СПбГЭТУ «ЛЭТИ»
ПЕРВЫЙ ЭЛЕКТРОТЕХНИЧЕСКИЙ

Р.Р. Фаткиева

Основы построения защищенных компьютерных сетей

Протоколы прикладного уровня

СПбГЭТУ «ЛЭТИ», 2021 г.

3.9 Протоколы прикладного уровня

3.9.1 Telnet

Telnet (Telecommunications Network Protocol) одно из старейших приложений Internet, появившееся в 1969 году. Предназначен для взаимодействия через информационную сеть сервера с терминалом. Передача данных осуществляется со стороны сервера по порту 23, со стороны терминала через любой порт. Работа программы клиента заключается в передаче введенных пользователем данных и команд на сервер и принятия данных с сервера. Как правило, протокол Telnet используется для тестирования серверов и сети.

Аутентификация

Функции Telnet в процедуре аутентификации сводятся к пересылке в открытом виде имени и пароля пользователя:

При соединении с сервером пользователю выдается сообщение с именем сервера и строкой для введения имени login.

При получении имени пользователя на сервере запускается программа login, которая запрашивает у пользователя пароль и проверяет его в своем файле паролей.

Если пароль верен, то запускается программа- оболочка пользователя.

Вопросы безопасности. Получение терминального доступа является привлекательным для нарушителя, а передача данных в Telnet в открытом виде делает возможность перехвата паролей легко осуществляемой путем прослушивания сети.

При тестировании сервера или сети администратором могут быть переданы по сети не мало служебных данных, которые могут заинтересовать нарушителя (например, настройки сервера)

Противодействие

Исключение сервиса Telnet в работе предприятия.

Для обеспечение более надежной аутентификации и передачи данных между сервером и клиентов использование протокола SSH, позволяющего осуществить шифрование и передачу данных через виртуальный канал, вместо протокола Telnet.

3.9.2 Протокол FTP

FTP (File Transfer Protocol)- протокол, определяющий правила передачи файлов и управления файлами на удаленной машине.

Работа с FTP, позволяет просматривать содержимое каталогов, файлов производить их поиск, копировать файлы, с удаленной машины

При использовании протокола FTP используется два различных порта сервера. Первый порт с номером 21 используется для передачи команд FTP. Второй порт используется для передачи данных между сервером и клиентом, номер порта зависит от используемого режима передачи данных.

Активный режим канала данных FTP

1. Клиент с произвольно установленного порта посылает запрос на подключение на порт 21 сервера.
2. Выполняется последовательность установления соединения SYN-ACK протокола TCP.
3. Если запрос на подключение не отвергается сервером, то запускается служба FTP. И происходит обмен между клиентом и сервером данных связанных с подключением, а также команд.
4. При необходимости пересылки данных клиент посылает команду (например, dir) и сервер открывает 20 порт.
5. Посылка команды port с номером порта указывает серверу на куда надо пересылать данные. Порт клиент выбирает динамически. Если клиент не выдает команду port, сервер осуществляет активное открытие на тот же самый номер порта, который использовался клиентом для управляющего соединения.
6. Клиент и сервер обмениваются информацией.
7. Клиент посылает команду ouit закрывая соединение
8. При данном режиме возникают проблемы при прохождении пакетов через устройства фильтрации, которые как правило, запрещают инициировать соединение узлам из сети Интернет узлами, находящимися во внутренней сети.

Пассивный режим канала данных FTP

1. Клиент открывает пассивный канал (действия с 1-3) после чего :
2. Сервер извещает клиента о том, на какой порт он ожидает запросов от клиента.

3. Клиент посылает на указанный порт пакет с флагом SYN и номером порта, на который ждет данные.
4. Сервер отвечает пакетом, с установленными флагами SYN и ACK
5. Клиент и сервер обмениваются информацией
6. Клиент посылает команду ouit закрывая соединение

Так как клиент сам инициировал соединение, то устройство фильтрации пакетов может разрешить прием данных от сервера во внешнюю сеть.

Вопросы безопасности.

1.Основным и значительным недостатком FTP-сервиса является передача пароля пользователя через сеть в открытом виде.

2. Возможность осуществления атака через посредника:

Нарушитель, находящийся на узле X, использует узел B для ретрансляции данных получаемых с узла A, при этом он имеет доступ к каталогам узла B, либо для передачи данных узлу B, от имени узла A, а не от своего.

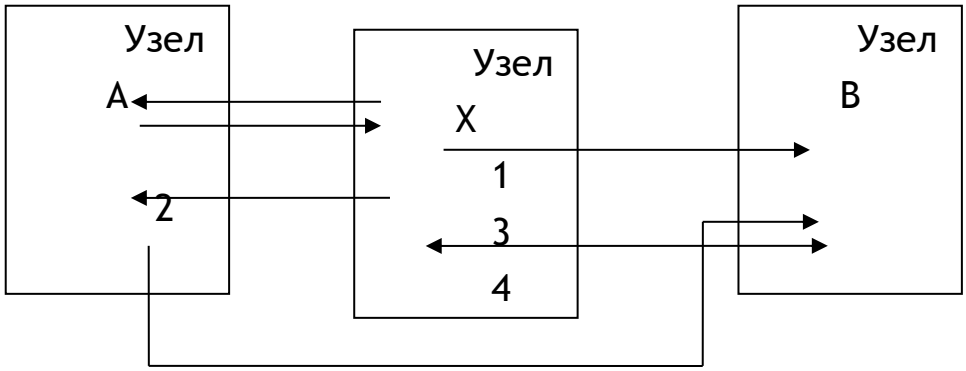

Рис. 3.9

1. Узел нарушителя посылает серверу A команду PASV, для открытия соединения.

2. В ответ узел A, отправляет сокет, по которому готов принять запрос на открытие канала передачи данных.

3. Узел нарушителя посылает серверу B команды:

- PORT с указанием сокета сервера A
- STOR

4. Узел нарушителя посылает серверу A — команду RETR.

5. Сервер A получив команду RETR отправляет файл по каналу передачи данных серверу B.

6. Сервер В получив команду STOR принимает файл из канала передачи данных. Таким образом, действия серверов А и В согласованы, и файл передается с хоста А на хост В.

7. Узел нарушителя Х устанавливает сеанс с узлом В и забирает необходимые ему данные.

Атака через посредника применяется для обхода правил фильтрации на межсетевых экранах, совершения атаки отказ в обслуживании.

3. Данные загружаемые с FTP-сервера могут быть не безопасными для клиента (троянские кони, апплеты)

4. Кража порта. Атака, связанная с возможностью указания номера порта для перехвата передаваемых данных, подстановки данных злоумышленника вместо ожидаемого файла, а также реализации атаки отказа в обслуживании:

1. Нарушитель угадывает номер порта канала передачи данных, который будет открыт в состоянии LISTEN сервером или клиентом. Угадать порт можно:
   - *В пассивном режиме*- путем анализа номеров портов, выделяемых атакуемой системой (например, открытием серий сеансов с последующим анализом закона, по которому изменяются номера портов, возвращаемых сервером, и предсказание следующего номера)
   - *В активном режиме* –путем анализа номера портов, с которых клиент выполняет соединение с компьютером нарушителя.

2. Подсоединившись к открывшемуся порту вместо легитимного узла, отправляет либо получает файл в зависимости от того, какая команда была подана в управляющем сеансе.

Противодействие

1. Безопасность пользовательского доступа может быть обеспечена работой протокола через зашифрованный канал передачи команд. Такая возможность имеется в программных пакетах SSH, обеспечивается аутентификация пользователей с помощью криптографических методов и шифрование всего потока данных.

2. Для борьбы с *атаками через посредника* необходим контроль по совпадению IP-адрес, указанного в команде Port с адресом клиента. При необходимости трехстороннего соединения сервер, отправляющий данные, должен исполнять команду Port только в случае, если указанный порт меньше 1024.

3.Использование прокси-сервера, использующего пассивный режим передачи данных, а также антивирусных программ.

4. Для предотвращения атаки сервер или клиент, открывший порт для канала передачи данных, должен контролировать, что соединение инициируется с того же IP-адреса, с которым установлен командный канал. Правда, в этом случае становятся невозможны «трехсторонние» сеансы.

5. Аудит сетевого трафика

3.9.3 Электронная почта

Для доставки почтовых сообщений используется три вида программ:

- транспортные агенты;
- агенты доставки;
- пользовательские агенты.

Транспортный агент (MTA) функционирует на почтовом сервере, выполняя роль маршрутизатора почтовых сообщений. Его функции включают в себя:

анализ и преобразование адресов и заголовков почтовых сообщений, в том числе:

- трансляция имени почтового домена отправителя;
- прием почтовых сообщений;
- управление очередью сообщений;
- разбор списков рассылки, псевдонимов (aliases), переадресация (forwarding);
- определение агента доставки и передача сообщения выбранному агенту доставки;
- опрос DNS на предмет имени почтового сервера адресата сообщения;
- установка служебных заголовков в сообщении, отражающих его маршрут и процесс обработки;
- преобразование адресов в формат другой почтовой системы (если MTA функционирует как шлюз между двумя почтовыми системами);
- возврат сообщений, в случае невозможности доставки по назначению
- изменение кодировки сообщения;
- криптографическое преобразование текста сообщения и аутентификация пользователя, в случае необходимости.

Агент доставки (MDA) производит доставку сообщения одним из способов:

- local —если сообщение направлено на почтовый ящик, находящийся на этом же компьютере; доставка производиться, например, добавлением содержимого сообщения в определенный файл.
- prog —если сообщение должно быть обработано какой-либо программой; доставка производится вызовом этой программы, на вход которой подается содержимое письма.
- SMTP —если сообщение адресовано на почтовый домен, при этом сообщение отсылается транспортному агенту на удаленный сервер с помощью протокола SMTP.

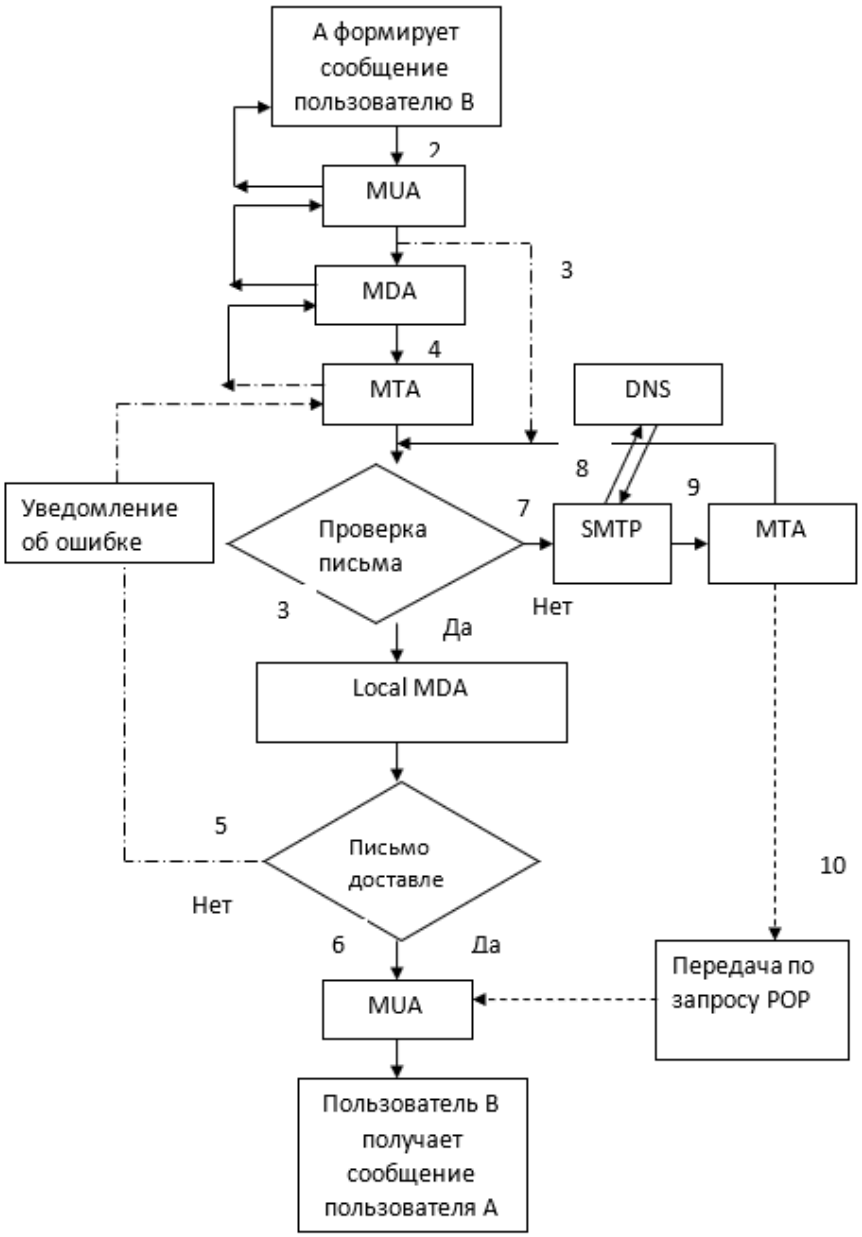

Рис. 3.10

Пользовательский агент (MUA) –оболочка пользователя для работы с электронной почтой, его функции:

- получение сообщений с почтового сервера;
- хранение, каталогизация и удаление почтовых сообщений;
- показ текста сообщения;
- распознавание и извлечение приложений;
- создание нового сообщения
- передача сообщений транспортному агенту для доставки получателю;

Пересылка сообщения. Рассмотрим работу службы электронной почты представленную на рис.

1. Пользователь А пересылает сообщение пользователю В.
2. С помощью программы работы с электронной почты сообщение поступает агенту доставки.
3. Если письмо направлено на почтовый ящик того же компьютера, то сообщение добавляется в определенный файл пользователя В
4. Если письмо направлено во внешнюю сеть, то выбирается транспортный агент, который, проанализировав заголовок сообщения, определяет, адресовано ли оно в почтовый домен, который он обслуживает (то есть адресат письма является локальным). В этом случае для доставки сообщения выбирается пункт 3, в противном случае п.7
5. Агент анализирует код завершения программы доставки, по которому определяет, было ли сообщение успешно доставлено или произошла ошибка. В случае ошибки формирует сообщение об ошибке, которое высылается отправителю письма
6. В случае успешного завершения работы агента доставки письмо доставляется на почтовый ящик пользователя В.
7. Если письмо, направленное во внешнюю сеть и адресовано другому почтовому домену, то сообщение передается SMTP–агенту.
8. SMTP– агент связывается с DNS-сервером для получения IP- адреса домена, на который адресовано сообщение.
9. SMTP–агент отсылает сообщение на удаленный сервер транспортного агента, по адресу присланному DNS-сервером с помощью протокола SMTP. Если удаленный MTA отказался принимать сообщение (например, адрес получателя не существует, то письмо будет возвращено пользователю соответствующим комментарием).

10. Получив письмо, удаленный транспортный агент, либо выполняет действия с п.3, либо ждет, когда клиент, не имеющий доступа к операционной системе сервера, свяжется с сервером по протоколу POP-3. или IMAP-4

Методы защиты электронных сообщений. Рассмотрим механизм аутентификации, использование которого возможно в любых протоколах –SASL (Simple Authentication and Security Layer):

1. Клиент подает команду аутентификации, содержащую указания на алгоритм аутентификации.

2. Происходит обмен запросами и ответами между сервером и клиентом. Сервером осуществляется запрос, на основании которого клиентом вычисляется ответ и возвращается серверу. Запросы и ответы определяются механизмом аутентификации. Получив ответ от клиента, сервер отвечает одним из способов:

Передачей следующего запроса, основанного на механизме аутентификации;

- объявлением об успешной аутентификации;
- объявлением о неуспешной аутентификации.

При получении запроса клиент может возвратить ответ или объявить об отказе от аутентификации.

Наиболее распространенными являются механизмы, основанные на MD5: DIGEST-MD5 и CRAM-MD5:

- CRAM-MD5 (RFC 2195). Сервер в приглашении или первом оклике посылает сроку, состоящую из случайного числа, временной отметки и доменного имени сервера. Ответ должен содержать идентификатор клиента, пробел и HMAC-MD5 оклика. В качестве ключа в HMAC-MD5 используется разделяемый секрет. Аутентификации сервера нет. Подвержен атаке со стороны фальшивого сервера (chosen plaintext attack).
- DIGEST-MD5 (RFC 2831). Начальная аутентификация: сервер посылает оклик, состоящий из имени/пароля, случайной строки, уровня обеспечиваемой защиты (аутентификация, аутентификация и защита целостности, и т д), максимального размера буфера шифрования, наличие поддержки utf-8 для имени и пароля, алгоритма аутентификации, алгоритма шифрования (3des, des, rc4-40, rc4, rc4-56). Клиент отвечает строкой, содержащей ответ на запрос сервера. Сервер проверяет ответ на оклик и посылает ответ в похожем формате для аутентификации.

Целостность обеспечивается добавлением к каждому сообщению типа сообщения и его номера. При криптографическом закрытии сообщение шифруется принятым алгоритмом шифрования и дополняется частично шифрованным заголовком, обеспечивающим целостность.

- PLAIN (RFC 2595). Применяется в сочетании с TLS, обеспечивающим шифрованное соединение и аутентификацию сервера. Клиент посылает идентификатор авторизации, идентификатор аутентификации и пароль в открытом виде. Сервер проверяет идентификатор аутентификации и пароль на соответствие своей базе и наличие у данного идентификатора аутентификации права на идентификатор авторизации.
- OTP (RFC 2444; One-Time-Password). Генератор одноразовых паролей использующий оклик сервера и секрет клиента для получения пароля на один сеанс. Последовательность паролей для одного и того же отклика получается с помощью многократного применения MD5, SHA1 или MD4. От сервера требуется хранение БД последних использованных паролей для каждого пользователя. Аутентификации сервера нет.

Однако не следует забывать, что аутентификация не спасает от (табл 4):

перехвата TCP-соединения, когда нарушитель ретранслирует команды аутентификации между клиентом и сервером, и после успешной аутентификации, блокирует клиента, передавая команды от его имени;

атаки со стороны фальшивого сервера.

Таблица 1

Реализация	Противодействие
Недобросовестное использование электронной почты. Атака на компьютер пользователя посредством рассылки почтовых вирусов	Установка запрета на автоматическое открытия вложений; При необходимости открытия подозрительного письма сохранение его на отдельной дискете, с последующей проверкой антивирусными программами; Своевременное обновление антивирусных программ и проверка файлов системы.
Недобросовестное использование электронной почты. Отправка поддельных писем от имени любого пользователя возможна путем изменения настройки своего пользовательского агента или непосредственно открыв SMTP-сеанс с транспортным агентом.	определение адреса нарушителя исследованием заголовков Received; идентификация пользователя с помощью протокола RFC-1413. Сервер посылает клиенту на порт 113 запрос с номерами портов открытого клиентом SMTP-соединения. Если соответствующая программа запущена, то серверу возвращается

	имя пользователя, что позволяет произвести идентификацию клиента; аутентификация SMTP-сеанса; криптографическое закрытия текста сообщения, в том числе и ЭЦП
Недобросовестное использование электронной почты. Чтение чужих писем нарушителем, при прослушивании сети на пути почтового сообщения или «нечестным» администратором почтового сервера	Криптографическое закрытия текста сообщения, в том числе и ЭЦП
Атака на почтовый сервер (для проникновения в ОС DOS-атаки). Атака с помощью агента доставки Узким звеном в обеспечении безопасности почтового сервера являются агенты доставки, так как получение сообщения с локальным адресом вызывает запуск некоторой программы, на вход которой подается почтовое сообщение неизвестного содержания, которое может быть использовано для проникновения в ОС сервера или осуществления DOS-атаки.	использование стандартных агентов доставки (т.к. обнаруженные ошибки быстро публикуются и исправляются) тестирование агентов доставки на наличие ошибок; разрешение пользователям использования для работы с почтой только проверенные, одобренные программы
Атака почтовой бомбой Атакуемая система переполняется письмами до тех пор, пока она не выйдет из строя из-за: ○ переполнения диска или квоты пользователя, последующие письма не принимаются. ○ переполнения входной очереди сообщениями, которые нужно обработать и передать дальше ○ установки максимального числа сообщений, которые пользователь может принять за один раз. Последующие сообщения отвергаются или уничтожаются. ○ большой размера почтового ящика, создающего трудности для получение системных предупреждений и сообщений об ошибках	
Использование почтового сервера в качестве ретранслятора спама, вирусов. Открытые ретрансляторы (транспортные агенты, используемые в качестве «перевалочного пункта» почтовых сообщений) могут использоваться нарушителями для рассылки спама и вирусов. Для этого осуществляется SMTP-сеанс с открытым ретранслятором, которому указывается произвольный список получателей и необходимое сообщение.	конфигурация транспортного агента таким образом, чтобы решение о возможности обработки сообщения принималось на основании IP-адреса SMTP-клиента, почтовых адресов отправителя и получателя; конфигурация транспортного агента на отказ приема сообщений из доменов или с адресов, зарекомендовавших себя как

Транспортный агент ретранслятора разошлет сообщение по всем указанным адресам.	распространители спама (на основе базы RBL); использование SMTP-аутентификации пользователей;
Перехват паролей в POP- и IMAP-сеансах, для получение и удаление почты без ведома пользователя	Аутентификация на основе протокола APOP или SASL (Simple Authentication and Security Layer).
перехват паролей в SMTP-сеансах, для отправки почты через данный сервер	
перехват TCP-соединения после успешной аутентификации клиента	

3.10 ПРОТОКОЛ HTTP

WWW-клиент-серверная технология, основанная на протоколе HTTP. В простейшем случае на запрос клиента сервер предоставляет некоторую информацию, называемую контектом. В более сложной ситуации в процессе передачи данных могут участвовать несколько промежуточных объектов. Каждый контект(документ) определяется указателем на ресурс– URL или URI.

URL ресурса, доступного через протокол http выглядит следующим образом:

протокол	**имя пользователя**	**пароль пользователя**	**Доменное имя www-сервера**	**Путь к файлу**	**TCP- порт сервера**	**метка, указывающая место, внутри документа, начиная с которого браузер должен показывать**	**аргументы запроса**
http://	user:	password	@www.server:	port	/path	#fragment	&query

Не все поля могут присутствовать в запросе, как правило, обязательное поле www.server через DNS преобразуется в IP-адрес по которому происходит соединение.

HTTP-прикладной клиент-серверный протокол, работающий поверх TCP, без сохранения состояния. То есть запрос от клиента является абсолютно самостоятельной единицей, не связанной ни с запросами этого или других клиентов. Рассмотрим стандартный запрос клиента к серверу:

1. Передача и прием запроса от клиента к серверу.
2. Разбор URI, определение имени запрошенного файла, виртуальный путь преобразуется в реальный.
3. 'Разбор заголовков HTTP-запроса.

4. Осуществление контроля доступа путем анализа параметров запроса, которые могут применяться с дополнительными модулями:
   - аутентификация – установление пользователя, от чьего имени HTTP-браузер произвел запрос;
   - авторизация— определение, имеет ли право пользователь, прошедший аутентификацию производить данный запрос.
5. Определение типа запрашиваемого документа (файл, подлежащий просмотру, CGI-программа, файл с директивами SSI, файл со встроенным кодом и т., а также язык документа, кодировка символов).
6. Генерация ответа. На этой стадии извлекаются или генерируются в результате работы программы файлы или процедуры-обработчика. Формируемый HTTP-ответ возвращается клиенту. Если запрос не может быть обслужен, сервер возвращает клиенту ответ с кодом ошибки.
7. Протоколирование. На этой стадии в журналы аудита заносятся записи об обслуживании запроса.
8. Освобождение ресурсов: памяти и файлов и т. п.

В процессе TCP соединения сервер и клиент обмениваются запросами-HTTP- сообщениями, состоящими из текстовых заголовков, пустой строкой, отделяемой данные от заголовка и собственно данных. Формат http-заголовка запроса выглядит следующим образом:

Строка запроса (1 строка)		
метод	URI	Протокол/версия
Заголовки запроса (1 и более строк)		
Заголовки данных (1 и более строк)		
Пустая строка		
Данные		

Первая строка запроса необходима для определения метода запроса к документу, которые бывают следующего вида:

get	Запрос документа, указанного в поле URI
Head	Выяснение характеристик документа, без передачи содержания (размер, дата модификации, тестирование ссылок)
Post	Пересылка данных на сервер с помощью форм
Options	Запрос списка параметров
Put Delete	Управление документами расположенными на сервере

Вторая строка запроса необходима для передачи параметров запроса, которые могут быть следующими:

Host	Доменное имя сервера, к которому обращается клиент

Accept	Тип данных, которые клиент может принять в ответ на свой запрос. Используется если сервер может предоставить один и тот же документ в различных форматах
Accept-Charset	Кодировка символов в запрашиваемом документе. Несколько кодировок перечисляются через запятую
Accept-Encoding	Возможность передачи документа в сжатом виде
Accept-Language	Язык документа, если сервер предоставляет документ на разных языках
User-Agent	Тип браузера клиента (Netscape, Explorer)
Referer	Указание документа, в котором находилась ссылка на запрашиваемый документ
Authorization	Аутентификация
Connection	Указание на соединение, которое поддерживается некоторое время для последующих запросов
If-Modified-Since	Сервер вернет запрос если метка времени последней модификации устарела
Range	Запрос части документа

Четвертая строка запроса необходима для передачи данных как от пользователя, так и от сервера, как правило, используется при передачи заполненных пользователем форм:

Content-Type	Необходим для определения интерпретации браузером данных, передаваемых пользователю
Last-Modified	Время последней модификации
Expires	Время истечение срока годности
ETag	Идентификатор, присеваемый документу, изменение документа, приводит к изменению идентификатора
Content-Lengh	Размер данных в байтах
Content - Encoding	Метод передачи данных в сжатом виде
Content-Range	Передача документа по частям, с номерами первого/последнего байта и общей длины документа

На запрос клиента сервер присылает ответ формат заголовка которого представлен:

Статусная строка (1 строка)		
Протокол/версия	код	статус
Заголовки ответа (1 и более строк)		
Заголовки данных (1 и более строк)		
Пустая строка		
Данные		

Статусная строка содержит код результата обработки запроса в виде трехзначного числа, и статуса- текстовой интерпретации кода, удобной для чтения пользователем.

Коды подразделятся на группы в зависимости от первой цифры:

1**- промежуточные информационные сообщения;

2**-успешная обработка запроса;

3**- для получения документа требуются дополнительные действия со стороны пользователя;

4**-запрос является ошибочным;

5**-запрос не обслужен из-за ошибки пользователя

Вторая строка ответа необходима для передачи параметров сервера, которые могут быть следующими:

Server	Тип, версия и дополнительные характеристики сервера (Apache/ 1.3.6 rus)
Data	Дата и время ответа
Accept-Range	Возможность выдачи документа по частям
Authorization	Аутентификация
Location	Новый адрес запрошенного документа, возвращается с кодом об ошибке 301, 302
Connection	Состояние соединение
Allow: Get,Head,Post	Перечисление методов, поддерживаемых сервером

Строка данных аналогична описанной в заголовке запроса.

Аутентификация. При обращении к www-серверу, содержащему открытую информацию аутентификация не используется, однако, протокол http предоставляет возможность использование 2 схем аутентификации:

- Basic
- Digest

Схема Basic выглядит следующим образом:

1. Браузер клиента обращается к серверу с запросом на некоторый документ.
2. Так как для доступа к ресурсам сервера требуется аутентификация в поле запроса– Authorization, которые браузер клиента не предоставил, то сервером генерируется сообщение кодом статусной строки 401 “Unauthorized” и заголовком ответа WWW-Authenticate: Basic realm= «Secret Database», в котором указывается метод аутентификации Basic и ресурс realm.
3. Получив ответ сервера, браузер генерирует окно с приглашением ввода имени и пароля пользователя. После ввода которых браузер вновь

обращается к серверу, в заголовке запроса которого указывается схема аутентификации, преобразованные по алгоритму Base 64 имя и пароль, разделенные двоеточием.

4. При получении запроса сервер извлекает из заголовка имя и пароль и сверяет их по своей базе данных. При положительной аутентификации запрос передается на обслуживание. При отрицательной трехкратно повторяется п.3, после чего в окне браузера клиента отображается содержимое ответа сервера.

Недостаток данной схемы заключается в том, что протокол http-протокол без сохранения состояния, поэтому при обращении к следующему документу в этой же сессии, сервер посылает запрос на новую аутентификацию. Однако браузер в этом случае, не беспокоя пользователя самостоятельно подставит уже введенные пользователем данные, и только в случае, если эти данные не пройдут аутентификацию на сервере, то браузер сгенерирует окно для ввода имени и пароля. Однако данная схема облегчает работу как пользователя (не требует постоянное введение пароля), так и работу нарушителя, прослушивающего сеть (так как пароль передается при каждом новом запросе).

Для устранения данного недостатка используют схему Digest

Схема Digest выглядит следующим образом:

1. Браузер клиента обращается к серверу с запросом на некоторый документ.
2. Так как для доступа к ресурсам сервера требуется аутентификация в поле запроса– Authorization, которые браузер клиента не предоставил, то сервером генерируется сообщение кодом статусной строки 401 “Unauthorized” и заголовком ответа:

Заголовок ответа	**Описание**
WWW-Authenticate: Digest realm= «документ»,	метод аутентификации Digest и ресурс realm
Domain= “URI1;URI2”,	Список документов, внутри которых будут действительны запрашиваемые имя и пароль
nonce=”значение1”	Случайное число, генерируемое сервером и используемое при хешировании пароля
[opaque=”значение2”]	Идентификатор сессии
Stale={true|false},	Установленное поле true указывает браузеру о неудачной аутентификации
algorithm={“MD5” “MD5-sess”}	Используемый алгоритм хэширования

Qpo={“auth”| “auth-int”}	

3. Получив ответ сервера, браузер генерирует окно с приглашением ввода имени и пароля пользователя. После ввода которых, браузер генерирует:
   - заголовок запроса, в котором указываются следующие поля:

Заголовок запроса	**Описание**
Authorization: Digest username= «Имя пользователя»,	Метод аутентификации Digest имя пользователя
realm=”документ”	ресурс
nonce=”значение1”	Случайное число, присланное сервером и используемое при хешировании пароля
uri=”URI”	Используется при хешировании пароля и должно совпадать с запрашиваемым ресурсом
algorithm={“MD5” “MD5-sess”}	Используемый алгоритм хэширования
qpo={“auth”| “auth-int”}	«Метод защиты»
[opaque=”значение2”]	Идентификатор сессии
cnonce==”значение3”	Число, генерируемое браузером и используемое при хэшировании пароля
nc= счетчик	Указывает количество одинаковых значений поля nonce
Response=”хеш пароля”	

   - хеш пароля, вычисление которого производится по следующему алгоритму:

MD5(MD5(A1):nonce:nc:cnonce:qop:MD5(A2)),

где для параметра algorithm=MD5- A1 = username:realm:пароль;

для параметра algorithm= “MD5-sess- A1 = username:realm:пароль:nonce:cnonce;

для параметра qpo=“auth” A2=метод запроса:URI;

для параметра qpo=“auth-int” A2= метод запроса:URI:MD5(данные запроса)

Заполненные заголовки запроса посылаются серверу.

4. При получении запроса сервер извлекает из заголовка данные и сверяет их по своей базе данных, содержащей значения хешированных паролей. При положительной аутентификации запроса сервер включает в ответ заголовок Authentication-Info:

Заголовок ответа	**Описание**

Authorization-Info: nextnonce=”значение”	Указание нового значение nonce, которое должно использоваться для вычисления дайджеста при последующем запросе
qpo={“auth”| “auth-int”}	«Метод защиты» Если qpo=auth, то A2=:URI Если qpo=auth-int, то A2 URI:MD5 (данные ответа)
cnonce==”значение2”	Число, генерируемое браузером и используемое при хэшировании пароля
nc= счетчик	Указывает количество одинаковых значений поля nonce
rsauth=”дайджест”	Сервер вычисляет дайджест ответа, по аналогии с дайджестом клиента

и передает запрос на обслуживание.

Таблица 4

Атака	Методы
На механизмы аутентификации (Authentication)	**Подбор** (Brute Force)- процесс проб и ошибок (возможно автоматизированный), угадывания ключевой информации (имя пользователя, пароль, номер кредитной карточки, ключ шифрования и т.д.) Возникает в случае, когда система позволяет пользователям использовать слабые пароли или ключи шифрования. Существует прямой подбор используются различные варианты пароля для одного имени пользователя и обратный-пароль остается неизменным, а перебирает различные имена пользователей.
	Недостаточная аутентификация (Insufficient Authentication) возможность получения доступа к информации или функциям сервера без должной аутентификации (доступ к ресурсам, сокрытым по определенному адресу, не указанному на основных страницах сервера или других общедоступных ресурсах). Некоторые Web-приложения по умолчанию используют для административного доступа ссылку в корневой директории сервера (/admin/). Так как, разработчик предполагает, что воспользоваться этой страницей невозможно, в связи с тем, что ссылки на неё отсутствует, то реализацией аутентификации пренебрегают. Необходимый URL находится перебором файлов и директорий с использованием сообщений об ошибках, журналов перекрестных ссылок или путем простого чтения документации.
	Небезопасное восстановление паролей (Weak Password Recovery Validation)-возможность несанкционированного получения, модификации или восстановления паролей других пользователей. Функция восстановления пароля применяется Web-серверами, при утрате пароля. Из существующих методов восстановления пароля используется метод "секретного вопроса", ответ на который указывается в процессе регистрации; "подсказка", помогающая вспомнить пароль; часть персональных данных идентифицирующих пользователя (ИНН, домашний адрес почтовый индекс и т.д.). После прохождения процедуры

	идентичности, система отобразит пароль или перешлет его по почте. Уязвимость имеет место, если атакующий получает возможность использовать данный механизм. Например, использование секретного вопроса или указание email пользователя в комбинации с домашним адресом и номером телефона приводит к тому, что данную информацию нарушитель может получить из сетевых справочников и легко воспользоваться ею. Использование подсказок помогает в реализации подбора паролей, стойкость пароля "221277King" нарушается использованием подсказки "д-р+люб писатель", что помогает сформировать относительно короткий словарь для атаки путем перебора.
На механизмы авторизации (Authorization)	**Предсказуемое значение идентификатора** сессии (Credential/Session Prediction)- позволяет перехватывать сессии других пользователей. Доступ некоторых серверов предполагает аутентификацию пользователя в виде указания имени и пароля при первом обращении и дальнейшее отслеживание его сессии, для чего Web-сервер генерирует уникальный идентификатор, присвоенный сессии пользователя, позволяющий обращаться к серверу без повторной аутентификации. Идентификатор сессии храниться в cookie, скрытых полях форм или URL. Если нарушитель сможет определить алгоритм, используемый для генерации идентификатора сессии, то возможно: 1) подключение к серверу, используя текущий идентификатор сессии для доступа к ресурсам или реализации новой атаки ; 2) вычисление или подбор следующего идентификатора сессии; 3) присвоение полученного значения идентификатора cookie/скрытому полю формы/URL.
	Недостаточная авторизация (Insufficient Authorization)- возможность получения информации или функций Web-сервера, доступ к которым должен быть ограничен. Процедура авторизации определяет, какие действия может совершать пользователь, служба или приложение. Некоторые серверы, после аутентификации, сохраняют в cookie или скрытых полях идентификатор "роли" пользователя в рамках Web-приложения. Если разграничение доступа основывается на проверке данного параметра без верификации принадлежности к роли при каждом запросе, нарушитель может повысить свои привилегии, просто модифицировав значение cookie
	Отсутствие таймаута сессии (Insufficient Session Expiration)- использование нарушителем перехваченного идентификатора, необходимого для авторизации. Отсутствие таймаута сессии позволяет нарушителю воспользоваться историей браузера для просмотра страниц пользователя. Если функция выхода из системы перенаправляет на основную страницу Web-сервера, а не завершает сессию, страницы, посещенные пользователем, могут быть просмотрены

	злоумышленником, что, обеспечивает нарушителя доступом к страницам сервера без повторной аутентификации. Большое значение таймаута увеличивает шансы подбора действующего идентификатора.
	Фиксация сессии (Session Fixation)- присваивание идентификатору сессии пользователя заданное значение, необходимое нарушителю. Выделяют два типа систем управления сессиями:"разрешающий"- позволяет браузеру указывать любой идентификатор; "строгий"- обрабатывает только идентификаторы, сгенерированные сервером, при этом нарушителю приходится поддерживать "сессию-заглушку" и периодически соединяться с сервером для избежание закрытия сессии по таймауту. В отличие от кражи идентификатора, данная атака предоставляет злоумышленнику гораздо больший простор для творчества. Атаки, направленные на фиксацию сессии обычно проходят в три этапа. 1) **Установление сессии** -заглушки на атакуемом сервере и получение или выбор идентификатора 2) **Фиксация сессии-**передача значение идентификатора сессии-заглушки браузеру пользователя и фиксация его идентификатора сессии (например, установив значение cookie в браузере с помощью XSS). 3) **Подключение к сессии-** после аутентификации пользователя на сервере нарушитель подключается к серверу, с зафиксированным идентификатором пользователя, и получает доступ к сессии пользователя.
Атаки на клиентов (Client-side Attacks)	**Подмена содержимого** (Content Spoofing)- подмена нарушителем страниц сгенерированных Web-сервером, на свои. Некоторые Web-страницы создаются с использованием динамических источников HTML-кода. Например, фрейм <frame src="http://foo.example/1.html" передается в параметре URL строкой http://foo.example/page?frame_src=http://foo.example/1.html. Если нарушитель заменить значение параметра "frame_src" на "frame_src= http://attacker.example/spoof.html", то будет отображаться результирующая страница, загруженная с сервера нарушителя, хотя в строке адреса браузера пользователя будет отображаться адрес сервера (foo.example). Эта атака так же может использована для создания ложных форм ввода пароля, пресс-релизов и т.д
	Межсайтовое выполнение сценариев (Cross-site Scripting, XSS)- передача атакуемому серверу исполняемого кода, который будет перенаправлен браузеру пользователя. Код создается на языках HTML/JavaScript, VBScript, ActiveX, Java, Flash, и т.д. Переданный код в URL, в заголовках HTTP запроса, значениях полей форм и т.д. исполняется на уязвимом сервере и получает возможность читать, модифицировать или передавать важные данные, доступные с помощью браузера. У атакованного пользователя может быть скомпрометирован аккаунт (кража cookie), его браузер может быть перенаправлен на другой сервер или осуществлена подмена содержимого сервера, возможно

	использование браузером жертвы для просмотра страниц сайта от имени атакуемого пользователя.
	Расщепление HTTP-запроса (HTTP Response Splitting)– посылка нарушителем специальным образом сформированный запрос к серверу, ответ на который интерпретируется целью атаки как два разных ответа Для реализации атаки необходимо как минимум три стороны: Web-сервер, содержащий уязвимость; цель атаки, взаимодействующая с Web-сервером под управлением нарушителя (кэширующий сервер-посредник или кэш браузера); сам нарушитель. Атака осуществляется при возвращении сервером данных, предоставленных пользователем в заголовках HTTP-ответа (при перенаправлении пользователя на другую страницу или сохранение данных пользователя в cookie). Основой расщепления HTTP-запроса является внедрение символов перевода строки (CR,LF) для закрытия первой (стандартной) транзакции и формирования второй в виде вопроса-ответа, полностью контролируемую нарушителем для: - межсайтового выполнение сценариев; -модификации данных кэша прокси-сервера, подделанные нарушителем данные в ответ на запрос пользователя сохраняются на жестком диске и на последующие запросы пользователей по данному адресу возвращают кэшированные данные. - межпользовательская атака-некоторые серверы-посредники разделяют одно TCP-соединение к серверу между несколькими пользователями, при этом второй и последующие пользователи получают в ответ страницу, сформированную нарушителем. - перехват страниц, содержащих пользовательские данные для получения доступа к важной или конфиденциальной информации.
Выполнение кода (Command Execution)	**Переполнение буфера** (Buffer Overflow)– изменение пути исполнения программы путем перезаписи данных в памяти системы. Переполнение буфера возникает, если объем данных превышает размер выделенного под них буфер, данные переписывают другие области памяти, что приводит к возникновению ошибки. Если нарушитель имеет возможность управлять процессом переполнения, то может вызвать отказы в обслуживании, изменить путь исполнения программы и выполнить в её контексте различные действия. Для этого используя переполнение буфера, нарушитель перезаписывает служебные области памяти, например, адрес возврата из функций в стеке или значения переменных в программе. Данная атака обычно возникает при создании программ на языках C и C++.
	Атака на функции форматирования строк (Format String Attack)– с помощью функций форматирования символьных переменных

	путь исполнения программы модифицируется методом перезаписи областей памяти Уязвимость возникает, если пользовательские данные применяются в качестве аргументов функций форматирования строк (printf, setproctitle, syslog и т.д.) При передаче приложению строки, содержащей символы форматирования ("%f", "%p", "%n" и т.д.), у нарушителя появляется возможность: - выполнить произвольный код на сервере; - считать значения из стека; - вызывать ошибки в программе/отказ в обслуживании. Например, Web-приложение хранит переменную emailAddress, используемую в качестве аргумента функции printf для каждого пользователя. Если значение переменной содержит символы форматирования, функция printf будет обрабатывать их согласно заложенной в неё логики. Поскольку дополнительных значений этой функции не передано, будут использованы значения стека, хранящие другие данные
	Внедрение операторов LDAP (LDAP Injection)- атаки направленные на Web-серверы, создающие запросы к службе LDAP на основе данных, вводимых пользователем. Если информация, полученная от клиента, должным образом не верифицируется, атакующий получает возможность модифицировать LDAP-запрос, выполняемый с тем же уровнем привилегий, с каким работает компонент приложения, выполняющий запрос (сервер СУБД, Web-сервер и т.д). Например, передача серверу строки: http://example/ldapsearch.asp?user=* приводит к формированию запроса с фильтром uid=*, что приводит к отображением всех объектов, имеющих атрибут uid.
	Выполнение команд ОС (OS Commanding)- выполнение команд ОС на Web-сервере путем манипуляции входными данными. Если информация, полученная от клиента, должным образом не верифицируется, атакующий имеет возможность выполнить команды ОС через компонент приложения. Например, некоторые языки сценариев позволяют запускать команды ОС, используя варианты функции exec. Если данные, передаются этой функции без проверки, нарушитель может выполнить команды ОС удаленно: exec("ls -la $dir",$lines,$rc); Используя символ ";" (Unix) в параметре dir можно выполнить команду ОС и просмотреть содержимое файла /etc/passwd: http://example/directory.php?dir=%3Bcat%20/etc/passwd.
	Внедрение операторов SQL (SQL Injection) создание SQL запросы к серверам СУБД на основе данных, вводимых пользователем. Если информация, полученная от клиента, должным образом не верифицируется, атакующий получает возможность модифицировать запрос к SQL-серверу, отправляемый приложением, с возможностью полного контроля на сервером СУБД и даже его ОС.

	Выделяют два метода эксплуатации внедрения операторов SQL: обычная атака–нарушитель подбирает параметры запроса, используя информацию об ошибках, генерируемую Web-приложением; атака вслепую – сервер возвращает понятную для пользователя информацию о неправильном вводе.
	Внедрение серверных расширений (SSI Injection)– передача исполняемого кода, который в дальнейшем будет выполнен на Web-сервере. Перед генерацией HTML страницы сервер может выполнять сценарии, например Server-site Includes (SSI). В некоторых ситуациях исходный код страниц генерируется на основе данных, предоставленных пользователем. При передаче серверу операторов SSI, можно осуществить выполнение команд ОС или включить в страницу запрещенное содержимое при следующем отображении. Например, выражение < !--#exec cmd="/bin/ls /" -- > может быть интерпретировано в качестве команды, просматривающей содержимое каталога сервера в Unix системах.
	Внедрение операторов XPath (XPath Injection)– направлены на Web-серверы, создающие запросы на языке XPath на основе данных, вводимых пользователем. Язык XPath предоставляет возможность обращения к частям документа на языке XML. Синтаксис XPath близок к языку SQL запросов. Если запросы XPath генерируются во время исполнения на основе пользовательского ввода, у атакующего появляется возможность модифицировать запрос с целью обхода логики работы программы.
Разглашение информации (Information Disclosure)	**Индексирование директорий** (Directory Indexin)–возможность получения информации о наличии файлов в Web каталоге, которые недоступны при обычной навигации по Web сайту. На запрос пользователя начальной страницы сайта, сервер просматривает основную папку, находит в ней файл, используемый по умолчанию, и на его основе генерирует ответ. При отсутствии данной страницы, Web-сервер предоставляет список файлов в директории. В этой ситуации нарушитель может получить доступ к данным, не предназначенным для свободного доступа (например, администратор, предполагает, что если гиперссылка на документ отсутствует, то он недоступен). Однако современные сканеры уязвимостей могут найти спрятанное содержимое или другие файлы через: –ошибки конфигурации возникают при настройке сложных конфигураций, где некоторые папки должны быть доступны для просмотра, которым может воспользоваться нарушитель. –ошибки реализации, когда сервер генерирует список файлов при получении определенного запроса. -базы данных поисковых машин, которые могут содержать кэш старых вариантов сервера, включая списки файлов
	Идентификация приложений (Web Server/Application Fingerprinting)– определение версий приложений для получения информации об используемых сервером и клиентом ОС, Web-

	северах и браузерах. Обычно подобные атаки осуществляются путем анализа информации, предоставляемой Web-сервером, например: заголовками HTTP-ответов, расширением файлов (.asp, jsp), сообщений об ошибках. Наличие данной информации необходимо для реализации атаки, так как она специфична для каждого варианта ОС или приложения. Детальная информация об инфраструктуре позволяет снизить количество ошибок, и как следствие - общий «шум», производимый атакующим.
	Утечка информации (Information Leakage)– уязвимость возникающая, при публикации на сервере информацию, которая может быть использована для компрометации системы (комментарии разработчиков или сообщения об ошибках). Анализ данной информации позволяет злоумышленнику произвести разведку и получить представление о структуре директорий сервера, используемых SQL запросах, названиях ключевых процессов и программ сервера.
	Обратный путь в директориях (Path Traversal) получение доступа к файлам, директориям и командам, находящимся вне основной директории Web-сервера. Большинство базовых атак, эксплуатирующих обратный путь, основаны на внедрении в URL символов "../", для изменения расположения ресурса, который будет обрабатываться сервером. При фильтрации этой последовательности сервером, злоумышленник может воспользоваться альтернативными кодировками для представления символов перехода по директориям (например, Unicode ("..%u2216" или "..%c0%af"), использование обратного слеша ("..\")). Возможность использования обратного пути в каталогах довольно часто возникает в приложениях, использующих механизмы шаблонов, а также в сценариях или CGI-программах.
	Предсказуемое расположение ресурсов (Predictable Resource Location)– получение доступа к скрытым данным или функциональным возможностям Нарушитель получает доступ к содержимому, не предназначенному для публичного просмотра (временные файлы, файлы резервных копий, конфигурации) путем подбора, который может быть оптимизирован использованием стандартного соглашения об именах файлов и директорий сервера. Получаемые файлы могут содержать информацию о дизайне приложения, информацию из баз данных, имена машин или пароли, пути к директориям, уязвимости, отсутствующие в основном приложении. Например наличие или отсутствие ресурса определяется по коду ошибки (404 в случае отсутствия папки или 403 в случае её наличия на сервере).
Логические атаки (Logical Attacks)	**Злоупотребление функциональными возможностями** (Abuse of Functionality)– использование функций Web-приложения для обхода механизмов разграничения доступа, например - использование функций поиска для получения доступа к файлам за пределами корневой директории Web-сервера;

	- использование функции загрузки файлов на сервер для перезаписи файлов конфигурации или внедрения серверных сценариев; - реализация отказа в обслуживании путем использования функции блокировки учетной записи при многократном вводе неправильного пароля. –использование программы "FormMail" для передачи данных из HTML-формы на указанный почтовый адрес может предоставить нарушителю возможность передавать почтовые сообщения в том числе и спам любому почтовому пользователю, при этом оставаясь полностью анонимным.
	Отказ в обслуживании (Denial of Service) нарушение доступности Web-сервера Атаки, направленные на отказ в обслуживании реализуются как на сетевом, так и на прикладном уровнях. Цель нарушителя использовать функции Web-приложения, для исчерпывания критичных ресурсов системы (вычислительные мощности, ОП, дисковое пространство, пропускная способность каналов связи), или прекращение функционирования системы. Атаки могут быть направлены на любой из компонентов Web-приложения: сервер СУБД, сервер аутентификации и т.д.
	Недостаточное противодействие автоматизации(Insufficient Anti-automation)- эта уязвимость возникает, когда сервер позволяет автоматически выполнять операции, которые должны проводиться вручную К автоматизированным программам относятся роботы поисковых систем, системы автоматизированного поиска уязвимостей и регистрации учетных записей. Подобные программы генерируют тысячи запросов в минуту, что может привести к падению производительности всего приложения.
	Недостаточная проверка процесса (Insufficient Process Validation) –недостаточная проверка сервером последовательности выполнения операций приложения В процессе доступа к некоторым функциям приложения ожидается, что пользователь выполнит ряд действий в определенном порядке, при не выполнении данного условия возникает ошибка, приводящая к нарушению целостности. Например, система электронной торговли предлагает скидку на продукт В, в случае покупки продукта А. Пользователь, не желая покупать продукт А, заполняет заказ на покупку обоих продуктов, получая скидку. После чего возвращается к форме подтверждения заказа и удаляет продукт А из покупаемых, путем модификации значений в форме. Если сервер повторно не проверит возможность покупки продукта В по указанной цене без продукта А, будет осуществлена закупка по низкой цене.

Литература

1. Таненбаум, Эндрю. Компьютерные сети [Текст] : учебное пособие / Э. Таненбаум; [Пер. с англ. В. Шрага], 2003. 991 с. 60
2. Олифер, Виктор Григорьевич. Компьютерные сети. Принципы, технологии, протоколы [Текст] : учеб. пособие для вузов по направлению ”Инфор матика и вычислит. техника” и по специальности ” Вычислит. машины, комплексы, системы и сети”, ”Автоматизир. машины, комплексы, системы и сети”, ” Програм. обеспечение вычислит. техники и автоматизир. систем” / В.Г. Олифер, Н.А. Олифер, 2006. 957 с. 133
3. Борисенко, Константин Алексеевич. Сети и телекоммуникации [Текст] / К. А. Борисенко, 2021. 43 с. 50 4 Дибров, Максим Владимирович. Сети и телекоммуникации. Маршрутизация в IPсетях в 2 ч. Часть 2 [Электронный ресурс] : Учебник и практикум для вузов / Дибров М. В., 2021. 351 с неогр.
4. Дибров, Максим Владимирович. Сети и телекоммуникации. Маршрутизация в IPсетях в 2 ч. Часть 1 [Электронный ресурс] : Учебник и практикум для вузов / Дибров М. В., 2021. 333 с неогр.
5. Шелухин О. И. Обнаружение вторжений в компьютерные сети (сетевые аномалии). Учебное пособие для вузов [Электронный ресурс] / О. И. Шелухин, Д. Ж. Сакалема, А. С. Филинова, 2018. 220 с