электронный журнал
МОЛОДЕЖНЫЙ НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК
Издатель ФГБОУ ВПО "МГТУ им. Н.Э. Баумана". Эл No. ФС77-51038.

 # 01, январь 2016

УДК 004.738.2

Обзор основных анонимных коммуникационных систем

Эренджeнова Д. С., *студент*
Россия, 105005, г. Москва, МГТУ им. Н.Э. Баумана,
кафедра «Информационная безопасность»

Ступин Н. С., *студент*
Россия, 105005, г. Москва, МГТУ им. Н.Э. Баумана,
кафедра «Информационная безопасность»

Научный руководитель: Алешин В.А., к.т.н, доцент
Россия, 105005, г. Москва, МГТУ им. Н.Э. Баумана,
кафедра «Информационная безопасность»
v.aleshin@bmstu.net

Введение

Системы анонимной коммуникации [1] можно разделить на две основные категории: системы с большими задержками и системы с малыми задержками. Анонимные системы с большими задержками обеспечивают большую анонимность, но, как правило, применяются для неинтерактивных приложений, которые могут допускать задержки в течение нескольких часов или более (электронная почта). Анонимные системы с малыми задержками обеспечивают большую производительность и предназначены для приложений в режиме реального времени, в частности, для просмотра веб-страниц.

Анонимные системы с большими задержками

Микс-сети (Mix Networks). Основой почти всех современных анонимных коммуникационных систем с большими задержками является *микс (mix – англ.)* [2]. Микс в качестве входа принимает зашифрованные открытым ключом сообщения, группирует несколько сообщений в *пакет (batch – англ.)*, а затем пересылает каждый по адресу назначения, извлеченному из расшифрованного входного сообщения. При этом входное сообщение заполняется случайными битами до фиксированной длины, чтобы противник не смог определить два идентичных сообщения, зашифрованных одним и тем же асимметричным ключом. Наблюдатель, который может только контролировать сообщения, входящие и исходящие из микса, не может соотнести входы и выходы на

основе битов сообщений, так как микс зашифровывает каждое сообщение. Кроме того, сообщения задерживаются, группируются и перемешиваются, так что наблюдатель не сможет связать сообщения, основываясь на времени их прибытия и отправления.

Очевидно, что использование простого микса предполагает не только единую точку отказа, но и единую точку доверия. Поэтому предлагается передавать сообщения не через один микс, а через упорядоченную последовательность миксов, *микс-сеть (mix network – англ.)* [3]. При этом отправитель многократно создает слои шифрования для каждого микса из этой последовательности, начиная с последнего микса в пути и заканчивая первым, и передает получившееся зашифрованное сообщение в первый микс. Каждый микс удаляет слой шифрования, извлекает адрес следующего микса в пути и зашифрованное сообщение для передачи в следующий микс. Как и раньше, в случае одного микса, наблюдатель не в состоянии связать входные сообщения микса с его выходами. Кроме того, каждый микс на пути сообщения знает только то, что было перед ним и после него в пути. Таким образом, ни один оператор микса не может скомпрометировать анонимность сообщений, отправленных через его микс. Для того чтобы длина сообщения не способствовала утечке информации о том, через сколько миксов сообщение уже прошло, каждый микс после удаления заголовка заполняет сообщение до фиксированной длины. На последнем этапе в пути случайно сгенерированное заполнение отделяется от сообщения, и получившееся сообщение отправляется конечному пользователю.

Как правило, при выборе последовательности миксов в качестве пути сообщения рассматриваются две стратегии: *свободные маршруты (free routes – англ.)* и *каскады миксов (mix cascades – англ.)*. Топология свободного маршрута, как следует из названия, означает, что клиенты могут выбрать любую упорядоченную последовательность миксов в сети в качестве пути их сообщения. В топологии каскада существует один или несколько предопределенных маршрутов, через которые передается весь трафик клиента. Было доказано, что сети свободных маршрутов могут обеспечить лучшую устойчивость в случае отказа узла, в то время как отказ микса в каскаде разорвет все связи в этом каскаде, так как в сети свободных маршрутов клиенты могут просто перенаправить свой трафик в обход отказавшего узла.

anon.penet.fi. Одним из первых развернутых ремейлеров был финский сервис обеспечения псевдонимности anon.penet.fi, созданный Йоханом Хельсингиусом в 1993 г. Клиент посылает сообщение на сервис, который затем генерирует псевдоним для

отправителя, если он еще не существует, и сохраняет соответствие между псевдонимным и реальным адресом электронной почты отправителя в базе данных. Затем сервис зачищает любую идентифицирующую информацию из сообщения и передает его получателю. Если ремейлер получает сообщение, адресованное ранее созданному псевдониму, ремейлер смотрит истинный адрес псевдонима в базе данных и сообщение может быть доставлено псевдонимному пользователю. Очевидно, что база данных псевдонимов может стать вероятной мишенью для атаки.

«Шифропанк» (Cypherpunk Remailers). В отличие от системы anon.penet.fi, система ремейлеров «Шифропанк» *(ремейлеры 1-го типа)* состоит из нескольких распределенных серверов. Клиенты выбирают последовательность серверов и подключают заголовок для каждого узла. Каждый узел использует свой закрытый ключ для расшифровки полученного сообщения, удаляет информацию заголовка и отправляет содержимое на следующий сервер. Однако в отличие от микс-сетей, «Шифропанк» не добавляет случайное заполнение сообщений после удаления заголовков. Таким образом, пассивный противник, наблюдая за сетью, потенциально может связать сообщения, зная, что сообщения уменьшаются в длине при прохождении через сеть. Кроме того, ремейлер не делает никакого пакетирования или задерживания, так что пассивный злоумышленник также может связать сообщения, проходящие через ремейлер, на основе количества входящих и исходящих сообщений. «Шифропанк» обеспечивает анонимность ответов с помощью *блоков ответа (reply blocks – англ.)*. Блок ответа создается послойно аналогично обычному сообщению. Отправитель может включить в анонимное сообщение блок ответа, чтобы получатель мог использовать этот блок для ответа на анонимное сообщение. Ответ направляется через сеть ремейлера в соответствии с адресом, зашифрованным в блоке ответа, обратно отправителю. Мэтт Гио [4] позже реализовал *nym-сервер*, который работает аналогично anon.penet.fi, выделяя в базе данных для каждого блока ответа псевдоним. Nym-сервер фактически не отображает псевдоним, т.к. блоки ответа отправляются на nym-сервер анонимно.

Mixmaster. Mixmaster *(ремейлер 2-ого типа)*, улучшил конструкцию по сравнению с ремейлером «Шифропанк», в т.ч. добавил заполнение и пакетирование сообщений. В Mixmaster микс каждые t секунд случайным образом выбирает $\min\{N - N_p, N * f\}$ сообщений для пересылки, где N – общее число сообщений в миксе, N_p – минимальное количество сообщений, удерживаемых в *пуле (pool – англ.)* до следующего раунда, и f –

доля сообщений, пересылаемых на каждом раунде. Однако в отличие от ремейлера «Шифропанк», Mixmaster не включает поддержку анонимных ответов. Mixmaster пытается отразить атаки воспроизведения, записывая идентификаторы пакетов, включенных в заголовок сообщения. Если микс получает сообщение с ID пакета, который уже обрабатывался, дубликат просто отбрасывается. Но поскольку миксы имеют ограниченный объем памяти, противник может легко воспроизвести сообщение еще раз после того, как ID его сообщения был удален из истории микса.

Mixminion. Mixminion *(ремейлер 3-его типа)*, также хранит историю недавно обработанных сообщений, но миксы периодически меняют ключ, который клиенты используют для шифрования сообщений. После изменения ключа сообщения, зашифрованные с помощью старого ключа, больше не принимаются и не могут быть воспроизведены. Таким образом, серверы Mixminion сохраняют только хэши ранее обработанных сообщений. Конструкция Mixminion улучшена по сравнению с Mixmaster в том, что она включает в себя распределенное множество избыточных *серверов каталогов (directory servers – англ.)*, предоставляющее клиентам информацию о текущих миксах в сети (например, их IP-адреса, информация об открытом ключе и т.д.). Как ремейлеры «Шифропанк», Mixminion поддерживает анонимные блоки ответа, только в Mixminion они являются *одноразовыми (single-use reply blocks, SURB – англ.)*. Таким образом, анонимный получатель должен создать новый SURB для каждого ответа, который хочет получить. Несмотря на то что в Mixminion одноразовые блоки ответа устойчивы к атакам воспроизведения, они все еще подвержены к долгосрочным атакам пересечения. Вместо того чтобы полагаться на блоки ответа, разработчики предложили хранить почту, отправленную псевдониму, на nym-сервере. Тогда владелец псевдонима может использовать собственный протокол поиска информации, чтобы извлечь свою почту из nym-сервера.

Анонимные системы с малыми задержками

Анонимные системы с малыми задержками основаны на понятии *прокси (proxy – англ.)*. Если миксы явно группируют и переупорядочивают входящие сообщения, то прокси просто пересылают весь входящий трафик непосредственно без какого-либо переупорядочивания пакетов. К примеру, клиенты компании *anonymizer.com* передают веб-трафик через прокси-серверы, эксплуатируемые компанией, так что веб-запросы

клиента исходят от её серверов. Такой подход является простым и эффективным, но требует, чтобы клиенты были уверены, что компания не контролирует их трафик.

Луковая маршрутизация. Узел, инициирующий передачу потока данных через сеть, создает многократно зашифрованный туннель, или *цепочку*, выбирая в качестве пути цепочки последовательность прокси, *луковых маршрутизаторов (onion routers – англ.).* Он генерирует для каждого маршрутизатора два симметричных ключа (прямой и обратный), зашифровывает их в т.н. «луковицу» и отсылает ее на первый маршрутизатор в цепи, который удаляет верхний слой шифрования, используя свой закрытый ключ, и извлекает симметричные ключи, а также адрес следующего сервера, куда следует передать сообщение. После того как цепочка построена, инициатор шифрует сообщение (например, веб-запрос) для каждого узла в цепочке с использованием прямых ключей, а затем посылает его в первый маршрутизатор в цепочке. Каждый маршрутизатор удаляет слой шифрования и пересылает получившееся сообщение на следующий сервер в цепи, пока оно не достигнет последнего сервера, который затем отправляет исходное сообщение по назначению. Когда получатель запроса отправляет ответ обратно по цепи, слой шифрования *«наращивается»* на каждом маршрутизаторе с помощью обратного ключа. Так как отправитель владеет всеми секретными ключами, ранее созданными для каждого узла, он может удалить все слои шифрования, чтобы восстановить оригинальный ответ.

PipeNet. Пользователи в PipeNet сначала выбирают случайную последовательность серверов в сети на подобии выбора пути в луковой маршрутизации. Затем клиенты создают многократно зашифрованный туннель путем создания симметричного ключа на первом узле, передачи его через зашифрованное соединение и создания другого ключа на втором узле, и т.д. За единицу времени по каждому каналу между сетевыми узлами отправляется ровно один пакет. В результате, один неправильно работающий узел может прекратить функционирование всей сети, так как он не сможет переслать сообщение в течение одной или нескольких единиц времени.

Crowds. Система Crowds была специально разработана для анонимного просмотра веб-страниц. Каждый пользователь в системе представлен приложением *jondo*. Jondo определяют к *толпе (crowd – англ.)* других jondo с помощью административного процесса, называемого *блендер (blender – англ.)*. Блендер также несет ответственность за информирование членов Crowds о регистрации новых jondo.

Когда браузер пользователя делает веб-запрос, его jondo создает случайный путь в сети со случайно выбранным jondo из толпы (возможно, даже с самим собой) и отправляет ему запрос. Каждый jondo выбирает одно из двух действий: отправляет запрос либо выбранному случайным образом другому пользователю из толпы, либо на конечный веб-сервер. При этом вероятность того, что запрос будет передан другому пользователю, $p_f > \frac{1}{2}$. Также каждый jondo запоминает узел-предшественник, таким образом, ответ от веб-сервера передается обратно по установленному пути. Попарные связи между jondo шифруются с использованием общих ключей, которые генерируются блендером, когда новый jondo присоединяется к толпе.

Так как каждый член толпы передает запросы другому члену, то когда jondo получает запрос, он не знает, был ли другой jondo источником запроса или он просто передал запрос от другого jondo; однако, один или несколько вредоносных jondo на пути сообщения могут сотрудничать, чтобы попытаться определить истинного инициатора соединения. Рассмотрим путь, который включает в себя по крайней мере один вредоносный сотрудничающий jondo. Разработчики Crowds доказали, что вероятность того, что предшественником первого сотрудничающего узла будет именно отправитель, такая же, как если бы он не был отправителем (свойство, которое они определили как *возможная невиновность*), если

$$n \geq \frac{p_f}{p_f - 1/2}(c + 1) \ ,$$

где n – общее количество jondo, c – количество сотрудничающих вредоносных jondo.

Очевидно, что существует также выбор между вероятностью пересылки p_f и длиной установленного пути и, следовательно, производительностью для пользователя. В Crowds нет никакого незашифрованного пакетирования, задерживания или переупорядочивания сообщений, так как они передаются через толпу. Это верно почти для всех конструкций анонимных систем с малыми задержками.

Java Anonymous Proxy (JAP). Как следует из названия, веб-миксы – система обеспечения анонимности интерактивных интернет-приложений, таких как просмотр веб-страниц. Однако веб-миксы довольно сильно отличаются от луковой маршрутизации и Crowds. Система состоит из программного обеспечения Java Anonymous Proxy (JAP), которое пересылает трафик клиента в один из нескольких доступных каскадов миксов. Каждый микс в каскаде зашифровывает полученные пакеты и сортирует пакеты между потоками данных перед их передачей в следующий микс. Последний микс в каскаде

пересылает трафик на «кэш-прокси», который направляет анонимный веб-запрос по назначению. Пытаясь помешать анализу трафика, конструкция веб-миксов предусматривает поддержание постоянной скорости трафика между клиентом и первым миксом в каскаде. Она будет вставлять фиктивный трафик, если клиент не имеет никакого фактического трафика данных для передачи. Также предложено постоянное заполнение трафика между кэш-прокси и последним миксом каждого каскада. Однако развернутая реализация не использует покрывающий трафик из-за чрезмерной нагрузки на сеть.

Tarzan. Как и в луковой маршрутизации, инициаторы в сети Tarzan строят цепочки путем создания симметричных ключей для каждого узла и многократного шифрования их открытыми ключами серверов в цепи. Симметричные ключи затем используются для передачи данных по цепи. Как в Crowds, все участники пропускают через себя трафик других пользователей. Особенность Tarzan заключается в том, как клиенты узнают о других серверах в сети. Вместо того чтобы полагаться на небольшое множество серверов каталогов, Tarzan использует peer-to-peer протокол сплетен, для того чтобы делиться информацией о других серверах. Когда узел инициализируется, он выбирает случайный соседний узел из множества начальных узлов и запрашивает у него информацию о других серверах, который тот знает. Затем запрашивающий узел может выбрать другой случайный узел из вновь изученного множества серверов и повторить процесс.

Tarzan также вводит покрывающий трафик в сети, предназначенный для того, чтобы помешать глобальному пассивному противнику определить отправителей по объемам трафика. Когда узел присоединяется к сети, он выбирает k других узлов, которые называются *имитаторами (mimics – англ.)*, и просит их обменять «имитирующий трафик» (mimic traffic – англ.) с ним. Каждый узел выбирает своих имитаторов на основе результата применения криптографической хэш-функции, включающей IP-адрес узла и текущую дату. Выбор имитаторов однозначен, что мешает противнику выбрать более k имитаторов. Выбор имитаторов также симметричен; в противном случае узлы бы только отправляли данные. Инициатор анонимного туннеля создает цепь, выбрав случайным образом первый узел из его множества имитаторов. Второй узел выбирается из множества имитаторов первого выбранного узла, и так далее.

В то время как узел находится в сети, он обменивается имитирующим графиком с соседними имитаторами, который неотличим для внешнего наблюдателя от трафика приложений. При этом скорость, с которой узел посылает исходящий трафик, сопоставима со скоростью, с которой он получает трафик. Поскольку инициатор цепи

также обменивается имитирующим трафиком с другими узлами, тому, кто наблюдает за узлом, будет очень трудно определить инициатора конкретной цепи. Кроме того, первый узел в цепи не знает, является ли полученный трафик покрывающим трафиком или трафиком приложения.

MorphMix. Так же как и Tarzan, MorphMix является peer-to-peer анонимной системой с малыми задержками, в которой все узлы в сети передают трафик других узлов. Как в Tarzan и луковой маршрутизации, MorphMix использует многократно вложенные симметричные шифрования для передачи данных по цепям. Особенностью MorphMix является более сложный подход к обнаружению узлов и созданию цепочек.

Для построения цепи отправитель *a* выбирает узел *b* из множества узлов, о которых он в данный момент знает. Затем узел *a* запрашивает у узла *b* список других узлов, которые ему известны, из которых *a* может выбрать следующий узел для его цепи. Пусть *x* будет следующим выбранным узлом из множества узлов узла *b*. Кроме того, пусть *w* будет другим узлом из числа известных узлу *a*, который будет использоваться в качестве *«свидетеля» (witness – англ.).* Узел *a* устанавливает симметричный ключ с узлом *x* по алгоритму Диффи-Хеллмана, используя открытый ключ узла-свидетеля *w*. Таким образом, исключается возможность проведения атаки «Человек посередине» (при условии, что *x* – честный узел). Инициатор повторяет этот процесс для каждого узла в его цепи.

В случае если новый узел делает запрос злонамеренному узлу, может быть выбран путь, состоящий только из вредоносных узлов. Эти узлы могут тайно сговориться для определения инициатора. Чтобы свести к минимуму эту атаку, разработчики MorphMix добавили *механизм обнаружения сговора (collusion detection mechanism – англ.),* отслеживающий узлы, появляющиеся в туннеле вместе, поскольку вероятность появления вместе сговорившихся узлов более высока, чем у несговорившихся. К сожалению, было доказано, что противник может с легкостью обойти этот механизм, предлагая отправителю множества сговорившихся узлов, пересекающихся как можно меньше.

Tor. Конструкция Tor внесла несколько изменений и улучшений по сравнению с первоначальной конструкцией луковой маршрутизации с точки зрения безопасности, эффективности и развертывания:

Серверы каталогов (Directory servers – англ.). Серверы каталогов несут ответственность за сбор и распространение информации об известных маршрутизаторах в

сети. Информация о каталогах может быть отправлена с других маршрутизаторов в сети для того, чтобы снизить нагрузку на серверы каталогов.

Создание цепочек. Клиенты Tor используют телескопический подход к построению цепочек, впервые описанный в Cebolla. Инициатор цепи итеративно обменивается эфемерными сеансовыми ключами с каждым маршрутизатором в цепи, используя обмен ключами Диффи-Хеллмана (DH) и одностороннюю аутентификацию RSA. Когда установлен общий ключ с первым узлом, инициатор может создать туннель через это зашифрованное соединение и сгенерировать эфемерный сеансовый ключ со вторым узлом, и так далее. Когда цепочка больше не используется, сеансовые ключи отбрасываются. Данный подход обеспечивает *совершенную прямую секретность*, а также предотвращает атаки воспроизведения, поскольку воспроизведение одной половины ключевой информации DH приводит к другому сеансовому ключу.

Скрытые сервисы (Hidden services – англ.). Скрытые сервисы позволяют пользователям предоставлять услуги, такие как веб-сервер, и при этом не раскрывать IP-адрес или место сервера. Скрытые сервисы поддерживают анонимные подключения к подмножеству узлов в сети, известные как *контактные точки (introduction points – англ.).* Клиенты подключаются к контактным точкам скрытого сервиса и указывают другой узел, известный как *точка рандеву (rendezvous point – англ.).* И клиент, и скрытый сервис подключаются к точке рандеву и создают зашифрованный туннель через нее. Этот хитрый подход предназначен для того, чтобы контактные точки не несли ответственность за контент, предложенный компрометирующим скрытым сервисом. Оверлье и Сайверсон [5] описали атаку, позволяющую противнику определить настоящий хост скрытого сервиса. Противник просто запускает узел в сети и отправляет много запросов на скрытый сервис. Со временем, скрытый сервис многократно выберет сервер противника в качестве первого узла в его цепи к точке рандеву, так как путь для цепочки выбирается случайным образом. Противник может статистически определить наиболее вероятный хост скрытого сервиса. Авторы описали контрмеру для их атаки, которую назвали *сторожевыми узлами (entry guards – англ.).* Скрытый сервис выбирает небольшое множество узлов-сторожей, из которого он всегда выберет первый узел для своих цепочек. Конечно, если некоторые или все из выбранных сторожевых узлов управляются противником, то атака еще может быть проведена. Однако если ни один из выбранных сторожевых узлов не скомпрометирован, то атака, описанная выше, *никогда* не произойдет, так как узел противника никогда не будет находиться рядом со скрытым сервисом в цепи на пути к точке рандеву.

Атаки анализа трафика

Анализ трафика [6] является эффективным инструментом, который разработчики систем анонимности используют для определения устойчивости системы к атакам. Анализ трафика игнорирует содержание сообщений и вместо этого пытается получить как можно больше информации только из метаданных сетевого трафика, таких как время прибытия пакета и длина сообщений.

Атаки на отпечатки (Website Fingerprinting – англ.). Противник может узнать, какие сайты посещает определенный пользователь. Для этого он следит за количеством и длиной пакетов данных, принятых отправителем в ответ на запрос. Из этих метаданных противник может построить отпечаток того, как выглядит трафик ответов веб-сайта, полученный с помощью зашифрованного соединения. Обеспечение того, чтобы весь переданный пакет данных имел ту же длину, помогает скрыть истинную природу трафика ответа.

Атаки по времени (Timing Attacks – англ.). Системы анонимности с низкой задержкой, такие как луковая маршрутизация, не вводят какие-либо задержки или значительные изменения временных характеристик анонимного подключения. В результате, пассивный глобальный противник, который может следить за всеми соединениями на входе и выходе анонимной сети, может связать входы и выходы на основе времени между прибытием пакетов. Для предотвращения данной атаки предлагают либо ко времени выполнения добавлять задержки случайной длительности, либо использовать покрывающий трафик.

Атаки от предшественника (Predecessor Attacks – англ.). При анализе Crowds Райтер и Рубин [7] рассмотрели атаку, в которой несколько злоумышленников присоединяются к множеству пользователей и пытаются определить инициатора конкретного постоянного соединения. Когда пути периодически преобразуются в группе, постоянное соединение должно быть восстановлено. Со временем, истинный инициатор соединения имеет большую вероятность быть непосредственным предшественником узла противника, чем любой другой узел. Райт и др. [8] проанализировали для каждой анонимной системы количество преобразований пути, необходимых противнику для определения инициатора. В итоге, было доказано, что нерасширяемые ограниченные сети более устойчивы к данной атаке.

Атаки раскрытия (Disclosure Attacks – англ.). Основаны на методе пересечения различных множеств получателей, когда с каждой новой итерацией количество получателей уменьшается или остается прежним. Атака раскрытия является вычислительно трудной для проведения из-за необходимости нахождения нескольких непересекающихся множеств, но существует вероятностный метод, который облегчает вычисления. Эффективного метода для абсолютного предотвращения атак раскрытия не найдено. Однако часто для снижения эффективности таких нападений предлагается вставка фиктивного трафика.

Заключение

Все больше людей обеспокоены угрозой личной конфиденциальности в Интернете, что, в свою очередь, привело к значительному прогрессу в области анонимных коммуникаций. В то же время для того чтобы анонимные сети стали практичными, надежными и доступными для всех пользователей Интернета необходимо решить несколько задач.

Для того чтобы анонимная коммуникационная система действительно повсеместно использовалась, она должна выдерживать большое количество не только пользователей, но и серверов. Решением может быть разделение анонимной сети на несколько более мелких сетей.

Улучшение пользовательского интерфейса также является ценным направлением исследований. В прошлом, установка и использование многих систем анонимности были несколько громоздкими, ограничивая их принятие широкой общественностью.

Одним из самых важных свойств анонимных систем является устойчивость к блокировке. Позволение законным клиентам получать список IP-адресов серверов, в то же время ограничение доступа к ним злоумышленникам, является сложной задачей.

Любой успешной анонимной сети требуется широкая пользовательская база, а также операторы серверов по всему миру. При анализе эффективности анонимной сети используются несколько факторов затрат и выгод от участия в системе анонимности, такие как финансовые или временные затраты отправки анонимного сообщения и преимущества поведения как честного, так и нечестного узла сети.

Таким образом, маловероятно, что будет найдено простое решение, обеспечивающее доказуемую безопасную, эффективную и практичную анонимность для пользователей по всему Интернету. Знания, извлеченные из текущего исследования анонимности, могут также помочь разрешить и поощрять будущие разработки глобальных

сетей, включающих в себя конфиденциальность и анонимность как фундаментальное свойство.

Список литературы

[1] Edman M., Yener B. On anonymity in an electronic society: A survey of anonymous communication systems // ACM Computing Surveys (CSUR). 2009. Vol. 42. No. 1. P. 1–35. DOI: 10.1145/1592451.1592456

[2] Chaum D. Untraceable electronic mail, return addresses, and digital pseudonyms // Communications of the ACM. 1981. Vol. 24. P. 84–88. DOI: 10.1145/358549.358563.

[3] Peng K. Anonymous communication networks: Protecting Privacy on the Web // Boca Raton, FL, Auerbach Publications, 2014. 203 p.

[4] Parekh S. Prospects for remailers // First Monday. 1996. Vol. 1. No. 2. Available at: http://www.firstmonday.dk/ojs/index.php/fm/article/view/476/833, accessed 15.02.2015.

[5] Overlier L., Syverson P. Locating hidden servers // In Proceedings of the IEEE Symposium on Security and Privacy. Oakland, CA, 2006. P. 100–114.

[6] Danezis G., Clayton R. Introducing traffic analysis // Digital Privacy: Theory, Technologies, and Practices. Boca Raton, FL, Auerbach Publications, 2007. P. 95–117.

[7] Reiter M., Rubin A. Crowds: Anonymity for Web transactions // ACM Transaction on Information and System Security. 1998. Vol. 1. No. 1. P. 66–92.

[8] Wright M., Adler M., Levine B. N., Shields C. The predecessor attack: An analysis of a threat to anonymous communications systems // ACM Transaction on Information and System Security. 2004. Vol. 4. No. 7. P. 489–522.