритм
	subjectPubkicKey	Ключ (строка битов)
v2	issuerUniqueID	Уникальный идентификатор центра, выпускающего сертификат
	subjectUniqueID	Уникальный идентификатор субъекта
v3	AuthorityKeyIdentifier	Идентификатор ключа, используемого для подтверждения подписи CA
	keyIdentifier	Идентификатор ключа
	authorityCertIssuer	Общее название CA
	authorityCertSerialNumber	Серийный номер сертификата CA

Продолжение таблицы 2

Версия	Элемент	Описание
	subjectKeyIdentifier	Идентификатор, используемый тогда, когда субъект имеет более одного ключа (например, во время возобновления сертификата)
	keyUsage	Применение ключа (строки битов) Цифровая подпись Невозможность отказа получателя/отправителя сообщения от факта его передачи/приема и содержания Шифрование ключа Шифрование информации Соглашение о ключе Подписание сертификата Подписание CRL
	privateKeyUsagePeriod	Период действия секретного ключа CA для подписи. Стандартно он короче периода действия соответствующего открытого ключа
	CertificatePolicies	Политика CA
	policyIdentifier	Идентификатор политики (как для ISO/IEC 9834-1)
	PolicyQualifiers	Критерии сертификации
	PolicyMappings	Используется только для сертификата CA
	IssuerDomainPolicy SubjectDomainPolicy	Оговаривает, что политика эмитента и политика сертификации субъекта одинаковы
	SupportedAlgorithms AlgorithmIdentifier IntendedUsage intendedCertificatePolicies	Определяют атрибуты каталога. Используются, чтобы сделать атрибуты известными заранее в случаях, когда партнер по связи использует данные каталога
	SubjectAltName	Альтернативное имя субъекта. Свободный выбор имени.
	OtherName	Произвольное имя
	rfc822Name	Адрес электронной почты
	dNSName	Имя домена
	x400Address	Адрес X.400 отправителя/получателя
	directoryName	Имя каталога EDI-имя
	ediPartyName	Унифицированный указатель ресурсов WWW
	uniformResourceIdentifier	URL
	iPAddress registeredID	IP-адрес объекта
	issuerAltName	Альтернативное имя CA
	subjectDirectoryAttributes	Необязательные атрибуты субъекта, например, почтовый адрес, номер телефона и т. п.
	BasicConstraints	Отличает ключ CA от ключей конечных пользователей (используется только для сертификата CA)
	cA	Для ключа CA cA истинно
	pathLenConstraint	Ограничение длины пути

Продолжение таблицы 2		
Версия	Элемент	Описание
	NameConstraints	Используется только при сертификации CA. Определяет сертификацию домена по имени по отношению к CA более низкого уровня в пределах пути, устанавливаемого параметром BasicConstraints
	PermittedSubtrees	CA более низкого уровня и домен его поддерева
	Base	Имя CA более низкого уровня
	minimum	Верхний предел домена
	maximum	Нижний предел домена
	excludedSubtree	CA более низкого уровня, исключенный из домена
	PolicyConstraints PolicySet InhibitPolicyMapping	Ограничения политики (используется только для requireExplicitPolicy CA)
	cRLDistributionPoints	Пункты распределения CRL
	distributionPoint	Имя центра распределения Abbreviates cRLIssuer
	reasons	Вид списка, распределяемого данным пунктом
	keyCompromise	Скомпрометированный ключ конечного пользователя
	cACompromise	Скомпрометированный ключ CA
	affiliationChanged	Измененная информация в сертификате (не повреждение)
	superseded	Приостановленный ключ
	cessationOfOperation	Завершение использования
	certificateHold	Приостановление использования
	cRLIssuer	Имя центра-эмитента CRL

3.7.1. Описание полей

— Версия

Данное поле описывает версию сертификата. При использовании дополнений версия должна быть установлена как X.509 version 3 (значение равно 2). Если дополнения не используются, версия должна быть 1 (значение должно быть не установлено).

— Идентификатор алгоритма ЭЦП

Поле содержит идентификатор криптографического алгоритма, используемого CA для выработки ЭЦП сертификата.

— Серийный номер сертификата

Серийный номер является целым числом, устанавливаемым CA для каждого сертификата. Значение должно быть уникальным для каждого сертификата, выпущенного данным CA. Имя Издателя и серийный номер сертификата совместно являются уникальным идентификатором сертификата.

— Имя Издателя сертификата

Поле Издатель идентифицирует объект (субъект), который сформировал ЭЦП и издал сертификат. Значение в поле Издатель должно содержать ненулевое значение DN (*distinguished name*). Поле Издатель определено в рекомендациях *X*.501 как тип *Name*. Значение поля состоит из набора иерархических атрибутов (*AttributeType*), таких как код страны и соответствующего ему значения (*AttributeValue*, например, RU). Тип компонентов *AttributeValue*, входящих в имя, определяется типом атрибута *AttributeType* и в основном используется *DirectoryString*.

— Срок действия сертификата

Данное поле определяет срок действия (в виде временного интервала), в течение которого CA управляет сертификатом (отслеживает состояние). Поле представляет последовательность двух дат: дата начала действия сертификата (*notBefore*) и дата окончания срока действия сертификата (*notAfter*). Оба этих значения могут быть закодированы либо как *UTCTime*, либо как *GeneralizedTime*.

— Имя Владельца сертификата

Поле Владелец идентифицирует объект (субъект), являющийся обладателем секретного ключа, соответствующего открытому ключу в сертификате.

— Открытый ключ Владельца

Данное поле используется для хранения открытого ключа и идентификации алгоритма, соответствующего открытому ключу. Поле *parameters* идентификатора может содержать дополнительные данные, присущие определенному алгоритму.

— Уникальный идентификатор Издателя и Владельца

Данное поле может использоваться только в сертификатах версии 2 или 3. Поле было предусмотрено в версии 2 сертификатов X.509 для целей обеспечения использования одинакового имени Владельца или Издателя в разных сертификатах. С введением дополнений в версии 3 такая необходимость отпала.

— Подпись Центра Сертификации

Битовая последовательность, содержащая значение ЭЦП, сформированной с использованием секретного ключа Центра Сертификации и алгоритмов хеширования и ЭЦП, указанных в поле Идентификатор алгоритма ЭЦП.

3.7.2. Сертификат X.509 версии 3

Данный раздел описывает версию 3 сертификата X.509. Версия 3 описала механизм дополнений, дополнительной информации, которая может быть помещена в сертификат. X.509 и рекомендации RFC 2459 описывают набор *стандартных* дополнений, но вместе с тем не ограничивают возможности использования любых других дополнений путем регистрации идентификатора (ISO или IANA).

Каждое дополнение состоит из трех полей:

— tipe,
— critical,
— value.

Дополнение представляет собой структуру, содержащую:

— идентификатор дополнения,
— признак «критичное / не критичное» дополнение,
— собственно значение дополнения, представленное в бинарном виде (OCTET STRING).

В свою очередь само дополнение может являться какой угодно сложной структурой (от простого текстового значения до сложной структуры), формат и интерпретация которого определяются идентификатором дополнения.

Основная цель критичных дополнений – предохранить сертификат, изданный CA, от возможности использования его в приложениях, которые не могут обработать такие дополнения. Таком образом, правила обработки дополнений, изложенные в рекомендациях, требуют от прикладного ПО отвергнуть сертификат, если

дополнение отмечено критичным и прикладное ПО не может его интерпретировать. В свою очередь, требование отвергнуть дополнение прикладным ПО, отмеченное как критичное, при невозможности его интерпретации, требует от прикладного ПО детального разбора дополнений сертификатов и затрудняет процесс модификации как прикладного ПО, так и ПО, обеспечивающего реализацию PKI.

Дополнения X.509 версии 3 Дополнения, используемые в сертификатах версии 3, определены рекомендациями X.509 версии 3 ITU-T и рекомендациями IETF RFC 2459.

Базовый идентификатор дополнений, определенный рекомендациями X.509: id-ce OBJECT IDENTIFIER ::= {joint-iso-ccitt(2) ds(5) 29}, где id-ce обозначает: Object identifier assignments for ISO certificate extensions.

Все дополнения, определенные указанными рекомендациями, можно разделить на две категории: ограничивающие и информационные дополнения. Первые ограничивают область применения ключа, определенного сертификатом, или самого сертификата. Вторые содержат дополнительную информацию, которая может быть использована в прикладном ПО пользователем сертификата.

К ограничивающим дополнениям относятся:

— базовые ограничения (*basicConstraints*);
— область применения ключа (*keyUsage*);
— расширенная область применения ключа (*extendedKeyUsage*);
— регламенты сертификата (модифицируемые ограничениями регламентов и соответствием регламентов) (*certificatesPolicies*);
— ограничения имен (*nameConstraints*).

К информационным дополнениям относятся:

— идентификаторы ключей (*subjectKeyIdentifier, authorityKeyIdentifier*);
— альтернативные имена (*subjectAltName, issuerAltName*);
— точка распространения СОС (*cRLDistributionPoint, issuingDistributionPoint*);
— способ доступа к информации CA (*authorityAccessInfo*).

Вместе с этим стандарт X.509 позволяет определить любые другие дополнения, необходимость которых определяется их использованием в конкретной системе (например: протокол SET).

Идентификатор ключа Издателя Дополнение Идентификатор ключа Издателя (*authorityKeyIdentifier*) используется для идентификации открытого ключа, соответствующего секретному ключу ЭЦП, использованному CA при подписи издаваемого сертификата (или СОС). Данное дополнение может быть использовано в случае, когда Издатель сертификата (CA) имеет несколько различных секретных ключей ЭЦП (например при плановой их смене), а также для однозначного построения цепочек сертификатов.

Идентификатор ключа Владельца Данное дополнение используется для идентификации различных сертификатов, содержащих открытый ключ. Для упрощения процедуры построения цепочки данное дополнение должно устанавливаться во всех сертификатах CA, которые включают дополнение *basicConstraints* с установленным значением *cA TRUE*. Во всех издаваемых CA сертификатах значение *keyIdentifier* в дополнении *authorityKeyIdentifier* должно быть идентично значению *subjectKeyIdentifier* сертификата CA.

Для сертификатов значение *subjectKeyIdentifier* должно вырабатываться из открытого ключа или с использованием метода генерации уникальных значений. Рекомендациями RFC 2459 предлагается два метода генерации идентификатора на основе значения открытого ключа:

1. Значение *keyIdentifier* определяется как 160 бит хэш-функции, вычисляемой по алгоритму SHA-1 из значения BIT STRING subjectPublicKey (исключая тэг, длину и неиспользованные биты).

2. Значение *keyIdentifier* определяется как 4-х битовое поле со значением 0100 и последующим за ним 60 битами наименьшей значимой части хэш-функции, вычисляемой по алгоритму SHA-1 из значения BIT STRING subjectPublicKey.

Для идентификации без использования открытого ключа можно также использовать монотонно возрастающую последовательность целых чисел.

Для сертификатов конечного пользователя данное дополнение используется для идентификации приложением различных сертификатов, содержащих определенный открытый ключ. Если конечный пользователь обладает несколькими сертификатами, особенно от разных CA, данное дополнение позволяет быстро определить требуемый сертификат. Для этих целей данное дополнение должно добавлять во все сертификаты конечных пользователей.

Значение данного дополнения для сертификатов конечных пользователей должно формироваться из значения открытого ключа способом, описанным выше.

Область применения ключа Данное дополнение определяет область применения секретного ключа, соответствующего открытому, содержащемуся в сертификате.

Таблица 3

Область применения ключа

Флаг	Применение ключа
digitalSignature	Ключ может быть использован для целей обеспечения целостности и авторства информации. Формирование и проверка ЭЦП электронных документов и информации, установление идентичности в процессе аутентификации и т.д.
nonRepudiation	Ключ может быть использован в приложениях, обеспечивающих неотрекаемость.
keyEncipherment	Ключ может быть использован для шифрования других ключей.
dataEncipherment	Ключ может быть использован для целей обеспечения конфидециальности и целостности информации.
keyAgreement	Ключ может быть использован в процессе формирования других ключей (например, по алгоритму Деффи-Хелмана).
keyCertSign	Ключ может быть использован для целей формирования ЭЦП сертификатов.
CRLSign	Ключ может быть использован для целей формирования ЭЦП CRL.
EncipherOnly	Ключ может быть использован только для шифрования.
DecipherOnly	Ключ может быть использован только для расшифрования.

Расширенная область применения ключа Данное дополнение *(Extended keyUsage)* предназначено для задания дополнительных областей применения ключа по требованиям прикладного ПО.

Значение *Область применения ключа* (*KeyPurposeId*) данного дополнения может быть определена любой организацией в зависимости от конкретных целей.

Идентификатор объекта для идентификации области применения должен назначаться в соответствии с IANA или ITU-T Rec. X.660 | ISO/IEC/ITU 9834-1.

Срок действия секретного ключа Данное дополнение позволяет Издателю сертификата задать различные сроки действия секретного ключа и сертификата. Дополнение содержит два опциональных компонента: *notBefore и notAfter*. Секретный ключ, соответствующий открытому в сертификате, не должен быть использован до или после времен, указанных соответствующими компонентами. CA не должен формировать сертификат, в котором не указан ни один из компонентов.

Регламенты использования сертификата Данное дополнение представляет собой последовательность, состоящую из одного или нескольких *информаторов регламента (PolicyInformation*), каждый из которых содержит *идентификатор объекта* (OID) и опциональный *квалификатор*.

Данный *информатор регламента* отображает регламент, с учетом которого сертификат был издан, и цели, для которых сертификат может быть использован. Опциональный *квалификатор*, который может присутствовать, не предусмотрен для целей изменения регламента, определенного информатором.

Приложения с определенными требованиями функционирования, должны содержать внутренний список регламентов, удовлетворяющих данным требованиям, для целей сравнения *идентификаторов объектов* в сертификате с имеющимся внутренним списком приложения. Если данное дополнение критичное, ПО, производящее обработку, должно обладать возможностью интерпретации данного дополнения (включая опциональный *квалификатор*). В противном случае сертификат должен быть отвергнут.

Соответствие регламентов Данное дополнение предназначено для использования в сертификатах CA. Оно содержит список парных *Идентификаторов Объектов* (OID). Каждая пара в свою очередь включает *Регламент Зоны Издателя* (*issuerDomainPolicy*) и *Регламент Зоны Владельца* (*subjectDomainPolicy*). Такая парность означает, что CA, выступающий в роли Издателя (*issuing CA*), принимает *Регламент Зоны Издателя* эквивалентным *Регламенту Зоны Владельца* для подчиненного CA (*subject CA*).

Пользователи, относящиеся к Издающему CA (*issuing CA*), могут принимать *Регламент Зоны Издателя*(*issuerDomainPolicy*) для соответствующих приложений. Дополнение *Соответствие Регламентов* ставит в известность пользователей Издающего CA о том наборе *Регламентов*, (*subject CA*), которые сравнимы с регламентами, соответствующими их требованиям.

Альтернативное имя Владельца Дополнение *Альтернативное Имя Владельца* может использоваться для двух целей. Во-первых, оно позволяет расширить границы идентификации Владельца сертификата. Для этого используются заранее определенные идентификаторы, которые включают адрес электронной почты, имя в DNS, IP адрес и URI. Во-вторых, оно предоставляет набор дополнительной справочной информации о Владельце сертификата. Для этого используется представление имени в различных видах и множественное представление имен. При необходимости введения такой дополнительной идентификации в сертификат должно использоваться дополнение *Альтернативное Имя Владельца* или *Альтернативное Имя Издателя*.

В связи с тем, что альтернативное имя может быть использовано для целей определения соответствия Владельца и открытого ключа, все части, идентифицирующие его и входящие в альтернативное имя, должны быть проверены CA. Уровень проверки дополнительной информации определяется Регламентом CA.

Альтернативное Имя Владельца может быть ограничено тем же способом, что и поле *Владелец* в сертификате, используя дополнение *nameConstraintsExtension*.

Кроме зарегистрированных типов полей, Издающий Центр может определить и использовать свои собственные типы имен, определив их в поле *otherName*.

В соответствии с рекомендациями X.681 синтаксис otherName поля определен в следующем виде:

Таблица 4

Поля дополнения «Альтернативное Имя»

Наименование	Тип	Назначение	Идентификатор
otherName	ANY DEFINED BY type ID	Любое, определяемое Издателем	CHOICE[0]
rfc822Name	IA5String	Адрес электронной почты rfc822 [6]	CHOICE[1]
DNSName	A5String	Имя DNS	CHOICE[2]
directoryName	IA5String	X.500 DN имя	CHOICE[4]
uniformResource identifier	IA5String	Адрес URI	CHOICE[6]
ipAddress	OCTET STRING	Адрес IP	CHOICE[7]
registeredID	OBJECT IDENTIFIER	Идентификатор ASN.1 объекта	CHOICE[8]

```
TYPE-IDENTIFIER ::= CLASS
{
&id OBJECT IDENTIFIER UNIQUE,
&Type
}
WITH SYNTAX {&Type IDENTIFIED BY &id}
```

Альтернативное имя Издателя Так же как и дополнение *Альтернативное Имя Владельца*, дополнение *Альтернативное имя Издателя* (*issuerAltName*) служит целям дополнительной ассоциации *Издателя* сертификата. Правила использования данного дополнения аналогичны использованию дополнения *Альтернативное Имя Владельца*.

Атрибуты Справочника Владельца сертификата Дополнение предусмотрено для хранения дополнительной информации, связанной с атрибутами директории X.500. Дополнение *Атрибуты Справочника Владельца сертификата* не рекомендуется использовать рекомендациями RFC 2459, но он может быть использован в частных реализациях.

Основные ограничения Дополнение *Базовые ограничения* идентифицирует, является ли Владелец сертификата Центром Сертификации, и какова длина цепочки сертификатов для этого CA.

Поле *pathLenConstraint* имеет смысл только при условии, если значение *cA* установлено в *TRUE*. В этом случае оно обозначает максимальное количество сертификатов CA, которые следуют за данным сертификатом в цепочке. Значение ноль означает, что только сертификаты конечного пользователя могут следовать в цепочке за данным сертификатом. При использовании значение *pathLenConstraint* больше или равно нулю. Если значение не установлено, это означает, что лимит на длину цепочки не определен.

Ограничения имени Дополнение *Ограничение имени* должно использоваться только в сертификатах CA. Оно указывает пространство имен, внутри которого должны быть расположены все имена Владельцев издаваемых сертификатов. Ограничения могут быть применимы как к имени Владельца (*Subject DN*), так и к альтернативному имени.

Ограничения определены в терминах допускаемого (*permittedSubtrees*) или исключаемого (*excludedSubtrees*) поддерева имен. Любое имя, совпадающее с ограничением в исключающем поддереве, является некорректным в независимости от возможного его присутствия в допускаемом поддереве.

При реализации данного дополнения RFC 2459 рекомендуется:

— не использовать поля *minimum* и *maximum* ни в какой из форм имен, так что *minimum* всегда ноль, а *maximum* всегда отсутствует;
— использовать только поля *permittedSubtrees* для задания разрешенного диапазона имен и не использовать *excludedSubtrees*, что согласуется с организационной или территориальной схемой иерархии.

Ограничение регламента Данное дополнение может быть использовано в сертификатах, издаваемых для CA. Дополнение *Ограничение регламента* накладывает ограничения на проверяемую цепочку в двух направлениях. Оно может использоваться для запрещения проверки соответствия регламентов (*policy mapping*) или требовать, чтобы каждый сертификат в цепочке содержал приемлемый идентификатор регламента (*policy identifier*).

Точка распространения CRL Точка распространения CRL является дополнением, которое определяет механизм и расположение CRL определенного типа в сети, и определяет зону действия CRL как:

— только для конечных пользователей;
— только для CA;
— или ограничивает коды мотивации.

Коды мотивировки, ассоциированные с точкой распространения, должны специфицироваться в поле *onlySomeReasons*. Если поле *onlySomeReasons* отсутствует, точка распространения должна содержать отзываемые сертификаты для всех кодов. CA может использовать Точку распространения CRL как основу для управления потоками данных при компрометации. В этом случае отзывы сертификатов с кодами *keyCompromise* и *cACompromise* располагаются в одной точке распространения, а все остальные — в другой.

Способ доступа к информации CA Данное дополнение определено в рекомендациях IETF RFC 2459 (в отличие от остальных стандартных дополнений, которые определены как в рекомендациях X.509, так и в RFC 2459).

Данное дополнение указывает на способы доступа к информации и сервисам CA, издавшим сертификат, в котором это дополнение установлено. Информация и сервис могут включать процедуры интерактивной проверки и получения Регламента (Регламентов) CA. Способ получения CRL не специфицируется данным дополнением, для этого используется дополнение *cRLDistributionPoints*.

3.8. Верификация цепочки сертификатов

Доверие любому сертификату пользователя определяется на основе цепочки сертификатов. Причем начальным элементом цепочки является сертификат CA, хранящийся в защищенном персональном справочнике пользователя.

Процедура верификации цепочки сертификатов описана в рекомендациях X.509 и RFC 2459 [7] и проверяет связанность между именем Владельца сертификата и его открытым ключом. Процедура верификации цепочки подразумевает, что все «правильные» цепочки начинаются с сертификатов, изданных одним CA. Под доверенным центром понимается главный CA, открытый ключ которого содержится в самоподписанном сертификате. Такое ограничение упрощает процедуру верификации, хотя наличие самоподписанного сертификата и его криптографическая проверка не обеспечивают безопасности. Для обеспечения доверия к открытому

ключу такого сертификата должны быть применены специальные способы его распространения и хранения, так как на данном открытом ключе проверяются все остальные сертификаты.

Алгоритм верификации цепочек использует следующие данные:

— X.500 имя Издателя сертификата;
— X.500 имя Владельца сертификата;
— открытый ключ Издателя;
— срок действия открытого (секретного) ключа Издателя и Владельца;
— ограничивающие дополнения, используемые при верификации цепочек;
— CRL для каждого Издателя (даже если он не содержит отзываемых сертификатов).

Цепочка сертификатов представляет собой последовательность из n сертификатов, в которой:

— для всех x в $\{1, (n-1)\}$, Владелец сертификата x есть Издатель сертификата $x+1$;
— сертификат $x = 1$ есть самоподписанный сертификат;
— сертификат $x = n$ есть сертификат конечного пользователя.

Одновременно с цепочкой сертификатов используется цепочка CRL, представляющая собой последовательность из n CRL, в которой:

— для всех CRL x в $\{1, n\}$, Издатель сертификата x есть Издатель CRL x;
— CRL $x = 1$ есть CRL, изданный Владельцем самоподписанного сертификата;
— CRL $x = n$ есть CRL, изданный Издателем сертификата конечного пользователя.

После построения двух цепочек (сертификатов и CRL) выполняется:

— криптографическая проверка сертификатов и CRL в цепочках;
— проверка сроков действия сертификатов и CRL;
— проверка соответствия имен Издателя и Владельца с использованием дополнения *nameConstraints*;
— проверка длины цепочки с использованием дополнения *basicConstraints*;
— проверка на отзыв сертификатов, причем, если сертификат промежуточного центра был отозван CRL вышестоящего центра, все сертификаты, изданные промежуточным центром, считаются недействительными;
— проверка приемлемых регламентов использования сертификата и приемлемых областей использования ключа с использованием дополнений *certificatesPolicies* и *extendedKeyUsage*.

3.9. Стандарты в области PKI

Стандарты в области PKI делятся на две группы: часть из них описывает собственно реализацию PKI, а вторая часть, которая относится к пользовательскому уровню, использует PKI, не определяя ее.

Стандартизация в области PKI позволяет различным приложениям взаимодействовать между собой с использованием единой PKI.

В особенности стандартизация важна в области:

— процедуры регистрации и выработки ключа;
— описания формата сертификата;
— описания формата CRL;
— описания формата криптографически защищенных данных;
— описания протоколов обмена информацией.

Основным центром по выпуску согласованных стандартов в области PKI является рабочая группа PKI (PKI working group) организации IETF (Internet Engineering Task Force), известная как группа PKIX (от сокращения PKI for X.509 certificates).

3.9.1. Стандарты PKIX

Спецификации PKIX основаны на двух группах стандартов: X.509 ITU-T (Международный комитет по телекоммуникациям) и PKCS (Public Key Cryptography Standards) firmy RSA Data Security. X.509 изначально был предназначен для спецификации аутентификации при использовании в составе сервиса X.500 директории. Фактически же, синтаксис электронного сертификата, предложенный в X.509, признан стандартом де-факто и получил всеобщее распространение независимо от X.500. Однако X.509 ITU-T не был предназначен для полного определения PKI. В целях применения стандартов X.509 в повседневной практике пользователи, поставщики и комитеты по стандартизации обращаются к стандартам PKCS.

PKIX группа издала следующие стандарты Internet (RFC):

— Internet X.509 Public Key Infrastructure Certificate and CRL Profile (RFC 2459) [7];
— Internet X.509 Public Key Infrastructure Certificate Management Protocols (RFC 2510) [8];
— Internet X.509 Certificate Request Message Format (RFC 2511) [9];
— Internet X.509 Public Key Infrastructure Certificate Policy and Certification Practices Framework (RFC 2527) [10];
— Internet X.509 Public Key Infrastructure Representation of Key Exchange Algorithm (KEA) Keys in Internet X.509 Public Key Infrastructure Certificates (RFC 2528) [11];
— Internet X.509 Public Key Infrastructure Operational Protocols - LDAPv2 (RFC 2559) [12];
— Internet X.509 Public Key Infrastructure Operational Protocols: FTP and HTTP (RFC 2585) [13];
— Internet X.509 Public Key Infrastructure LDAPv2 Schema (RFC 2587) [14];
— X.509 Internet Public Key Infrastructure Online Certificate Status Protocol — OCSP (RFC 2560) [15].

Стандарт X.509 ITU-T является фундаментальным стандартом, лежащим в основе всех остальных, используемых в PKI. Основное его назначение — определение формата электронного сертификата и CRL.

3.9.2. PKCS

Из серии стандартов, изданных фирмой RSA Data Security, наиболее важными и используемыми в PKI являются:

— PKCS #7 Cryptographic Message Syntax Standard;
— PKCS #10 Certificate Request Syntax Standard;
— PKCS #12 Personal Information Exchange Syntax Standard.

Вместо устаревшего стандарта RSA PKCS #7, описывающего форматы криптографических сообщений, в июне 1999 года был принят RFC 2630 «Cryptographic Message Syntax».

3.9.3. Стандарты, основанные на PKI

Большинство стандартов, использующих криптографию, разработано с учетом использования PKI.

— S/MIME

  Стандарт S/MIME определен IETF для обеспечения защищенного обмена сообщениями. S/MIME использует PKI для формирования ЭЦП и шифрования информации. В группе стандартов S/MIME наиболее важными являются следующие: Cryptographic Message Syntax, Message Specification, Certificate Handling и Certificate Request Syntax.

— SSL и TLS

Протокол SSL (разработанный фирмой Netscape) и соответствующий ему стандарт IETF TLS (RFC 2246) являются наиболее используемыми стандартами для обеспечения защищенного доступа к Web. Вместе с этим, SSL и TLS широко используются для создания клиент-серверных приложений, не использующих Web. Оба эти протокола в своей основе используют PKI.

— Secure Electronic Transactions (SET)

Протокол SET был разработан фирмами Visa и MasterCard и предназначен для обеспечения системы электронных банковских расчетов с использованием пластиковых карт. В данном протоколе PKI является фундаментом, на котором базируется вся система аутентификации участников расчетов.

— IPSec

Протокол IPSEC (Internet Protocol Security Protocol) разработан IETF как протокол для шифрования IP и является одним из основных протоколов, используемых для построения VPN. PKI в протоколе IPSec используется для аутентификации и шифрования. В настоящее время протокол еще широко не распространен, но повсеместное развитие PKI приводит к возрастанию количества реализаций IPSec.

3.10. Создание собственного центра сертификации

Если компания реализует собственный CA [16], она должна создать также *Сертификационную политику — Certificate Policy* и *Положение об использовании сертификата — Certificate Practice Statement*.

Сертификационная политика очерчивает требования, необходимые в процессе аутентификации для получения сертификата от CA, а также может определять уровень полномочий (например, «этот сертификат имеет право подписи информации по сделкам, не превышающим один миллион долларов»). В случае конечной точки IPSec сертификационная политика определяет, какая информация должна быть предъявлена CA для аутентификации до выпуска сертификата для этой конечной точки. Он также определяет, какую информацию будет содержать индивидуальный сертификат и как он будет выглядеть. Сертификационная политика определяет обновление CRL или требования размещения уведомления об аннулированном сертификате на сервере OCSP (Online Certificate Status Protocol). Кроме того, сертификационная политика может также определять требования физической безопасности, которым должен соответствовать CA.

Положение об использовании сертификата представляет собой документ, описывающий подробности реализации сертификационной политики и объясняющий порядок соотнесения CA с требованиями сертификационной политики. По правилам American Bar Association, идентификатор Положения об использовании сертификата Объектный идентификатор (Object Identifier—OID) также должен включаться в сертификат. OID представляет собой представление компании в численной форме. Эта информация в дальнейшем позволит RA проверять квалификацию CA.

Лучшим справочным материалом при написании сертификационной политики и положения об использовании сертификата является RFC 2527 (Internet X.509 Public Key Infrastructure Certificate Policy and Certification Practices Framework), увидевшая свет в марте 1999 года. Прежде чем публиковать сертификационную политику и положение об использовании сертификата, необходимо также проконсультироваться с юристами компании.

Хотя разработка сертификационной политики и положения об использовании сертификата может показаться ненужной, так как компания, использующая CA, полностью контролирует их реализацию, тем не менее, это необходимо по двум причинам. Во-первых, это гарантирует оптимальный уровень безопасности, поскольку компания создает документы и по эксплуатационным вопросам, и по проверке безопасности. Во-вторых, это необходимо в случае, когда компания планирует провести взаимную сертификацию с другой PKI.

Взаимная сертификация означает, что сертификат, выпущенный одной PKI, становится применим в другой PKI. И сертификационная политика, и положение об использовании сертификата каждой из PKI будут проверены другой PKI с целью

убедиться, что их сертификаты можно рассматривать как равнозначные относительно важных для них аспектов (не совсем верно было бы называть их полностью «равнозначными», так как юридически это означало бы и словесное совпадение). Взаимная сертификация представляет собой простое физическое действие: каждый CA выпускает для другого сертификат, содержащий *Расширенное отображение политики — Policy Mappings Extension*, закрепляющий вышеуказанное соглашение. Сложность состоит в согласовании соглашений сертификационной политики и положения об использовании сертификата между двумя PKI.

Взаимная сертификация может быть обременительна, когда существует множество PKI, так как в этом случае число взаимных сертификатов растет в геометрической прогрессии. Если имеются две PKI, будет два сертификата, но если PKI будет четыре, им придется обменяться уже двенадцатью сертификатами. В какой-то момент EE — утратит способность проходить эту цепь для выяснения допустимости возможно легитимного предъявленного ему сертификата, т. е. реализация этого EE не имеет вычислительных возможностей для прохождения цепи такой протяженности. В данном случае решением будет *шунтированная PKI (Bridged PKI)*.

Отдельный шунтированный CA обменивается взаимными сертификатами со всеми корневыми CA имеющихся PKI. Это уменьшает длину цепочки, которую необходимо пройти EE, чтобы провести аутентификацию сертификата, предъявленного ей извне ее PKI. Это основная причина реализации шунтиромванной PKI, однако такая технология выгодна потому, что она упрощает аннулирование взаимной сертификации. Вместо размещения большого количества сертификатов необходимо будет разместить только один — тот, что выпущен шунтированным CA для взаимной сертификации.

4. Практическое развёртывание PKI

4.1. Программа OpenSSL

4.1.1. Формат командной строки

Для работы с ключами и сертификатами используется программа OpenSSL. Программа имеет следующий формат командной строки:

```
# openssl команда [ опции ] [ аргументы ]
```

Для работы с сертификатами интересны следующие возможности:

- создание и управление ключами RSA и DSA — команды `rsa`, `dsa`, `dsaparam`, `gendsa`, `genrsa`;
- работа с сертификатами X.509 — команды `x509`, `req`, `verify`, `ca`, `crl`, `pks12`, `pks7`, `crl2pkcs7`;
- работа с S/MIME — команда `smime`.

Основные опции команды следующие:

`-config файл` — задаёт конфигурационный файл;
`-in файл` — задаёт входной файл (в варианте `-infiles список_файлов` возможна обработка нескольких файлов);
`-out файл` — задаёт выходной файл;
`-days количество_дней` — время действия сертификата.

4.1.2. Соглашения об именах каталогов

В качестве корневого каталога выбирается `/etc/ssl`, а кофигурационым файлом будет файл `/etc/ssl/openssl.cnf`. Тогда структура каталога `/etc/ssl` примет следующий вид:

`/etc/ssl/certs` — каталог, в который помещаются сертификаты;
`/etc/ssl/crl` — каталог, содержащий список аннулированных сертификатов;
`/etc/ssl/newcerts` — временный каталог для создаваемых сертификатов;
`/etc/ssl/keys` — каталог для секретных ключей сертификатов;
`/etc/ssl/req` — каталог для файлов запросов на получение сертификатов;

`/etc/ssl/private` — каталог для секретных данных (должен иметь соответствующие права доступа). В нём находятся следующие файлы:

`/etc/ssl/private/cakey.pem` — секретный ключ CA;

`/etc/ssl/private/.rand` — файл со случайными данными. Логичнее, казалось-бы, использовать стандартные файлы `/dev/[u]random`. Но к сожалению некоторые версии OpenSSL некорректно работают с этими файлами. Поэтому проще создать файл `/etc/ssl/private/.rand` вручную.

`/etc/ssl/cacert.pem` — сертификат CA;

`/etc/ssl/index.txt` — файл, содержащий список сертификатов. Записи маркируются следующими метками:

V — действительные сертификаты (Valid);

E — сертификаты с истёкшим сроком действия (Expired);

R — аннулированные сертификаты (Revoked).

Первоначально этот файл пуст;

`/etc/ssl/serial` — файл, содержащий порядковый номер сертификата (в шестнадцатиричном формате). Первоначально можно выставить в «01».

4.1.3. Конфигурационный файл

Возможный вариант конфигурационного файла `/etc/ssl/openssl.cnf`:

```
# {{{ Умолчания

[ ca ]
# Секция по умолчанию
default_ca                      = ca_default

[ ca_default ]
# Корневой каталог
dir                             = /etc/ssl
# Сертификаты (каталог)
certs                           = $dir/certs
# Списки аннулированных сертификатов (каталог)
crl_dir                         = $dir/crl
# База данных сертификатов
database                        = $dir/index.txt
# Временный каталог для создаваемых сертификатов
new_certs_dir                   = $dir/newcerts
# CA сертификат
certificate                     = $dir/cacert.pem
# Текущий порядковый номер сертификата
serial                          = $dir/serial
# Текущий CRL
crl                             = $dir/crl/crl.pem
# Секретный ключ CA
private_key                     = $dir/private/cakey.pem
# Случайные данные
RANDFILE                        = $dir/private/.rand
# Расширение по умолчанию для новых сертификатов
x509_extensions                 = server_cert
# Время действия сертификатов
default_days                    = 365
# Время действия CRL
default_crl_days                = 30
# Используемая хэш-функция
default_md                      = md5
# Сохранять порядок DN как в запросе
# Иначе, как в политике
preserve                        = no
# Политика по умолчанию
policy                          = policy_match

# }}}
```

```
# {{{ Политики

# Политика для CA
[ policy_match ]
countryName                     = match
stateOrProvinceName             = match
organizationName                = match
organizationalUnitName          = optional
commonName                      = supplied
emailAddress                    = optional

# Политика для <<всего остального>>
[ policy_anything ]
countryName                     = optional
stateOrProvinceName             = optional
localityName                    = optional
organizationName                = optional
organizationalUnitName          = optional
commonName                      = supplied
emailAddress                    = optional

# }}}
# {{{ Запросы

[ req ]
# Длина ключа по умолчанию
default_bits                    = 4096
default_keyfile                 = privkey.pem
distinguished_name              = req_distinguished_name
attributes                      = req_attributes
x509_extensions                 = v3_ca
string_mask                     = nombstr
req_extensions                  = v3_req

[ req_distinguished_name ]
countryName                     = Country Name (2 letter code)
countryName_default             = RU
countryName_min                 = 2
countryName_max                 = 2
stateOrProvinceName             = State or Province Name (full name)
stateOrProvinceName_default     = Russia
localityName                    = Locality Name (eg, city)
localityName_default            = Moscow
0.organizationName              = Organization Name (eg, company)
0.organizationName_default      = Peoples Friendship University
organizationalUnitName          = Organizational Unit Name (eg, section)
organizationalUnitName_default  = faculty
commonName                      = Common Name (eg, YOUR name)
commonName_max                  = 64
emailAddress                    = Email Address
emailAddress_default            = cert@sci.pfu.edu.ru
emailAddress_max                = 40

[ v3_req ]
# По умолчанию запрос не для сертификата CA
basicConstraints                = CA:FALSE
keyUsage        = nonRepudiation, digitalSignature, keyEncipherment

[ req_attributes ]
challengePassword               = A challenge password
challengePassword_min           = 4
challengePassword_max           = 20
unstructuredName                = An optional company name

# }}}
# {{{ CA
```

```
109
110 [ v3_ca ]
111 subjectKeyIdentifier           = hash
112 authorityKeyIdentifier         = keyid:always,issuer:always
113 # CA:true для CA сертификата
114 basicConstraints               = critical,CA:true
115 keyUsage                       = cRLSign, keyCertSign
116 # Область применения сертификата CA
117 nsCertType                     = sslCA, emailCA
118 subjectAltName                 = email:copy
119 issuerAltName                  = issuer:copy
120 nsComment                      = "OpenSSL Generated Certificate"
121
122 # }}}
123 # {{{ CRL
124
125 [ crl_ext ]
126 authorityKeyIdentifier         = keyid:always,issuer:always
127
128 # }}}
129 # {{{ Серверные сертификаты
130
131 [ server_cert ]
132 # CA:FALSE для не CA сертификатов
133 basicConstraints               = CA:FALSE
134 # Область применения сертификата
135 nsCertType                     = server
136 nsComment                      = "OpenSSL Generated Certificate"
137 subjectKeyIdentifier           = hash
138 authorityKeyIdentifier         = keyid,issuer:always
139 issuerAltName                  = issuer:copy
140 subjectAltName                 = email:copy
141
142 # }}}
143 # {{{ Клиентские сертификаты
144
145 [ client_cert ]
146 # CA:FALSE для не CA сертификатов
147 basicConstraints               = CA:FALSE
148 # Область применения сертификата
149 nsCertType                     = client, email, objsign
150 nsComment                      = "OpenSSL Generated Certificate"
151 subjectKeyIdentifier           = hash
152 authorityKeyIdentifier         = keyid,issuer:always
153 issuerAltName                  = issuer:copy
154 subjectAltName                 = email:copy
155 keyUsage                       = digitalSignature, keyEncipherment
156
157 # }}}
```

При работе значения параметров будут считываться в интерактивном режиме (из командной строки). При этом можно задать значения параметров по умолчанию, а также граничные значения параметров. Значения же самих параметров будут являться подсказками при вводе данных:

```
параметр                        = подсказка
параметр_default                = значение по умолчанию
параметр_min                    = минимальное значение
параметр_max                    = максимальное значение
```

Для отмены интерактивного режима следует задать параметр:

```
prompt                          = no
```

4.2. Форматы файлов сертификатов

Используюся следующие основные форматы файлов.

Формат DER Файлы в этом формате имеют расширение `.der`, являются бинарными файлами. Содержат информацию в кодировке DER.

Формат PEM Файлы в этом формате имеют расширение `.pem`, являются ascii-файлами. Они представляют собой информацию в кодировке DER, преобразованную по алгоритму BASE64. Может содержать несколько блоков информации, ограниченных разделителями, зависящими от типа содержания.

4.3. Создание CA

Рассматривается вариант развёртывания PKI с созданием собственного CA. Для этого необходимо выполнить следующие действия:

1. Создать секретный ключ CA;
   ```
   # openssl genrsa -out /etc/ssl/private/cakey.pem -des3 4096
   ```
   Длина ключа выставляется в 4096 бит. При генерации запрашивается пароль, на котором по алгоритму 3DES криптуется закрытый ключ.
2. Создать запрос на сертификацию;
   ```
   # openssl req -new -key /etc/ssl/private/cakey.pem \
       -config /etc/ssl/openssl.cnf -out /etc/ssl/req/careq.pem
   ```
   При генерации запроса следует указать CN для сертификата (например, как «NS faculty of PFUR root certificate»).
3. Далее возможно два варианта:
   — можно послать запрос на сертификацию в один из CA;
   — можно создать *самоподписанный (self-signed)* сертификат.
   ```
   # openssl x509 -req -signkey /etc/ssl/private/cakey.pem \
       -in /etc/ssl/req/careq.pem -extfile /etc/ssl/openssl.cnf \
       -out /etc/ssl/cacert.pem -days 3650
   ```

При создании самоподписанного сертификата последние два действия можно выполнить одной командой:

```
# openssl req -x509 -new -key /etc/ssl/private/cakey.pem \
    -config /etc/ssl/openssl.cnf -out /etc/ssl/cacert.pem -days 3650
```

4.4. Управление сертификатами

4.4.1. Создание сертификата

Вначале создаётся секретный ключ и запрос на получение сертификата:

```
# openssl req -new -config /etc/ssl/openssl.cnf \
    -keyout /etc/ssl/keys/newkey.pem -out /etc/ssl/req/newreq.pem
```

В зависимости от предназначения сертификата задаётся разное значение CN. Если сертификат предназначен для хоста, то в качестве CN задаётся полное имя (FQN) хоста. Для защиты электронной почты (S/MIME) в качестве CN задаётся адрес e-mail.

Следут заметить, что для удобства администрирования следует задавать осмысленное имя сертификата.

Далее следует подписать запрос.

```
# openssl ca -config /etc/ssl/openssl.cnf \
    -extensions server_cert \
    -in /etc/ssl/req/newreq.pem -out /etc/ssl/certs/newcert.pem
```

При генерации необходимо ввести пароль, которым зашифрован секретный ключ сертификата CA.

Значения параметров, не указанные в командной строке, берутся из конфигурационного файла. Опция `-extensions` позволяет выбрать тип сертификата. Следуя нашему конфигурационному файлу, мы можем видеть, что для получения серверного сертификата необходимо задать расширение `server_cert` (в конфигурационном файле — это расширение по умолчанию), а для получения клиентского сертификата следует задать расширение `client_cert`.

В результате выполнения этой команды изменяются файлы `/etc/ssl/index.txt` и `/etc/ssl/serial`.

4.4.2. Аннулирование сертификата

Для того, чтобы аннулировать сертификат, следует задать следующую команду:

```
# openssl ca -config /etc/ssl/openssl.cnf \
    -revoke /etc/ssl/certs/newcert.pem
```

При этом обнавляется база данных `/etc/ssl/index.txt`, и сертификат помечается как аннулированный.

Теперь необходимо обновить список CRL:

```
# openssl ca -gencrl -config /etc/ssl/openssl.cnf -crlexts crl_ext \
    -out /etc/ssl/crl/crl.pem
```

При задании параметра `-crlexts` создаётся CRLv2, в противном случае создаётся CRLv1.

4.4.3. Продление действия сертификата

При возникновении необходимости продлить действие сертификата задаётся следующая команда:

```
# openssl ca -config /etc/ssl/openssl.cnf \
    -in /etc/ssl/req/newreq.pem -out /etc/ssl/certs/newcert.pem \
    -startdate текущая_дата -enddate конечная_дата
```

Дата задаётся в формате YYMMDDHHMMSS (например 031201000000 соответствует времени 00:00:00 1 декабря 2003 года).

4.4.4. Просмотр содержимого сертификата

Содержимое файла сертификата можно просмотреть в текстовом виде:

```
# openssl x509 -in /etc/ssl/certs/newcert.pem -noout -text
```

5. Заключение

При использовании цифровых сертификатов возникают две альтернативы: либо обращаться за услугами к компаниям, специализирующихся на услугах PKI, либо организовывать собственный CA.

В организационном плане создание CA требует выработки его политики, правил сертификации. Для реализации операций сертификации формируется политика операционной работы CA, работы с персоналом и оборудованием, политика управления сертификатами.

Правильная организация процесса выпуска и обслуживания цифровых сертификатов, точное следование нормам практики сертификации являются основой безопасного и надёжного функционирования CA, ключевых элементов обеспечения безопасности корпоративных информационных ресурсов.

Литература

1. *Шнайер Б.* Прикладная криптография. Протоколы, алгоритмы, исходные тексты на языке Си. — М.: Триумф, 2003. — С. 816. — `http://www.ssl.stu.neva.ru/psw/crypto/appl_rus/appl_cryp.htm`.
2. *Семенов Г.* Не только шифрование, или Обзор криптотехнологий // Jet Info. — № 3(94). — 2001. — С. 4–15. — `http://www.jetinfo.ru/2001/3/2001.3.pdf`.

3. *Давыдов А. Н.* Обзор инфраструктур открытых ключей // Труды научно-технической конференции «Безопасность информационных технологий». Секция № 2: Информационная безопасность сложных систем. — Т. 1. — Пенза: 2001. — С. 34–36. — http://beda.stup.ac.ru/rv-conf/v01/015/.
4. *Янглав Р.* PKI: как это работает. — http://www.iso.ru/journal/articles/96.html.
5. Инфраструктура открытых ключей. — http://www.atnn.ru/default.html?/pki.html.
6. *Crocker D. H.* Standard for the Format of ARPA Internet Text Messages. — 1982. — http://ietf.org/rfc/rfc0822.txt?number=822.
7. Internet X.509 Public Key Infrastructure. Certificate and CRL Profile. / R. Housley, W. Ford, W. Polk, D. Solo. — 1999. — http://ietf.org/rfc/rfc2459.txt.
8. *Adams C.* Internet X.509 Public Key Infrastructure. Certificate Management Protocols. — 1999. — http://ietf.org/rfc/rfc2510.txt?number=2510.
9. Internet X.509 Certificate Request Message Format / M. Myers, C. Adams, D. Solo, D. Kemp. — 1999. — http://ietf.org/rfc/rfc2511.txt?number=2511.
10. *Chokhani S., Ford W.* Internet X.509 Public Key Infrastructure. Certificate Policy and Certification Practices Framework. — 1999. — http://ietf.org/rfc/rfc2527.txt?number=2527.
11. *Housley R., Polk W.* Internet X.509 Public Key Infrastructure. Representation of Key Exchange Algorithm (KEA) Keys in Internet X.509 Public Key Infrastructure Certificates. — 1999. — http://ietf.org/rfc/rfc2528.txt?number=2528.
12. *Boeyen S., Howes T., Richard P.* Internet X.509 Public Key Infrastructure. Operational Protocols - LDAPv2. — 1999. — http://ietf.org/rfc/rfc2559.txt?number=2559.
13. *Housley R., Hoffman P.* Internet X.509 Public Key Infrastructure. Operational Protocols: FTP and HTTP. — 1999. — http://ietf.org/rfc/rfc2585.txt?number=2585.
14. *Boeyen S., Howes T., Richard P.* Internet X.509 Public Key Infrastructure. LDAPv2 Schema. — 1999. — http://ietf.org/rfc/rfc2587.txt?number=2587.
15. X.509 Internet Public Key Infrastructure. Online Certificate Status Protocol - OCSP / M. Myers, R. Ankney, A. Malpani et al. — 1999. — http://ietf.org/rfc/rfc2560.txt?number=2560.
16. *Горбатов В., Полянская А.* Доверительные центры как звено системы обеспечения безопасности корпоративных информационных ресурсов // Jet Info. — № 11(78). — 1999. — С. 12–20. — http://www.jetinfo.ru/1999/11/1999.11.pdf.

UDC 681.322

Public Key Infrastructure

D. S. Kulyabov, A. V. Korolkova

Telecommunication Systems Department
Peoples' Friendship University of Russia
Miklukho-Macklaya str., 6, Moscow, 117198, Russia

The articlepresents a survey on the theoretical basis of the PKI. Main terms and components are considered and some practical recommendations on production deploying of the PKI are given.