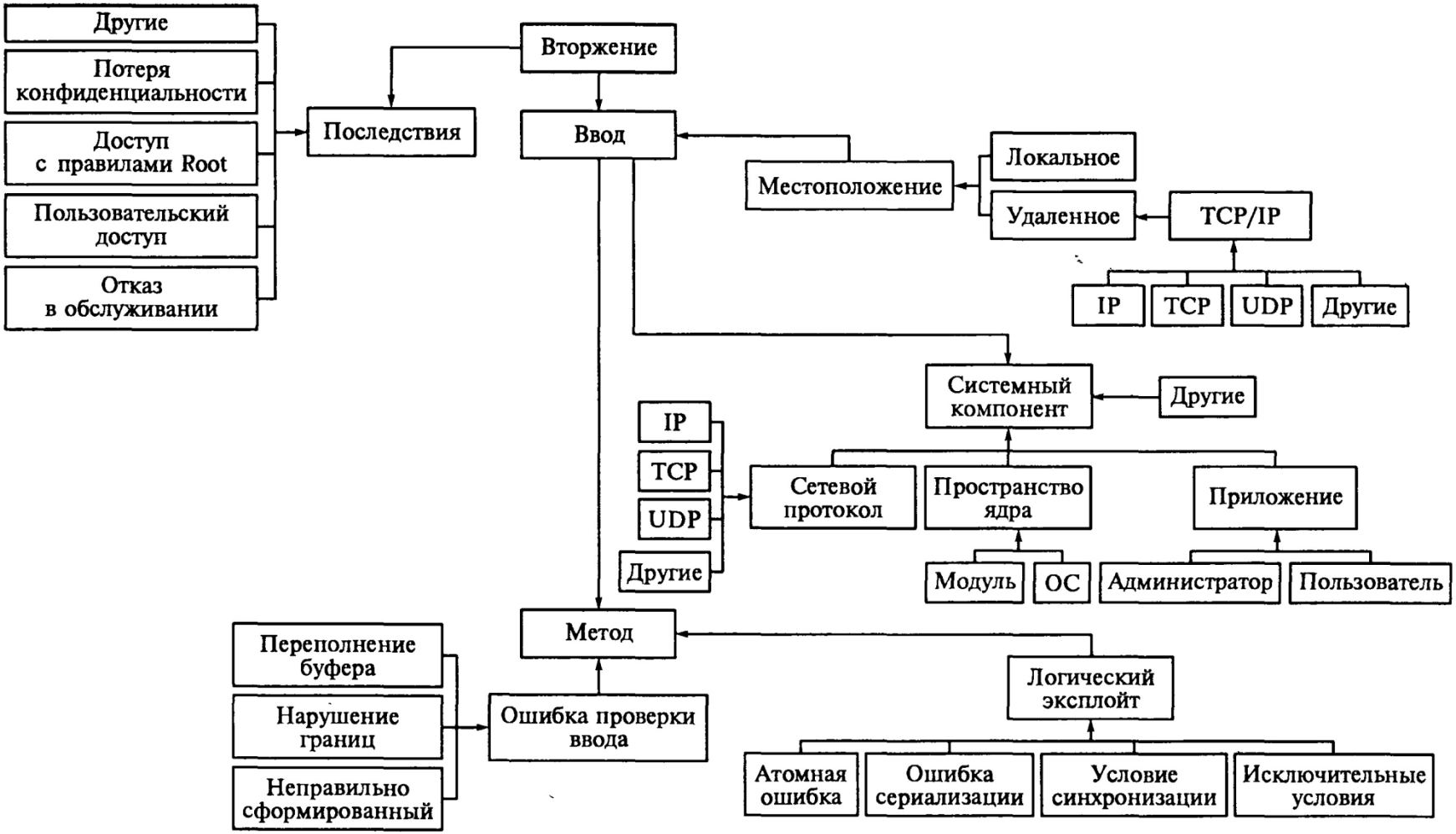
Удалённые сетевые атаки

Онтология сетевых атак



Основные термины

- Угроза безопасности – потенциальная возможность нарушения режима ИБ ЛВС (для скрытого нецелевого использования ресурсов КС, либо иного воздействия, препятствующего нормальному функционированию КС.)
- Уязвимость – намеренная или случайная ошибка в системе, позволяющая реализовать угрозу ИБ
- Атака – процесс подготовки и реализации каких-либо угроз безопасности ЛВС
- Вирус - программа, способная создавать свои копии и внедрять их в файлы, системные области компьютера, компьютерных сетей, а также осуществлять иные деструктивные действия. Копии сохраняют способность дальнейшего распространения.
- Вредоносная программа — компьютерная программа или переносной код, предназначенный для реализации угроз ИБ К вредоносным программам относятся компьютерные вирусы, «трояны», сетевые черви и др.

Атака на информационную систему - событие или совокупность событий, которые применительно к каждому отдельно взятому объекту должны рассматриваться в качестве попыток совершения информационного воздействия противоправного или деструктивного характера.

Обнаружение атак - это процесс оценки подозрительных действий в защищаемой сети, который реализуется либо посредством анализа журналов регистрации операционной системы и приложений либо сетевого трафика.

Цель обнаружения атак - выявить признаки атак либо во время их, либо постфактум. В качестве таких признаков могут выступать:

- повтор определенных событий;
- неправильные или несоответствующие текущей ситуации команды;
- использование уязвимостей;
- несоответствующие параметры сетевого трафика;
- непредвиденные атрибуты;
- необъяснимые проблемы;
- дополнительные знания о нарушениях.

Основные типы атак

- ***Предварительный сбор информации.*** Эти атаки включают ping-sweeps, передачу DNS-зоны, разведку с помощью e-mail, сканирование TCP или UDP-портов и анализ общественно доступных серверов с целью нахождения cgi-дыр.
- ***Попытки несанкционированного доступа.*** Атаки, основанные на использовании преимуществ скрытых возможностей или ошибок (exploit) для получения несанкционированного доступа к системе. Сюда также относятся попытки получения НСД с использованием «троянских коней».
- ***Атаки типа "отказ в обслуживании" (Denial of Service, DoS).*** Когда нарушитель пытается разрушить сервис (или компьютер), перегрузить сеть, перегрузить центральный процессор или переполнить диск. Нарушитель не пытается получить информации, а только затрудняет или блокирует доступ к ней для легальных пользователей.

Методы предварительного сканирования

- Сканирование позволяет осуществлять поиск каналов передачи данных. Идея сканирования заключается в том, чтобы исследовать как можно больше потенциальных каналов связи и определить, какие именно находятся в состоянии ожидания соединения.
- Сканирование портов может являться первым шагом в процессе взлома или предупреждения взлома, помогая определить потенциальные цели атаки. С помощью соответствующего инструментария путем отправления пакетов данных и анализа ответов могут быть исследованы работающие на машине службы (**Web-сервер, FTP-сервер, mail-сервер**, и т. д.), установлены номера их версий и используемая операционная система.
- Службы, находящиеся на хостах, адресуются двумя идентификаторами: IP-адресом и номером порта. Существует 65536 возможных номеров портов.
- Некоторые сканеры портов ищут только наиболее часто используемые, или наиболее уязвимые порты определённого хоста или набора хостов.

Определение активных портов удаленного компьютера

Типы сканирований:

- Ping (определение состояния сервера по ICMP)
- SYN-сканирование
- TCP-сканирование
- UDP-сканирование
- ACK-сканирование
- FIN-сканирование
- Другие типы сканирования

Результат сканирования порта обычно подпадает под одну из трёх категорий:

- **Открыт**, или **соединение принято** (*open*): хост послал ответ, подтверждающий, что хост «слушает» — принимает соединения на данный порт.
- **Закрыт**, **не слушает** (*closed*): хост послал ответ, показывающий, что соединения на данный порт будут отвергнуты.
- **Заблокирован**, **отфильтрован** (*filtered*, *firewalled*): от хоста не поступило ответа.

Уязвимости, связанные с открытыми портами, подразделяются на:

- проблемы с безопасностью и стабильностью, связанные с функционированием программ, предоставляющих сервисы,
- проблемы с безопасностью и стабильностью, связанные с операционной системой, работающей на хосте.

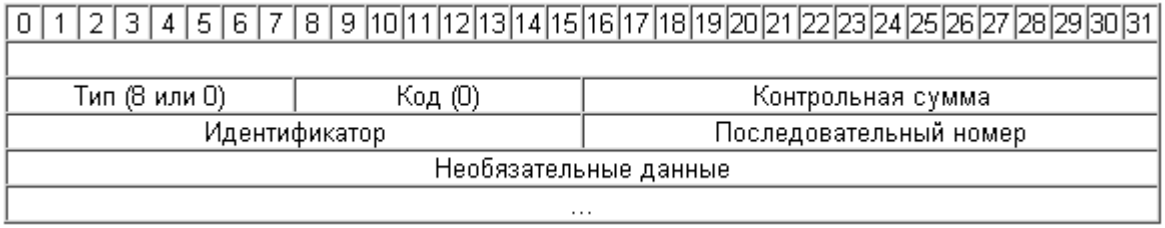
Определение состояния сервера методом ICMP-сканирования

- Перед непосредственным сканированием портов удаленного хоста необходимо выяснить его состояние – работает он в сети или нет.
- В сетях, организованных на базе стека протоколов TCP/IP, для этой цели используется протокол ICMP (RFC 792: Internet Control Message Protocol).
- Обмен информацией между маршрутизатором и хостом реализован с помощью т.н. ICMP-сообщений.
- Хост отправляет серверу ICMP-сообщение и ожидает получения ответа, также представляющего собой ICMP-сообщение (т.н. ICMP-эхо).
- В начале любого ICMP-сообщения находятся три поля: **«Тип сообщения»**, **«Код»** (причина ошибки) и **«Контрольная сумма»**. Поле “Тип” определяет смысл ICMP-сообщения и соответствующий ему формат.
- Используются типы сообщений **«Запрос эха»** (8) и **«Ответ на эхо»** (0).
- ICMP-запросы передаются в IP-пакетах, успешный прием ответа свидетельствует о работоспособности транспортной системы, т.е. выполнена маршрутизация, работоспособны все промежуточные маршрутизаторы, получатель активен и работает корректно, программное обеспечение протоколов IP и ICMP выполняет свои функции.

Определение состояния сервера методом ICMP-сканирования

Код	Значение
0	Ответ на эхо
3	Получатель недостижим
4	Подавление источника
5	Изменение маршрута
8	Запрос эха
11	Превышено время дейтаграммы
12	Ошибка параметров дейтаграммы
13	Запрос временной метки
14	Ответ на запрос временной метки
17	Запрос маски адреса
18	Ответ на запрос маски адреса

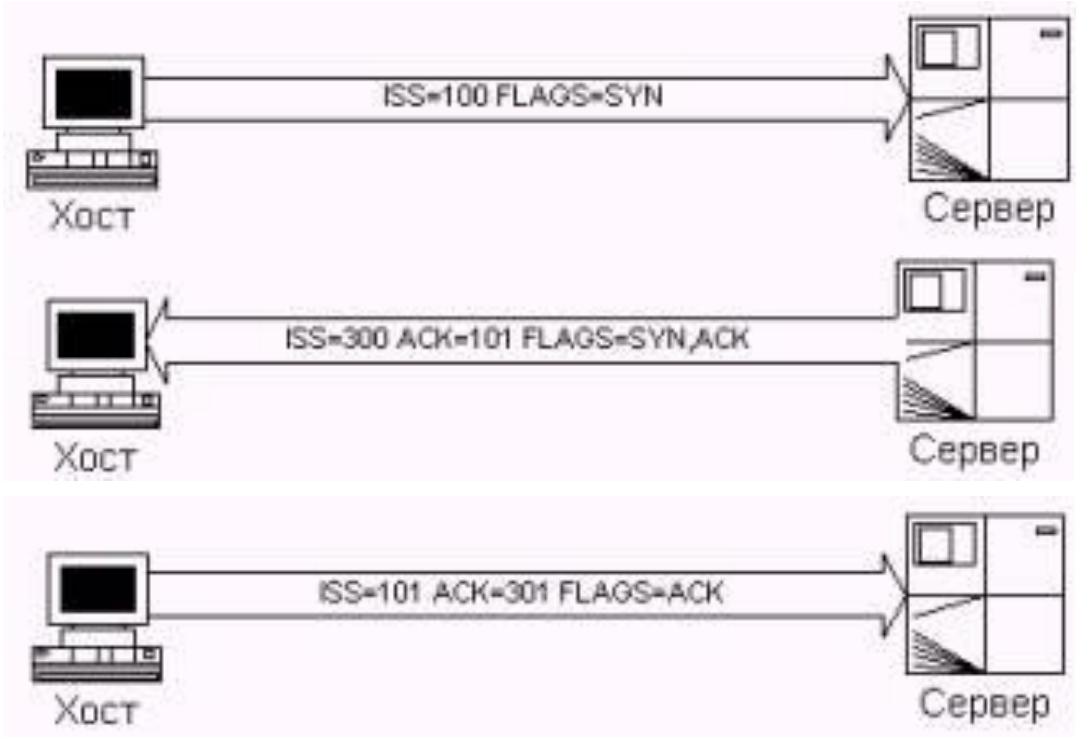
Значения поля «Тип»



Формат сообщения "запрос эха" и ответа на него

Сканирование TCP-портов флагом SYN

Данный метод известен еще как «сканирование с установлением наполовину открытого соединения» (half-open scanning), поскольку полное установление TCP-соединения не производится.



В случае приема SYN|ACK-пакета хост немедленно отправляет RST-пакет для сброса устанавливаемого сервером соединения и не продолжает процесс обмена синхропакетами. Таким образом, производится проверка способности сканируемого сервера установить соединение по указанному порту.

Сканирование TCP-портов флагом FIN

FIN-пакеты способны обойти средства защиты сети. Идея заключается в том, что согласно RFC 793, на прибывший на закрытый порт FIN-пакет сервер должен ответить RST-пакетом (TCP-пакет с установленным в нем флагом RST). FIN-пакеты на открытые порты игнорируются сервером.

Отказ в обслуживании (DoS, DDoS)

- Атаки, заключающиеся в посылке специальным образом сформированных данных, которых система не ожидает, что приводит к её остановке или перезагрузке.
- Атаки, приводящие к переполнению системы или сети при помощи такого большого количества данных, что его нельзя обработать.

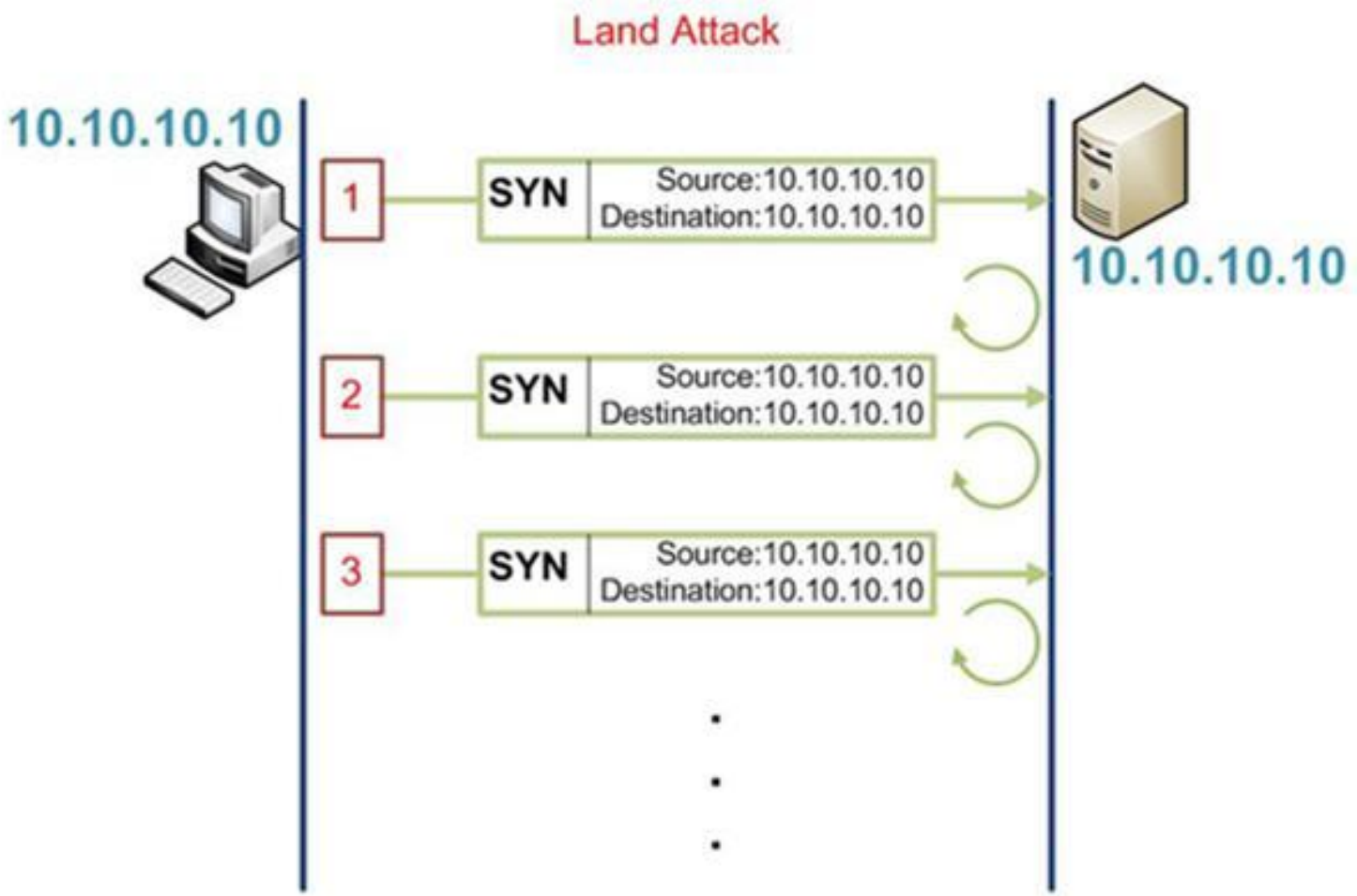
Атака Teardrop

```
10:25:48.205383 wile-e-coyote.45959 > target.net.3964: udp 28 (frag 242:36@0+)
10:25:48.205383 wile-e-coyote > target.net: (frag 242:4@24)
```

В верхней строке показан фрагмент номер 242 с 36 октетами данных со смещением 0. Во второй строке представлены 4 дополнительных октета данных со смещением 24. Таким образом, чтобы обработать этот пакет, операционная система должна вернуться от 36 к 24. Отрицательные числа могут быть преобразованы в очень большие положительные. Вполне вероятно, что операционная система перезапишет бесполезными данными раздел памяти, используемый какой-то другой программой. После пары таких попыток система будет подавлена.

Атака Land

```
10:56:32.395383 200.0.0.104.139 > 200.0.0.104.139: S
10:56:35.145383 200.0.0.104.139 > 200.0.0.104.139: S
10:56:36.265383 200.0.0.104.139 > 200.0.0.104.139: S
```



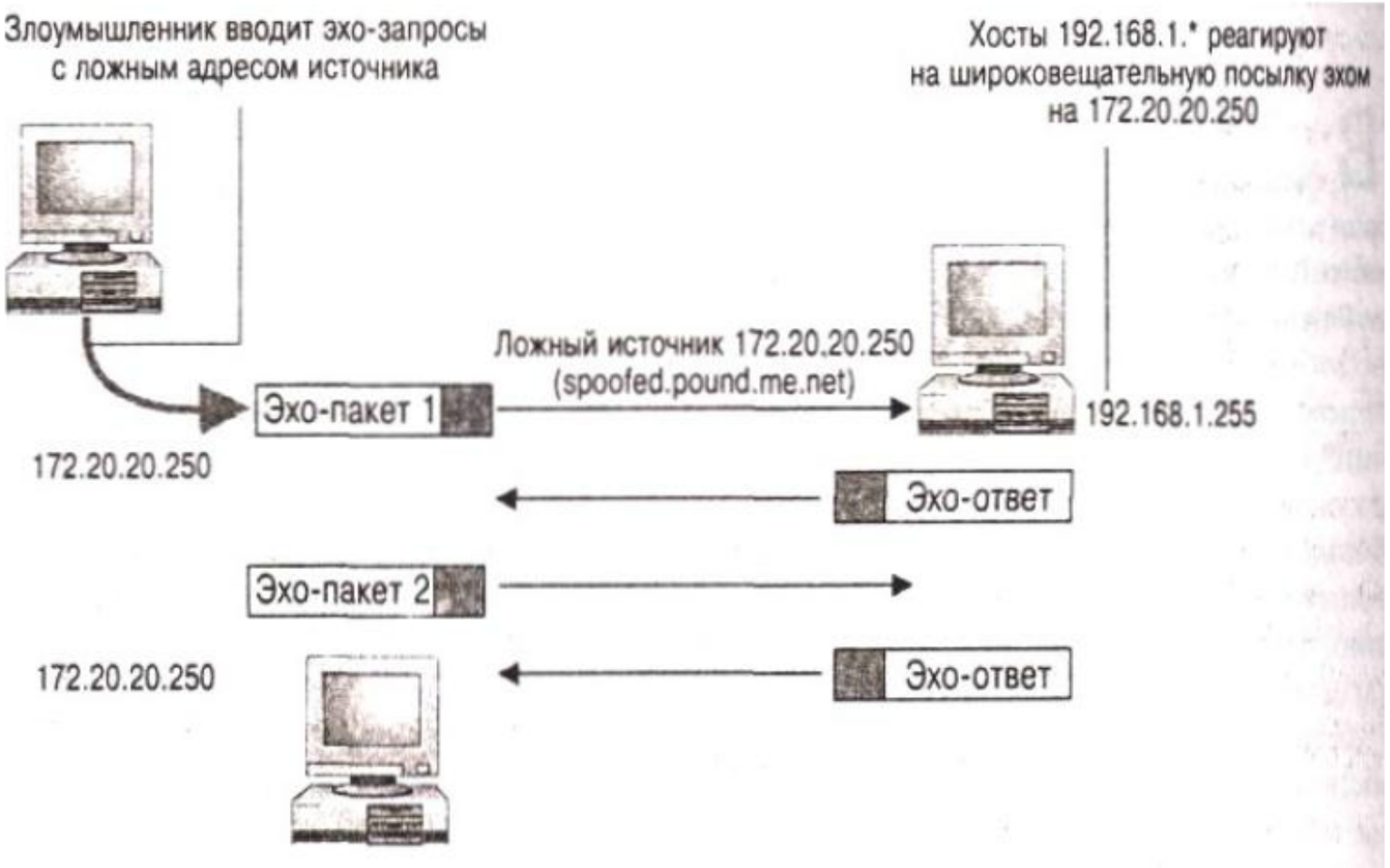
Атака Ping of Dead

`ping -l 65510 target ip`

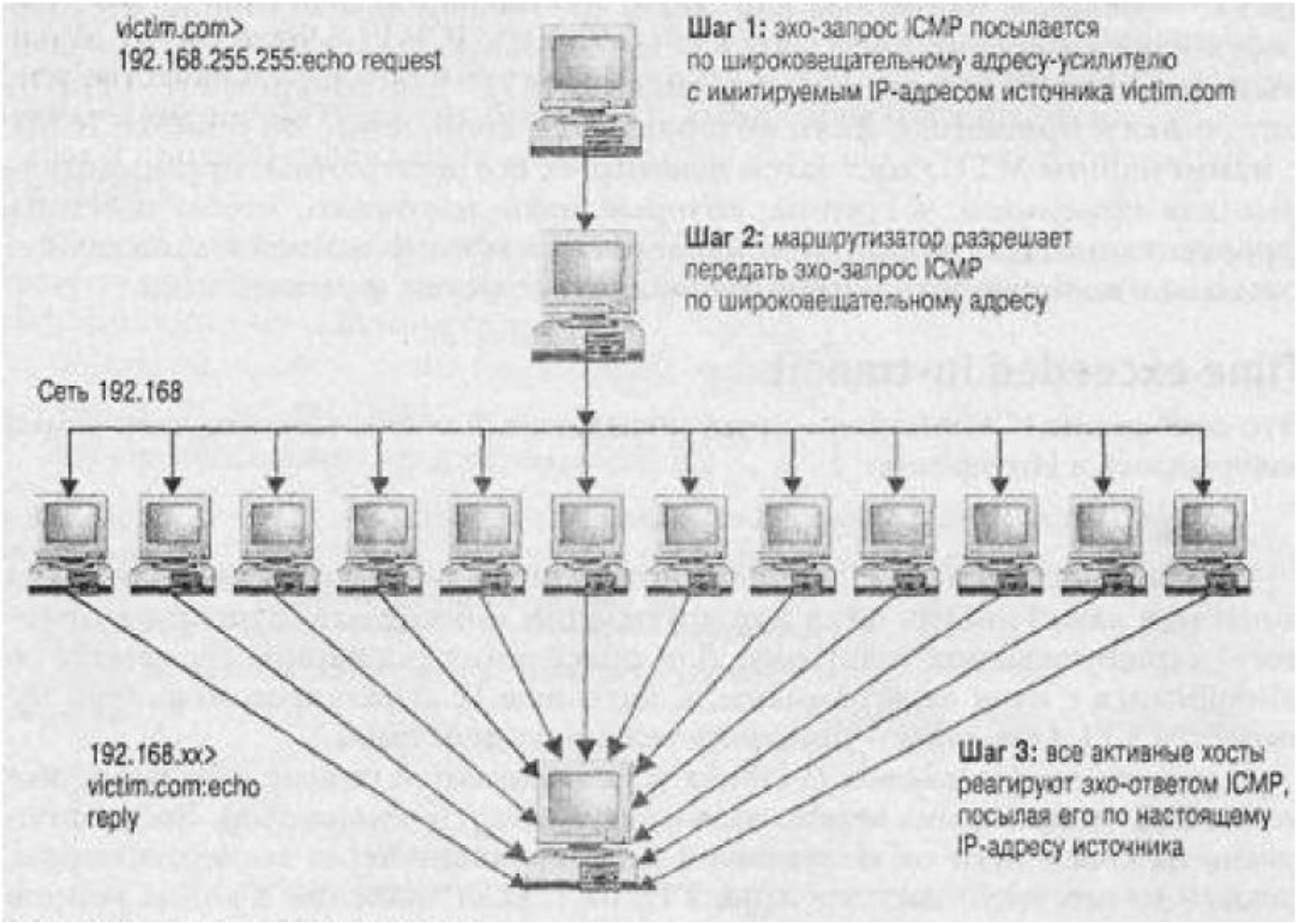
Цель — превысить максимально разрешенный размер пакета ICMP (65535 байтов). Если система не умеет обращаться с такими данными, происходит авария.

```
12:43:58.431 pinger> target icmp: echo request (frag 4321:380@0+)
12:43:58.431 pinger> target: (frag 4321 380@2656+)
12:43:58.431 pinger> target: (frag 4321 380@3040+)
12:43:58.431 pinger> target: (frag 4321 380@3416+)
12:43:58.431 pinger> target: (frag 4321 380@376+)
12:43:58.431 pinger> target: (frag 4321 380@3800+)
12:43:58.431 pinger> target: (frag 4321 380@4176+)
12:43:58.431 pinger> target: (frag 4321 380@760+)
...
12:43:58.491 pinger> target: (frag 4321:380@63080+)
12:43:58.491 pinger> target: (frag 4321:380@63456+)
12:43:58.491 pinger> target: (frag 4321:380@63840+)
12:43:58.491 pinger> target: (frag 4321:380@64216+)
12:43:58.491 pinger> target: (frag 4321:380@64600+)
12:43:58.491 pinger> target: (frag 4321:380@64976+)
12:43:58.491 pinger> target: (frag 4321:380@65360)
```

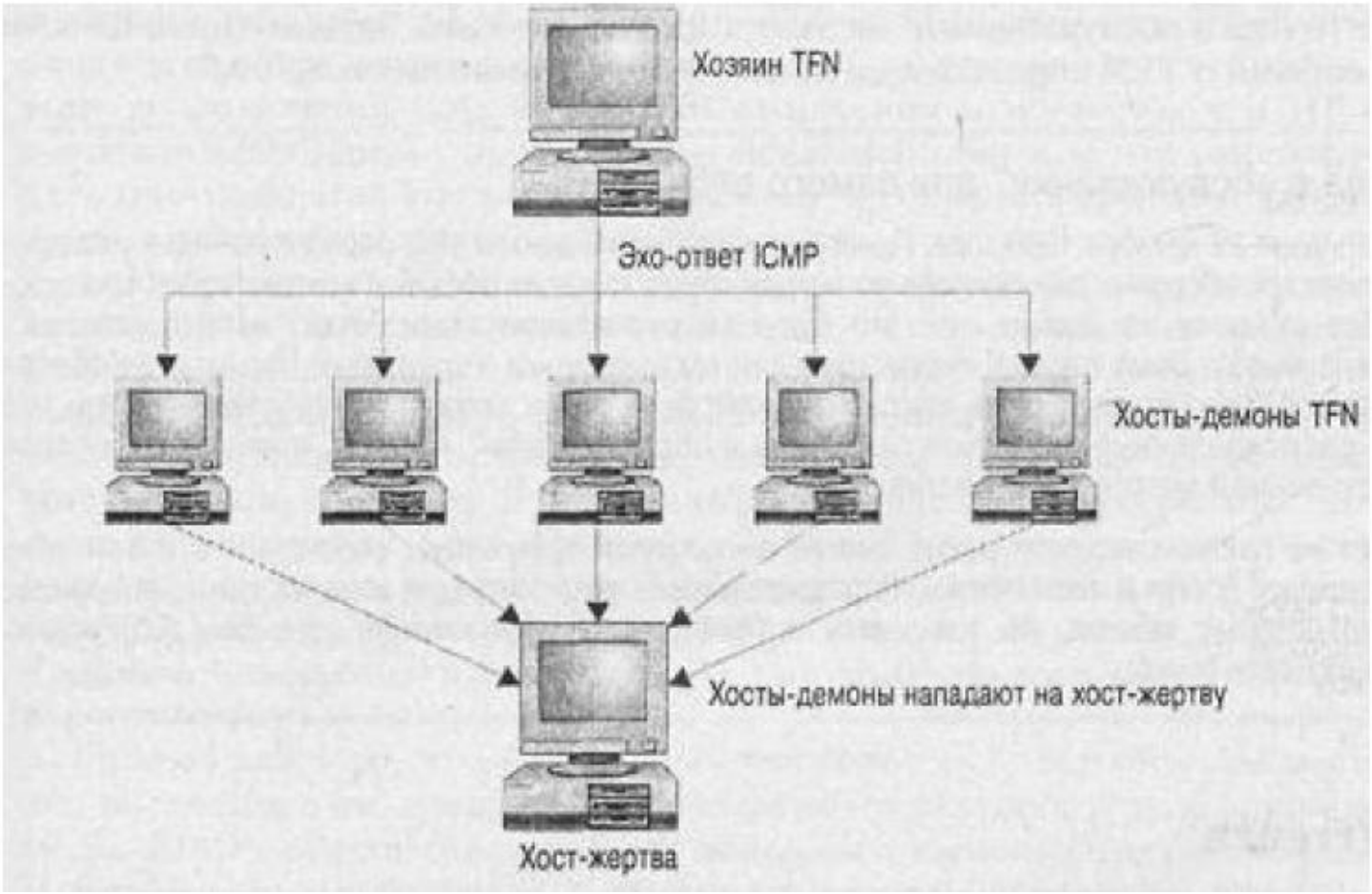
Атака Smurf с использованием сети-усилителя



Атака Smurf с использованием сети-усилителя



DDoS-атака с использованием бот-сети



Сценарий взлома сети с Windows-хостами

- Сетевая разведка в Internet. Выявление активных систем, почтовых адресов.
- Определение типа системы и средств безопасности при помощи Nmap
- Эксплойтинг с использованием зараженной Web-страницы или засылка «троянской» программы при помощи e-mail, подключение через нулевой сеанс…
- Сбор информации о системе, получение учётных записей и паролей пользователей
- Компрометация средств защиты, установка удалённого управления
- Установка скрытого канала
- Установка вредоносного программного обеспечения

Сценарий взлома сети с Unix-хостами

- Сетевая разведка в Internet. Выявление активных систем, почтовых адресов.
- Идентификация системы UNIX. и средств безопасности при помощи Nmap
- Сканирование портов при помощи Nmap и определение приложений, использующих порты с использованием программы Netcat
- Эксплойтинг хоста – поиск эксплойта для активного приложения, его загрузка и компиляция.
- Получение shell (доступа к командной оболочке)
- Создание учётной записи, повышение привилегий
- Компрометация средств защиты
- Установка троянской программы, например, подмена утилит уровня ядра knark.
- Установка скрытого канала
- Установка вредоносного программного обеспечения

Терминология

- **Уязвимость** используется для обозначения недостатка в системе, используя который, можно нарушить её целостность и вызвать неправильную работу. Уязвимость может быть случайной или намеренной.
- **Эксплойт, эксплоит** — это компьютерная программа, фрагмент программного кода или последовательность команд, использующие уязвимости в программном обеспечении и применяемые для проведения атаки на вычислительную систему.
- **Шелл-код** (англ. shellcode) — это двоичный исполняемый код, внедряемый в программу и исполняемый в ходе атаки на обнаруженную в ней уязвимость (обычно для организации перехвата управления). Как правило он очень ограничен по размеру (размером пакета, размером буфера уязвимого приложения) и предназначен для выполнения строго определенной задачи наиболее эффективным способом. Эффективностью шелл-кода можно пожертвовать ради обеспечения его функциональной гибкости.
- **Переполнение буфера** (Buffer Overflow) — один из типов уязвимостей, возникающих, когда компьютерная программа записывает данные за пределами выделенного в памяти буфера.
- Переполнение буфера обычно возникает из-за неправильной работы с данными, полученными извне, и памятью, при отсутствии жесткой защиты со стороны языка программирования, подсистемы программирования (компилятор или интерпретатор) и операционной системы. В результате переполнения могут быть испорчены или перезаписаны данные, расположенные следом за буфером (или перед ним).

Shell-код

- **Шелл-код** (англ. *shellcode*, код запуска оболочки) — это двоичный исполняемый код, который обычно передаёт управление командному процессору, например '/bin/sh' Unix shell, command.com в MS-DOS и cmd.exe в операционных системах Microsoft Windows. Шелл-код может быть использован как полезная нагрузка эксплойта, обеспечивая взломщику доступ к командной оболочке (англ. *shell*) в компьютерной системе.

- **Типы shell-кодов:**
  - Shell-код для привязки к порту (англ. *port binding shellcode*).
  - Shell-код для использования существующего дескриптора сокета.
  - Shell-код, выполняющий системный вызов.
  - Shell-код для обратного соединения (англ. *reverse shell shellcode*).
  - Кросплатформенный shell-код.

- Шелл-код обычно внедряется в память эксплуатируемой программы, после чего на него передается управление путём переполнения стека, или при переполнении буфера в куче, или используя атаки форматной строки. Передача управления шелл-коду осуществляется перезаписью адреса возврата в стеке адресом внедрённого шелл-кода, перезаписью адресов вызываемых функций или изменением обработчиков прерываний. Результатом этого является выполнение шелл-кода, который открывает командную строку для использования взломщиком.

Пример shell-кода

```
char shellcode[] =

"x33xc9x83xe9xebxd9xeexd9x74x24xf4x5bx81x73x13x8a"

"xd4xf2xe7x83xebxfcxe2xf4xbbx0fxa1xa4xd9xbexf0x8d"

"xecx8cx6bx6ex6bx19x72x71xc9x86x94x8fx9bx88x94xb4"

"x03x35x98x81xd2x84xa3xb1x03x35x3fx67x3axb2x23x04"

"x47x54xa0xb5xdcx97x7bx06x3axb2x3fx67x19xbexf0xbe"

"x3axebx3fx67xc3xadx0bx57x81x86x9axc8xa5xa7x9ax8f"

"xa5xb6x9bx89x03x37xa0xb4x03x35x3fx67";
```

Классификация эксплоитов

Удалённый эксплойт (*remote*) - работает через сеть и использует уязвимость в защите без какого-либо предварительного доступа к уязвимой системе;

Локальный эксплойт (*local*) - запускается непосредственно в уязвимой системе, требуя предварительного доступа к ней. Обычно используется для получения взломщиком прав суперпользователя.

Типы уязвимостей, которые используют эксплойты:

- уязвимости форматной строки:
- уязвимости сетевых протоколов;
- уязвимости возникающие в состоянии «Гонки»;
- ошибки при работе со стеком, позволяющие осуществить переполнение;
- ошибки при работе с кучей, позволяющие осуществить затирание;
- ошибки при работе с целыми числами.

Назначение зксплойтов:

1. Эксплойты для операционных систем.
2. Эксплойты для прикладного ПО (музыкальные проигрыватели, офисные пакеты).
3. Эксплойты для браузеров (Internet Explorer, Mozilla Firefox, Opera и др.).
4. Эксплойты для интернет-продуктов (IPB, WordPress, VBulletin, phpBB).
5. Эксплойты для интернет-сайтов (facebook.com, hi5.com, livejournal.com).
6. Другие эксплойты.

Использование уязвимостей

- Эксплойты могут быть классифицированы по типу используемой ими уязвимости, такой как: переполнение буфера, SQL-инъекция, межсайтовый скриптинг, подделка межсайтовых запросов и т. д.

- Эксплойты, как правило, нацелены на уязвимости в конкретных целевых системах, поэтому к эксплойту, как правило, прилагается список версий ОС и оборудования, на которых он работает.

- Связки (сборки) эксплойтов представляют из себя пакет эксплойтов сразу под несколько программ (версий) и/или под разные уязвимости в них. В последних версиях связок производится выбор эксплойта именно под конкретную программу пользователя.

- Особой популярностью пользуются так называемые 0-day эксплойты (эксплойты «нулевого дня»), использующие недавно появившиеся уязвимости, которые еще не стали известны общественности

Инструменты для исследования программ и написании shell-кодов и эксплойтов

- **IDA Pro.** Самый лучший дизассемблер в мире. На данный момент его еще никто не смог обойти по функциональности.
- **Soft-Ice.** Входит в число лучших отладчиков режима ядра. При работе с данным отладчиком можно не ограничиваться отладкой пользовательских приложений, но и отлаживать ядерные компоненты. (Устаревший, но как говорится надежда умирает последней).
- **Kernel Debugger.** Мощный отладчик уровня ядра, входящий в пакет разработки драйверов ядра DDK (Device Driver Kit).
- **OllyDbg.** Отличный отладчик с большим функционалом, множеством пагинов и настроек.
- **Metasploit Framework.** Исключительная в своем роде платформа, с открытым исходным кодом, для создания, тестирования и использования эксплойтов.
- **Nasm.** Этот пакет содержит ассемблер nasm и дизассемблер ndisasm.
- **Masm32.** Если сложить вместе ASM-компилятор и линкер от Microsoft, кучу библиотек и заголовочных файлов, целый ряд разных утилит, то мы получим пакет MASM32.
- **gdb.** GNU отладчик.
- **objdump.** Инструмент для дизассемблирования объектных файлов и получения из них важной информации.
- **ktrace.** Утилита, имеющаяся только для систем *BSD. Она позволяет трассировать выполнение выполнение системных вызовов.
- **strace.** Эта программа очень похожа на ktrace, она тоже позволяет трассировать все системные вызовы. В большинстве дистрибутивов Linux strace устанавливается по умолчанию.
- **readelf.** Эта программа позволяет получить разнообразную информацию о двоичном файле в формате ELF.

Переполнение буфера

Переполнение буфера является наиболее популярным способом взлома, так как большинство языков высокого уровня используют технологию «стекового кадра» — размещение данных в стеке процесса, смешивая данные программы с управляющими данными (в том числе адреса начала стекового кадра и адреса возврата из исполняемой функции).

Переполнение буфера может вызывать аварийное завершение или зависание программы, ведущее к *отказу обслуживания* (DoS). Отдельные виды переполнений позволяют загрузить и выполнить произвольный машинный код от имени программы и с правами учетной записи, от которой она выполняется.

Переполнение буфера

В случае если буфер расположен в стеке и стек "растёт вниз" (например в архитектуре x86), то с помощью переполнения буфера можно изменить адрес возврата выполняемой функции, так как адрес возврата расположен после буфера, выделенного выполняемой функцией. Тем самым есть возможность выполнить произвольный участок машинного кода в адресном пространстве процесса.

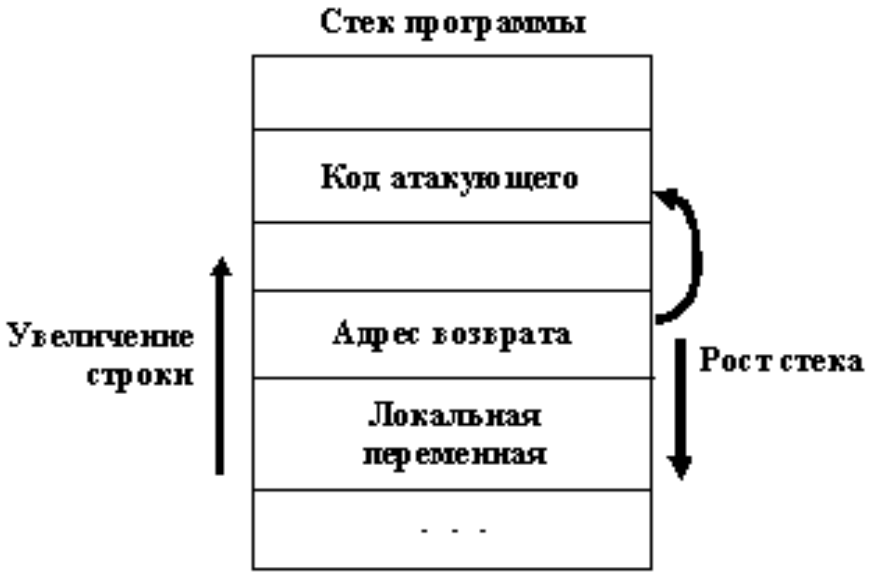


Схема атаки “срыв стека”

Пример переполнения буфера

Рассмотрим гипотетическую программу, которая исполняется с привилегиями суперпользователя и выполняет некоторые функции - например, изменение пароля пользователя. Если программа не проверяет длину введённого нового пароля, то любые данные, длина которых превышает размер выделенного для их хранения буфера, будут просто записаны поверх того, что находилось после буфера.

```
int namelen (void) {
char pass[21];
gets(name);
return strlen(pass);
}
```

При вводе пароля размером более 20 символов частью строки будет замещен адрес возврата из функции. Далее, при выполнении инструкции возврата из подпрограммы, управление будет передано по адресу, который образуют соответствующие позиции введенной строки и в обычной ситуации будет получено сообщение об ошибке операционной системы.