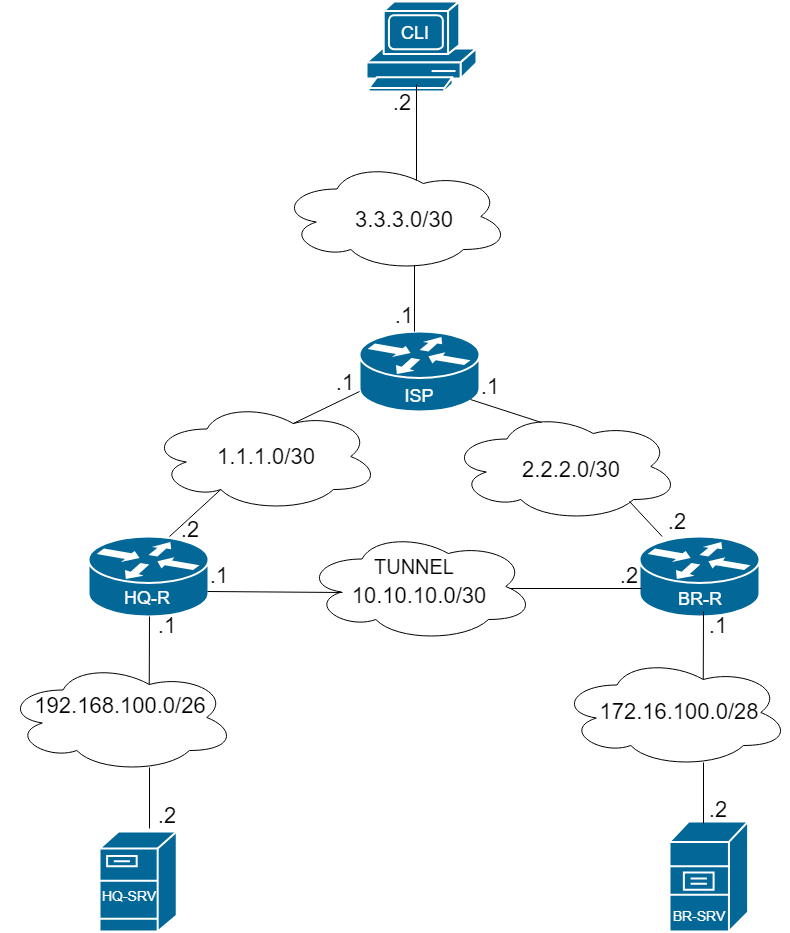
**Модуль 1**

1.

|  |  |  |  |
| --- | --- | --- | --- |
| **Имя устройства** | **IPv4** | **IPv6** | **NIC** |
| CLI | 3.3.3.2/30 | 2001:3::2/64 | ISP-CLI |
| ISP | 1.1.1.1/30 | 2001:1::1/64 | ISP-HQ |
| 2.2.2.1/30 | 2001:2::1/64 | ISP-BR |
| 3.3.3.1/30 | 2001:3::1/64 | ISP-CLI |
| HQ-R | 1.1.1.2/30 | 2001:1::2/64 | ISP-HQ |
| 192.168.100.1/27 | 2000:192::1/123 | HQ |
| 10.10.10.1/30 | 2001:10::1/64 | TUNNEL |
| HQ-SRV | 192.168.100.2/27(DHCP) | 2000:192::2/123(DHCP) | HQ |
| BR-R | 2.2.2.2/30 | 2001:2::2/64 | ISP-BR |
| 172.16.100.1/29 | 2000:172::1/125 | BR |
| 10.10.10.2/30 | 2001:10::2/64 | TUNNEL |
| BR-SRV | 172.16.100.2/29 | 2000:172::2/125 | BR |
| HQ-CLI | (DHCP) | (DHCP) | HQ |
| HQ-AD | 192.168.100.10/27 | 2000:192::10/123 | HQ |



Топология сети L3 IPV4

№1 Имена: HQBR-R conf – hostname (Имя) – com - con; HQBR-SRVCLI (СLI) hostnamectl set-hostname (Имя) – exec bash

ip-адресация: CLI параметры соединения - ipV4 – вручную – вводим ip, маску, шлюз(ISP-CLI), DNS (8.8.8.8)

HQBR-R: Conf – int gi (1/0/12) – ip add (1.1.1.2/30) – ip firewall disable – com – con

BR-SRV: echo (172.16.100.2/28) > /etc/net/ifaces/ens192/ipv4address –

еcho via default (172.16.100.1) > /etc/net/ifaces/ens192/ipv4route –

+днс (HQ-SRV) echo nameserver (192.168.100.2) > /etc/net/ifaces/ens192/resolv.conf

Systemctl restart network ip a

№2 Nat(динам марш): HQBR-R conf – object-group network LOCAL\_NET – ip add-range (192.168.100.1-192.168.100.62172.16.100.1-172.16.100.14) - ex – object-group network PUBLIC\_POOL – ip add (1.1.1.22.2.2.2) – ex – nat source – pool TRANSLATE\_ADDRESS – ip add (1.1.1.22.2.2.2) – ex – ruleset SNAT – to int gi 1/0/1 – rule 1 - match source-add LOCAL\_NET – action source-nat pool TRANSLATE\_ADDRESS – enable – end - conf – ip route 0.0.0.0/0 (1.1.1.12.2.2.1)– ex – com – con (проверка ping 8.8.8.8, 8.8.8.8 source ip (192.168.100.1172.16.100.1), sh ip nat tr

Туннель: HQBR-R conf – tunnel gre 10 – ip firewall disable – local add (1.1.1.22.2.2.2) – remote add (2.2.2.21.1.1.2) – ip add (10.10.10.12/30) – mtu 1426 – ttl 18 – enable – end – com – con (проверка ping 10.10.10.12)

Ospf: HQBR-R conf – router ospf 10 – router-id (1.1.1.12.2.2.2) – area 1.1.1.1 – network (192.168.**100.0/27172**.16.100.0/28)– enable – ex – enable – ex – tunnel gre 10 - ip ospf instance 10 - ip ospf area 1.1.1.1 – ip ospf – end – com – con (проверка sh ip ospf neig, sh ip route)

эффективно управлять большими и сложными сетями, обеспечивая быструю сходимость OSPF топ выбор для масштабирования

№3 DHCP(респредел): HQ-R conf – ip dhcp-server - ip dhcp-server pool LAN\_HQ - network (192.168.100.0/27) – add-range (192.168.100.1-192.168.100.30) – excluded-add-range (192.168.100.1) - excluded-add-range (192.168.100.10) – add (192.168.100.2) mac-add(находится на HQ\_SRV ip) (00:0c:29:ef:2d:9b) - default-router 192.168.100.1 – dns-server 192.168.100.2 – end – com – con

HQ-SRVCLI: vim /etc/net/ifaces/ens192/options (меняем на dhcp); dhcpcd (проверка ip a)

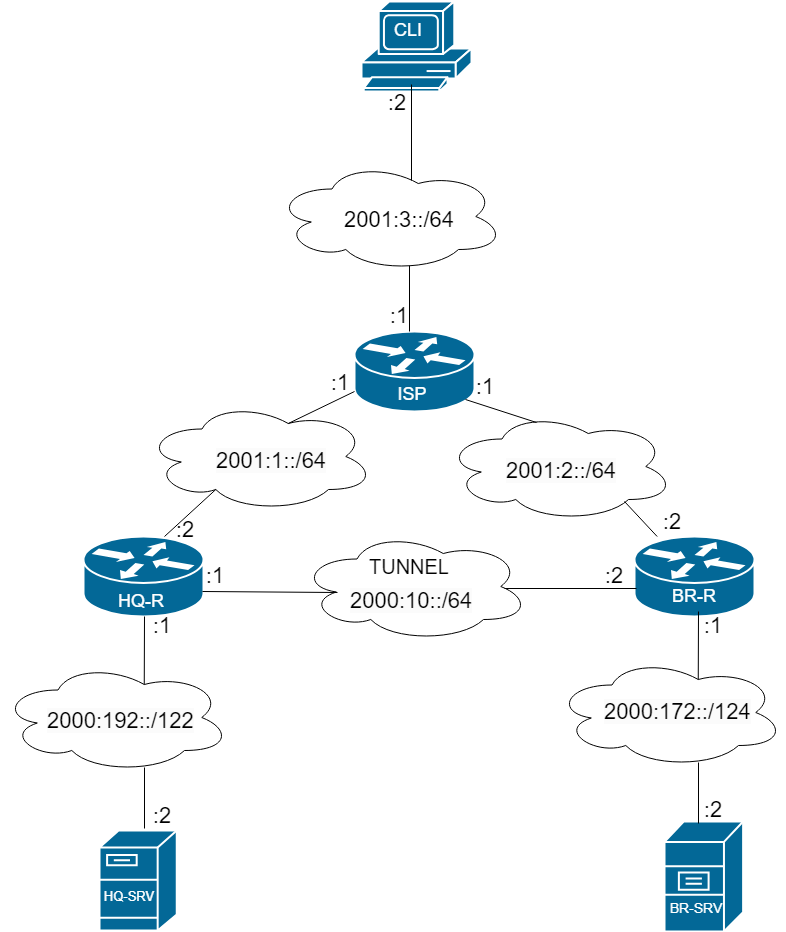
№4 Учетки: CLI меню – центр управления; HQ-SRV: useradd admin -c “Admin” – passwd admin – P@ssw0rd – usermod -aG wheel admin (проверка cat /etc/passwd); HQ-R: conf – username admin – password P@ssw0rd – end – com – con; BR-SRV: useradd branch-admin -c ‘’Branch-admin” - passwd branch-admin – P@ssw0rd; BR-R: conf – username branch-admin – password P@ssw0rd – privilege 15 – end – com – con; BRHQ-R: conf – username network-admin – password P@ssw0rd – privilege 15 – end – com – con; BR-SRV: useradd network-admin -c ‘’Network-admin” - passwd network-admin – P@ssw0rd

№5 Прпуск iperf3: CLI su – apt-get update – apt-get install iperf3 (проверка от CLI>ISP iperf3 -c 3.3.3.1; от ISP>CLI iperf3 -R -c 3.3.3.1 + 2 скрины iperf3- измеряет пропускную способность сети с CLI(3.3.3.2)>ISP(3.3.3.1) обьем за 10сек \*гб средняя способность \*гб В режиме реверс -R

№6 Backup: HQBR-R conf – archive – type local – count-backup 30 – time-period 1440 – by-commit – end – com – con (проверка dir flash:bachup) 2 скрина

№7 SSH: HQ-R conf – object-group service SRV\_SSH – port 2222 - ex– object-groupe network SERVER\_IP – ip add 192.168.100.2 – ex – nat destination – pool SERVER\_POOL – ip add 192.168.100.2 – ip port 22 – ex – rulest DNAT – ftom int gi 1/0/1 – rule 1 – match destination-add PUBLIC\_POOL – match protocol tcp – match destination-port SRV\_SSH – action destination-nat pool SERVER\_POOL – enabe – end – com – con

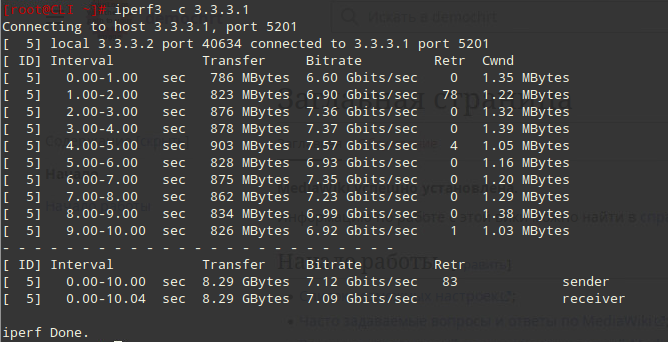
№8 Запрет: HQ-R conf – ip access-list extended NO\_ACCESS\_SSH\_CLI – rule 1 – action deny – match protocol tcp – match source-address 3.3.3.2 255.255.255.252 – match destination-port 2222 – enable – ex – rule 2 - action permit – enable – ex – ex – int gi 1/0/1 – server-acl input N0\_ACCESS\_SSH\_CLI – end – com - con



Топология сети L3 IPv6

2. Выбор OSPF в качестве протокола динамической маршрутизации для растущей сети обоснован его способностью эффективно управлять большими и сложными сетями, обеспечивая быструю сходимость, гибкость в использовании адресного пространства, поддержку различных топологий и устройств от разных производителей. Эти преимущества делают OSPF отличным выбором для сетей, которые планируется масштабировать в будущем.

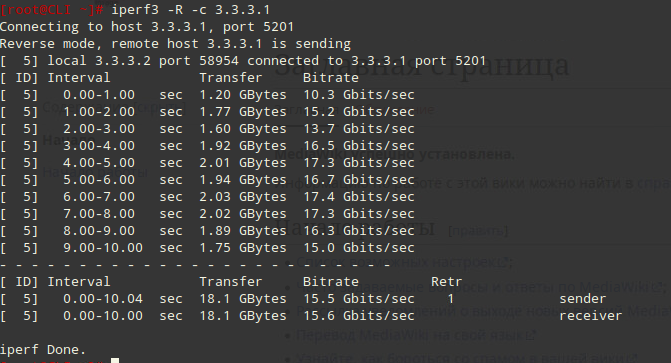
5.



В приведённом скриншоте показан результат работы утилиты iperf3, которая используется для измерения пропускной способности сети. Команда была выполнена на локальной машине CLI (адрес 3.3.3.2), соединённой с удалённым сервером ISP (адрес 3.3.3.1) по порту 5201.

Описание результатов:

* Общий объём переданных данных за 10 секунд составил 8.29 Гбайт.
* Средняя пропускная способность составила 7.12 Гбит/сек.



На предоставленном скриншоте показан результат работы утилиты iperf3 в режиме реверса (-R), что означает, что сервер ISP (хост 3.3.3.1) отправляет данные клиенту CLI (хост 3.3.3.2). Это позволяет измерить пропускную способность канала от сервера к клиенту.

Описание результатов:

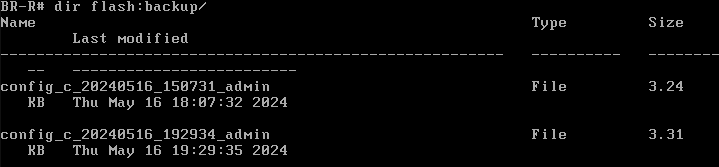
* Общий объём переданных данных за 10 секунд составил 18.1 Гбайт.
* Средняя пропускная способность составила 15.5 Гбит/сек.

6. Резервное копирование выполняется ежедневно, а также по команде COMMIT.

**HQ-R**



**BR-R**



**Модуль 2**

3. Выбор Samba DC в качестве контроллера домена представляет собой рациональное и экономически выгодное решение, которое обеспечивает высокую совместимость, безопасность и гибкость. Это позволяет организациям эффективно управлять своими сетевыми ресурсами без значительных затрат на программное обеспечение и лицензирование.

4. Использование файлового сервера на базе SMB является оптимальным решением для большинства корпоративных и домашних сетей благодаря его совместимости, простоте настройки, высоким показателям безопасности и производительности. Эти преимущества делают SMB подходящим выбором для эффективного и безопасного обмена файлами в разнообразных сетевых средах.

MediaWiki

MediaWiki is a free and open-source wiki software package written in PHP. It serves as the platform for Wikipedia

and the other Wikimedia projects, used by hundreds of millions of people each month. MediaWiki is localised in over

350 languages and its reliability and robust feature set have earned it a large and vibrant community of third-party

users and developers.

MediaWiki is:

feature-rich and extensible, both on-wiki and with hundreds of extensions;

scalable and suitable for both small and large sites;

simple to install, working on most hardware/software combinations; and

available in your language.

For system requirements, installation, and upgrade details, see the files RELEASE-NOTES, INSTALL, and UPGRADE.

Ready to get started?

https://www.mediawiki.org/wiki/Special:MyLanguage/Download

Setting up your local development environment?

https://www.mediawiki.org/wiki/Local\_development\_quickstart

Looking for the technical manual?

https://www.mediawiki.org/wiki/Special:MyLanguage/Manual:Contents

Seeking help from a person?

https://www.mediawiki.org/wiki/Special:MyLanguage/Communication

Looking to file a bug report or a feature request?

https://bugs.mediawiki.org/

Interested in helping out?

https://www.mediawiki.org/wiki/Special:MyLanguage/How\_to\_contribute

MediaWiki is the result of global collaboration and cooperation. The CREDITS file lists technical contributors

to the project. The COPYING file explains MediaWiki's copyright and license (GNU General Public License, version

2 or later). Many thanks to the Wikimedia community for testing and suggestions.

№1 Имена; ip-адресация

HQ-R,BR-R: conf>hostname(имя)>com>con

HQ-SRV,BR,SRV,HQ-CLI,CLI: hostnamectl set-hostname(имя)>exec bash

CLI параметры>вручную>шлюз(ISP-CLI)>DNS8.8.8.8 (проверка ip a,ya.ru)

HQ-R: conf>int gi 1/0/1>ip add 1.1.1.2/30>ip firewall disable>com>con (проверка 1.1.1.1)

HQ-R: conf>int gi 1/0/2>ip add 192.168.100.1/27>ip firewall disable>com>con

BR-R: conf>int gi 1/0/1>ip add 2.2.2.2/30>ip firewall disable>com>con

BR-R: conf>int gi 1/0/2>ip add 172.16.100.1/29>ip firewall disable>com>con (проверка2.2.2.2)

BR-SRV: echo 172.16.100.2/29 > /etc/net/ifaces/ens192/ipv4address>

еcho default via 172.16.100.1 > /etc/net/ifaces/ens192/ipv4route>

+днс(HQ-SRV) echo nameserver 192.168.100.2 > /etc/net/ifaces/ens192/resolv.conf>

Systemctl restart network>ip a

Обзор

Протокол NTP (англ. Network Time Protocol — протокол сетевого времени) — сетевой протокол синхронизирующий

хронометраж времени между распределенными серверами времени и клиентами . С помощью этой синхронизации возможна

активация лицензий на устройстве, ведение системных журналов с сопоставлением по времени и прочих событий, возникающих на устройстве

NTP использует UDP в качестве транспортного протокола с портом 123. Для всех сообщений NTP используется UTC

(англ. Universal Coordinated Time) - всемирное координированное время , которое совпадает со средним временем по Гринвичу.

Устройства линейки ESR могут не синхронизировать время с другими хостами, время которых может быть неоднозначным

или некорректным по следующим причинам:

если устройство, с которым ESR необходимо синхронизироваться , само не синхронизировано

если устройство, с которым ESR необходимо синхронизироваться имеет значительно отличающееся время, даже если

его stratum ниже (лучше), чем у других.

если ESR не синхронизирован с другими серверами времени

Далее по тексту будут использоваться такие понятия как синхронизировать и синхронизироваться. В первом случае

понятие подразумевает подвергнуть синхронизации устройства согласно переданным им данных времени (временные метки).

Во втором случае устройство само будет синхронизировано по параметрам, полученным от другого устройства.

Устройства Eltex ESR поддерживают следующие режимы ассоциаций:

Client

Server

Symmetric active

Symmetric passive

Broadcast client

В симметричном режиме оба устройства выступают равноправными пирами, обменивающимися данными хронометража и, в зависимости

от этих данных, синхронизируют время с пиринговой машиной или же сами синхронизируют её.

Хост, работающий в симметричном активном режиме отправляет периодические сообщения независимо от состояния доступности или

уровня своего однорангового узла-соседа. Работая в этом режиме, хост объявляет о своей готовности к синхронизации и быть

синхронизированным одноранговым узлом.

Симметричный пассивный режим активируется при поступление сообщения о работе в симметричном активном режиме соседнего

однорангового узла, и сохраняется только до тех пор, пока одноранговый узел доступен и работает на уровне stratum, меньшем

или равном хосту; в противном случае связь разрывается. Однако связь всегда будет сохраняться до тех пор, пока в ответ не

будет отправлено хотя бы одно сообщение. Работая в этом режиме, хост объявляет о своей готовности к синхронизации и будет

синхронизирован одноранговым узлом.

Как правило, настройка симметричного режима работы осуществляется между устройствами, имеющие синхронизацию с точным времени

внешних серверов времени.

В контексте настроек ESR симметричный режим настраивается с помощью команды ntp peer <ip-address> . Чтобы между двумя устройствами

ESR установился симметричный активный режим, необходимо на каждом из устройств настроить данные команды. Команды указания интервала

отправки NTP меток соответствуют тем же, что и при режиме работы с сервером:

№2 Динамическая маршрутизация NAT,Тунуль,OSPF

HQ-R: conf>object-group network LOCAL\_NET>ip add-range 192.168.100.1-192.168.100.30>ex>

object-group network PUBLIC\_POOL>ip add 1.1.1.2>ex>nat source>pool TRANSLATE\_ADDRESS>

ip add 1.1.1.2>ex>ruleset SNAT>to int gi 1/0/1>rule 1>match source-add LOCAL\_NET>

action source-nat pool TRANSLATE\_ADDRESS>enable>end>conf>ip route 0.0.0.0/0 1.1.1.1>ex>

com>con (проверка ping8.8.8.8 source ip 192.168.100.1,sh ip nat tra)

BR-R: conf>object-group network LOCAL\_NET>ip add-range 172.16.100.1-172.16.100.6>ex>

object-group network PUBLIC\_POOL>ip add 2.2.2.2>ex>nat source>pool TRANSLATE\_ADDRESS>

ip add 2.2.2.2>ex>ruleset SNAT>to int gi 1/0/1>rule 1>match source-add LOCAL\_NET>

action source-nat pool TRANSLATE\_ADDRESS>enable>end>conf>ip route 0.0.0.0/0 2.2.2.1>ex>

com>con (проверка ping8.8.8.8 source ip 192.168.100.1,sh ip nat tra)

HQ-R: conf>tunnel gre 10>ip firewall disable>local add 1.1.1.2>remote add 2.2.2.2>

ip add 10.10.10.1/30>mtu 1426>ttl 18>enable>end>com>con (проверка ping 10.10.10.1)

BR-R: conf>tunnel gre 10>ip firewall disable>local add 2.2.2.2>remote add 1.1.1.2>

ip add 10.10.10.2/30>mtu 1426>ttl 18>enable>end>com>con (проверка ping 10.10.10.2)

HQ-R: conf>router ospf 10>router-id 1.1.1.1>area 1.1.1.1>network 192.168.100.0/27>

enable>ex>enable>ex>tunnel gre 10>ip ospf instance 10>ip ospf area 1.1.1.1>ip ospf>

end>com>con (проверка sh ip ospf neig, sh ip route)

BR-R: conf>router ospf 10>router-id 2.2.2.2>area 1.1.1.1>network 172.16.100.0/29>

enable>ex>enable>ex>tunnel gre 10>ip ospf instance 10>ip ospf area 1.1.1.1>ip ospf>

end>com>con (проверка sh ip ospf neig, sh ip route) эффективно управлять большими

и сложными сетями, обеспечивая быструю сходимость OSPF топ выбор для масштабирования

Механизмы защиты NTP

Работа в Virtual Routing and Forwarding

На устройствах линейки ESR имеется возможность задания VRF, в котором устройство будет устанавливать

Peer-to-Peer или Client-Server (в качестве Client) отношения. Для этого необходимо использовать команду

vrf <INSTANCE> после указания адреса удалённой устройства времени:

При этом, все устройства, выступающие в роли NTP-клиентов для данного ESR, и находящиеся в других VRF,

смогут получать синхронизацию времени. Значение таймштампов, отправляемое ESR NTP-клиентам, определяет

маршрутизатор на основании полученных значений от NTP-устройств, с которыми настроены Peer-to-Peer или

Client-Server (ESR - в качестве Client) соседства.

Для защиты NTP на устройствах линейки ESR необходимо использовать NTP object-group.

Данный функционал ограничивает NTP-запросы в сторону NTP-сервера и позволяет фильтровать NTP-сообщения

от несанкционированных устройств.

Режим работы serve-only подразумевает обработку и ответ на NTP-query ( запрос времени ) от NTP-клиентов

и одноранговых пиров. При этом сам NTP-сервер не будет пытаться синхронизироваться ( Selection Algorithm

- RFC5905 ) с временем, полученным в запросе от однорангового пира. Так же в данном режиме запрещены control

message.

Режим работы query-only подразумевает обработку и ответ на NTP-query ( запрос времени ) от NTP-клиентов и

одноранговых пиров. При этом сам NTP-сервер не будет пытаться синхронизироваться ( Selection Algorithm -

RFC5905 ) с временем, полученным в запросе от однорангового пира, но будет отвечать на control message.

№3 автоматическое распределение DHCP

HQ-R: conf>ip dhcp-server>ip dhcp-server pool LAN\_HQ>network 192.168.100.0/27>

add-range 192.168.100.1-192.168.100.30>excluded-add-range 192.168.100.1>

excluded-add-range 192.168.100.10>

add 192.168.100.2 mac-add(находится на HQ\_SRV ip) (00:0c:29:ef:2d:9b ip a)>

default-router 192.168.100.1>dns-server 192.168.100.2>end>com>con

HQ-SRV: vim /etc/net/ifaces/ens192/options (меняем на dhcp); dhcpcd (проверка ip a)

HQ-CLI: vim /etc/net/ifaces/ens192/options (меняем на dhcp); dhcpcd (проверка ip a)

Задача:

Необходимо установить client-server ассоциации между ESR и NTP Servers из определённого

пула адресов для получения времени и распространения его в LAN-сегмент. NTP-клиенты в LAN

сегменте должны лишь получать NTP-reply на запросы времени NTP-сервера от ESR, не пытаясь

синхронизировать его. В LAN-сегменте имеется так же сервер для мониторинга NTP-сервера

внутри периметра.

Решение:

Предвариетльная настройка

Предварительно нужно выполнить следующие действия:

создать зону безопасности

описать взаимодействие между созданной зоной и зоной self

указать зону безопасности для интерфейса gi1/0/1;

настроить IP-адрес для интерфейса gi1/0/1, чтобы обеспечить IP-связность с NTP-сервером.

OSPF (англ. Open Shortest Path First) — протокол динамической маршрутизации, основанный

на технологии отслеживания состояния канала (link-state technology) и использующий для

нахождения кратчайшего пути алгоритм Дейкстры.

Протокол OSPF был разработан IETF в 1988 году. Последняя версия протокола представлена в

RFC 2328 (1998 год). Протокол OSPF представляет собой протокол внутреннего шлюза (Interior

Gateway Protocol — IGP). Протокол OSPF распространяет информацию о доступных маршрутах между

маршрутизаторами одной автономной системы.

OSPF имеет следующие преимущества:

Высокая скорость сходимости по сравнению с дистанционно-векторными протоколами маршрутизации;

Поддержка сетевых масок переменной длины (VLSM);

Оптимальное использование пропускной способности с построением дерева кратчайших путей

№4 учетки

CLI: меню>центр управления>управление системой>локальные>Admin

HQ-SRV: useradd admin -c “Admin”>passwd admin>P@ssw0rd>

usermod -aG wheel admin (проверка cat /etc/passwd)

HQ-R: conf>username admin>password P@ssw0rd>end>com>con

BR-SRV: useradd branch-admin -c "Branch-admin”>passwd branch-admin>P@ssw0rd;

BR-R: conf>username branch-admin>password P@ssw0rd>privilege 15>end>com>con;

BR-R: confus>ername network-admin>password P@ssw0rd>privilege 15>end>com>con;

HQ-R: confus>ername network-admin>password P@ssw0rd>privilege 15>end>com>con

BR-SRV: useradd network-admin -c "Network-admin”>passwd network-admin>P@ssw0rd>

usermod -aG wheel network-admin

DNS (англ. Domain Name System «система доменных имён») — компьютерная распределённая

система для получения информации о доменах. Чаще всего используется для получения IP-адреса

по имени хоста (компьютера или устройства), получения информации о маршрутизации почты и/или

обслуживающих узлах для протоколов в домене (SRV-запись).

Распределённая база данных DNS поддерживается с помощью иерархии DNS-серверов, взаимодействующих

по определённому протоколу.

Основой DNS является представление об иерархической структуре имени и зонах. Каждый сервер, отвечающий

за имя, может передать ответственность за дальнейшую часть домена другому серверу (с административной

точки зрения — другой организации или человеку), что позволяет возложить ответственность за актуальность

информации на серверы различных организаций (людей), отвечающих только за «свою» часть доменного имени.

Начиная с 2010 года в систему DNS внедряются средства проверки целостности передаваемых данных, называемые

DNS Security Extensions (DNSSEC). Передаваемые данные не шифруются, но их достоверность проверяется криптографическими

способами. Внедряемый стандарт DANE обеспечивает передачу средствами DNS достоверной криптографической информации (сертификатов),

используемых для установления безопасных и защищённых соединений транспортного и прикладного уровней.

DNS обладает следующими характеристиками:

Распределённость администрирования. Ответственность за разные части иерархической структуры несут разные люди или организации.

Распределённость хранения информации. Каждый узел сети в обязательном порядке должен хранить только те данные, которые входят в

его зону ответственности, и (возможно) адреса корневых DNS-серверов.

Кэширование информации. Узел может хранить некоторое количество данных не из своей зоны ответственности для уменьшения нагрузки на сеть.

Иерархическая структура, в которой все узлы объединены в дерево, и каждый узел может или самостоятельно определять работу нижестоящих узлов,

или делегировать (передавать) их другим узлам.

Резервирование. За хранение и обслуживание своих узлов (зон) отвечают (обычно) несколько серверов, разделённые как физически, так и логически,

что обеспечивает сохранность данных и продолжение работы даже в случае сбоя одного из узлов.

DNS важна для работы Интернета, так как для соединения с узлом необходима информация о его IP-адресе, а для людей проще запоминать

буквенные

(обычно осмысленные) адреса, чем последовательность цифр. В некоторых случаях это позволяет использовать виртуальные серверы,

например, HTTP-серверы,

различая их по имени запроса. Первоначально преобразование между доменными и IP-адресами производилось с использованием специального

текстового файла hosts,

который составлялся централизованно и автоматически рассылался на каждую из машин в своей локальной сети. С ростом Сети возникла

необходимость в эффективном,

автоматизированном механизме, которым и стала DNS.

DNS была разработана Полом Мокапетрисом в 1983 году; оригинальное описание механизмов работы содержится в RFC 882 и RFC 883.

В 1987 публикация RFC 1034 и RFC 1035

изменила спецификацию DNS и отменила RFC 882, RFC 883 и RFC 973 как устаревшие.

№5 Прпуск iperf3

CLI: su>apt-get update>apt-get install iperf3 (проверка от CLI>ISP iperf3 -c 3.3.3.1;

от ISP>CLI iperf3 -R -c 3.3.3.1 + 2 скрины iperf3- измеряет пропускную способность сети с

CLI(3.3.3.2)>ISP(3.3.3.1) обьем за 10сек \*гб средняя способность \*гб В режиме реверс -R

№6 Backup

HQ-R: conf>archive>type local>count-backup 30>time-period 1440>by-commit>end>com>con

BR-R:conf>archive>type local>count-backup 30>time-period 1440>by-commit>end>com>con

(проверка dir flash:bachup) 2 скрина

SSH (англ. Secure Shell — «безопасная оболочка»[1]) — сетевой протокол прикладного уровня,

позволяющий производить удалённое управление операционной системой и туннелирование TCP-соединений

(например, для передачи файлов). Схож по функциональности с протоколами Telnet и rlogin, но, в отличие

от них, шифрует весь трафик, включая и передаваемые пароли. SSH допускает выбор различных алгоритмов шифрования.

SSH-клиенты и SSH-серверы доступны для большинства сетевых операционных систем.

SSH позволяет безопасно передавать в незащищённой среде практически любой другой сетевой протокол. Таким образом,

можно не только удалённо работать на компьютере через командную оболочку, но и передавать по шифрованному каналу

звуковой поток или видео (например, с веб-камеры)[2]. Также SSH может использовать сжатие передаваемых данных для

последующего их шифрования, что удобно, например, для удалённого запуска клиентов X Window System.

Большинство хостинг-провайдеров за определённую плату предоставляет клиентам доступ к их домашнему каталогу по SSH.

Это может быть удобно как для работы в командной строке, так и для удалённого запуска программ (в том числе

графических приложений).

SSH — это протокол прикладного уровня. SSH-сервер обычно прослушивает соединения на TCP-порту 22. Спецификация протокола

SSH-2 содержится в RFC 4251. Для аутентификации сервера в SSH используется протокол аутентификации сторон на основе алгоритмов

электронно-цифровой подписи RSA или DSA, но допускается также аутентификация при помощи пароля (режим обратной совместимости с

Telnet) и даже ip-адреса хоста (режим обратной совместимости с rlogin).

Аутентификация по паролю наиболее распространена. При каждом подключении подобно https вырабатывается общий секретный ключ для

шифрования трафика.

При аутентификации по ключевой паре предварительно генерируется пара открытого и закрытого ключей для определённого пользователя.

На машине, с которой требуется произвести подключение, хранится закрытый ключ, а на удалённой машине — открытый. Эти файлы

не передаются

при аутентификации, система лишь проверяет, что владелец открытого ключа также владеет и закрытым. При данном подходе,

как правило,

настраивается автоматический вход от имени конкретного пользователя в ОС.

Аутентификация по ip-адресу небезопасна, эту возможность чаще всего отключают.

Для создания общего секрета (сеансового ключа) используется алгоритм Диффи — Хеллмана (DH). Для шифрования передаваемых

данных используется

симметричное шифрование, алгоритмы AES, Blowfish или 3DES. Целостность передачи данных проверяется с помощью CRC32 в

SSH1 или HMAC-SHA1/HMAC-MD5 в SSH2.

Для сжатия шифруемых данных может использоваться алгоритм LempelZiv (LZ77), который обеспечивает такой же уровень

сжатия, что и архиватор ZIP.

Сжатие SSH включается лишь по запросу клиента, и на практике используется редко.

№7 SSH

HQ-R: conf>object-group service SRV\_SSH>port 3035>ex>object-groupe network SERVER\_IP>

ip add 192.168.100.2>ex>nat destination>pool SERVER\_POOL>ip add 192.168.100.2>ip port 22>

ex>rulest DNAT>ftom int gi 1/0/1>rule 1>match destination-add PUBLIC\_POOL>match protocol tcp>

match destination-port SRV\_SSH>action destination-nat pool SERVER\_POOL>enabe>end>com>con

(проверка с CLI ssh -p 3035 user@1.1.1.2)

№8 контроль доступа

HQ-R: conf>ip access-list extended NO\_ACCESS\_SSH\_CLI>rule 1>action deny>match protocol tcp>

match source-address 3.3.3.2 255.255.255.252>match destination-port 3035>enable>ex>rule 2>

action permit>enable>ex>ex>int gi 1/0/1>server-acl input N0\_ACCESS\_SSH\_CLI>end>com>con

systemctl restart network(проверка с CLI ssh -p 3035 user@1.1.1.2,BR-SRV CLI ssh -p 3035 user@1.1.1.2)

Iperf3 — кроссплатформенная консольная клиент-серверная программа — генератор TCP, UDP и SCTP трафика для

тестирования пропускной способности сети. Написана с нуля и не совместима с предыдущей реализацией, хотя

и содержит часть кода предыдущей реализации.

Утилита позволяет генерировать трафик различного типа для анализа пропускной способности сети. Поддерживается

многопоточная работа (в версии 2 посредством запуска на разных портах, в версии 3 — параметром у клиента). По

умолчанию тест выполняется в направлении от клиента к серверу. Чтобы проводить двунаправленное тестирование со

стороны клиента, необходимо использовать ключ -r (или -d для одновременной передачи пакетов в обоих направлениях)

в версии 2 и ключ -R в версии 3.

№9NTP

HQ-R: conf> int loopback 1>ip add 1.2.3.4/32 >end>com>con>conf>clock timezone gmt +3>

ntp enable>ntp broadcast-client enable>end>com>con(првоерка sh ntp con)

BR-R: conf> clock timezone gmt +3>no ntp broadcast-client enable>ntp enable>ntp server1.2.3.4>

end>com>con(проверка sh ntp peers)

BR-SRV,HQ-SRV,HQ-CLI: vim /etc/chrony.conf>первая строчка server 1.2.3.4 iburst prefer

timedatectl set-timezone E/M>systemctl restart chronyd> (проверить chronyc sources)

CLI:vim /etc/chrony.conf>первая строчка server 1.1.1.2 iburst prefer

timedatectl set-timezone E/M>systemctl restart chronyd> (проверить chronyc sources)

Возможности

Поддерживаются базовые возможности Active Directory:

Аутентификация рабочих станций Windows и Linux и служб;

Авторизация и предоставление ресурсов;

Групповые политики (GPO);

Перемещаемые профили (Roaming Profiles);

Поддержка инструментов Microsoft для управления серверами (Remote Server Administration Tools) с

компьютеров под управлением Windows (под WINE не работает);

Поддержка протоколов SMB2 и SMB3 (в том числе с поддержкой шифрования);

Репликация с другими серверами (в том числе с Windows 2012).

Не поддерживается

Не поддерживаются следующие возможности[1]:

Не реплицируется хранилище SYSVOL (можно использовать osync[2][3]);

Доверительные отношения с некоторыми ограничениями[4]):

Правила SID filtering не реализованы;

Невозможно добавить пользователей и группы в доменные группы доверенного домена;

Не поддерживаются поддомены[5];

В Samba отсутствует полноценная поддержка фантомных объектов[6];

Проблемы при построении структуры AD (Active Directory browser), проблемы при поиске объектов в другом

домене;

При применении групповых политик в многодоменной среде могут быть проблемы с определением, в каком домене

находится компьютер;

DFS работает только в режиме одиночного сервера;

Не реализован протокол DFS-R[7];

Не полностью реализована поддержка контроллеров домена только на чтение[8] - RODC[9];

Ограничения размера базы TDB в 4 ГБ[10], приводит к невозможности перехода от леса доменов к плоской структуре

с одним доменом (решено в samba-4.9 и старше с использованием бекенда LMDB[11]).

Не закончено тестирование поддержки CTDB (хранение данных в кластерной инфраструктуре);

Не закончено тестирование работы Winbind.

№10 DNS

HQ-SRV: vim /etc/bind/options.conf> listen-on{ any; };

listen-on-v6 { any; };

fotward first;

forwarders { 77.88.8.8; };

// include "etc/bind"

allow-query { any; };

systemctl restart bind

vim /etc/bind/local.conf> создаем зоны с нуля esk>v>y>p

zone "hq.work" {

type master;

file "hq.db"

};

zone "branch.work" {

type master;

file "br.db";

};

zone "100.168.192.in-addr.arpa" {

type master;

file "reveshq.db";

};

zone "100.16.172.in-addr.arpe" {

type master;

file "reversbr.db";

};

Настройка Kerberos

Откройте от имени суперпользователя файл /etc/krb5.conf.

Проверить что установлено значение false в строке "dns\_lookup\_realm = false".

Раскомментируйте строку в [libdefaults] "default realm" и введите название области

заглавными буквами.

Ниже, под строкой [realms] вместо EXAMPLE.COM введите название области, а вместо

example.com в "default domain" введите IP-адрес сервера с Samba.

Под строкой [domain\_realm] example.com и EXAMPLE.COM замените на ваш домен сохраняя регистр.

Альтернативный вариант

В момент создания домена Samba автоматически конфигурирует шаблон файла krb5.conf

для вашего домена, и оставляет его в директории /var/lib/samba/private/krb5.conf.

Как следствие, можно его просто скопировать с заменой:

Принцип работы SSH

Работа SSH-протокола базируется на нескольких этапах:

открытие транспортного канала аутентификации,

аутентификация,

подключение.

Рассмотрим каждый пункт подробнее.

Открытие транспортного канала

Первым шагом идет установка TCP-соединения. TCP использует сегменты для определения готовности узла-получателя к приему данных.

Когда отправитель хочет установить соединение, TCP отправляет сегмент SYN протоколу на принимающем хосте. Принимающий

TCP возвращает сегмент ACK, чтобы подтвердить успешное получение SYN. Отправляющий TCP отправляет еще один ACK, затем —

переходит к отправке данных. Этот обмен управляющей информацией называется трехсторонним рукопожатием.

cp /etc/bind/zone/localdomain /etc/bind/zone/hq.db

cp /etc/bind/zone/localdomain /etc/bind/zone/br.db

cp /etc/bind/zone/localdomain /etc/bind/zone/revershq.db

cp /etc/bind/zone/localdomain /etc/bind/zone/reversbr.db

chown root:named /etc/bind/zone/hq.db

chown root:named /etc/bind/zone/br.db

chown root:named /etc/bind/zone/revershq.db

chown root:named /etc/bind/zone/reversbr.db

SSH (англ. Secure Shell — «безопасная оболочка»[1]) — сетевой протокол прикладного уровня,

позволяющий производить удалённое управление операционной системой и туннелирование TCP-соединений

(например, для передачи файлов). Схож по функциональности с протоколами Telnet и rlogin, но, в отличие

от них, шифрует весь трафик, включая и передаваемые пароли. SSH допускает выбор различных алгоритмов шифрования.

SSH-клиенты и SSH-серверы доступны для большинства сетевых операционных систем.

SSH позволяет безопасно передавать в незащищённой среде практически любой другой сетевой протокол. Таким образом,

можно не только удалённо работать на компьютере через командную оболочку, но и передавать по шифрованному каналу

звуковой поток или видео (например, с веб-камеры)[2]. Также SSH может использовать сжатие передаваемых данных для

последующего их шифрования, что удобно, например, для удалённого запуска клиентов X Window System.

Большинство хостинг-провайдеров за определённую плату предоставляет клиентам доступ к их домашнему каталогу по SSH.

Это может быть удобно как для работы в командной строке, так и для удалённого запуска программ (в том числе

графических приложений).

SSH — это протокол прикладного уровня. SSH-сервер обычно прослушивает соединения на TCP-порту 22. Спецификация протокола

SSH-2 содержится в RFC 4251. Для аутентификации сервера в SSH используется протокол аутентификации сторон на основе алгоритмов

электронно-цифровой подписи RSA или DSA, но допускается также аутентификация при помощи пароля (режим обратной совместимости с

Telnet) и даже ip-адреса хоста (режим обратной совместимости с rlogin).

Аутентификация по паролю наиболее распространена. При каждом подключении подобно https вырабатывается общий секретный ключ для

шифрования трафика.

При аутентификации по ключевой паре предварительно генерируется пара открытого и закрытого ключей для определённого пользователя.

На машине, с которой требуется произвести подключение, хранится закрытый ключ, а на удалённой машине — открытый. Эти файлы

не передаются

при аутентификации, система лишь проверяет, что владелец открытого ключа также владеет и закрытым. При данном подходе,

как правило,

настраивается автоматический вход от имени конкретного пользователя в ОС.

Аутентификация по ip-адресу небезопасна, эту возможность чаще всего отключают.

Для создания общего секрета (сеансового ключа) используется алгоритм Диффи — Хеллмана (DH). Для шифрования передаваемых

данных используется

симметричное шифрование, алгоритмы AES, Blowfish или 3DES. Целостность передачи данных проверяется с помощью CRC32 в

SSH1 или HMAC-SHA1/HMAC-MD5 в SSH2.

vim /etc/bind/zone/hq.db

$TTL 1D

@ IN SOA hq.work. root.hq.work. (

2024021400 ;serial

12H ;refresh

1H ;retry

1W ;expire

1H ;ncache

)

IN NS hq.work.

IN A 127.0.0.0

hq-r IN A 192.168.100.1

hq-srv IN A 192.168.100.2

SSH (англ. Secure Shell — «безопасная оболочка»[1]) — сетевой протокол прикладного уровня,

позволяющий производить удалённое управление операционной системой и туннелирование TCP-соединений

(например, для передачи файлов). Схож по функциональности с протоколами Telnet и rlogin, но, в отличие

от них, шифрует весь трафик, включая и передаваемые пароли. SSH допускает выбор различных алгоритмов шифрования.

SSH-клиенты и SSH-серверы доступны для большинства сетевых операционных систем.

SSH позволяет безопасно передавать в незащищённой среде практически любой другой сетевой протокол. Таким образом,

можно не только удалённо работать на компьютере через командную оболочку, но и передавать по шифрованному каналу

звуковой поток или видео (например, с веб-камеры)[2]. Также SSH может использовать сжатие передаваемых данных для

последующего их шифрования, что удобно, например, для удалённого запуска клиентов X Window System.

Большинство хостинг-провайдеров за определённую плату предоставляет клиентам доступ к их домашнему каталогу по SSH.

Это может быть удобно как для работы в командной строке, так и для удалённого запуска программ (в том числе

графических приложений).

SSH — это протокол прикладного уровня. SSH-сервер обычно прослушивает соединения на TCP-порту 22. Спецификация протокола

SSH-2 содержится в RFC 4251. Для аутентификации сервера в SSH используется протокол аутентификации сторон на основе алгоритмов

электронно-цифровой подписи RSA или DSA, но допускается также аутентификация при помощи пароля (режим обратной совместимости с

Telnet) и даже ip-адреса хоста (режим обратной совместимости с rlogin).

Аутентификация по паролю наиболее распространена. При каждом подключении подобно https вырабатывается общий секретный ключ для

шифрования трафика.

При аутентификации по ключевой паре предварительно генерируется пара открытого и закрытого ключей для определённого пользователя.

На машине, с которой требуется произвести подключение, хранится закрытый ключ, а на удалённой машине — открытый. Эти файлы

не передаются

при аутентификации, система лишь проверяет, что владелец открытого ключа также владеет и закрытым. При данном подходе,

как правило,

настраивается автоматический вход от имени конкретного пользователя в ОС.

Аутентификация по ip-адресу небезопасна, эту возможность чаще всего отключают.

Для создания общего секрета (сеансового ключа) используется алгоритм Диффи — Хеллмана (DH). Для шифрования передаваемых

данных используется

симметричное шифрование, алгоритмы AES, Blowfish или 3DES. Целостность передачи данных проверяется с помощью CRC32 в

SSH1 или HMAC-SHA1/HMAC-MD5 в SSH2.

vim /etc/bind/zone/br.db

$TTL 1D

@ IN SOA branch.work. root.branch.work. (

2024021400 ;serial

12H ;refresh

1H ;retry

1W ;expire

1H ;ncache

)

IN NS branch.work.

IN A 127.0.0.0

br-r IN A 172.16.100.1

br-srv IN A 172.16.100.2

SSH (англ. Secure Shell — «безопасная оболочка»[1]) — сетевой протокол прикладного уровня,

позволяющий производить удалённое управление операционной системой и туннелирование TCP-соединений

(например, для передачи файлов). Схож по функциональности с протоколами Telnet и rlogin, но, в отличие

от них, шифрует весь трафик, включая и передаваемые пароли. SSH допускает выбор различных алгоритмов шифрования.

SSH-клиенты и SSH-серверы доступны для большинства сетевых операционных систем.

SSH позволяет безопасно передавать в незащищённой среде практически любой другой сетевой протокол. Таким образом,

можно не только удалённо работать на компьютере через командную оболочку, но и передавать по шифрованному каналу

звуковой поток или видео (например, с веб-камеры)[2]. Также SSH может использовать сжатие передаваемых данных для

последующего их шифрования, что удобно, например, для удалённого запуска клиентов X Window System.

Большинство хостинг-провайдеров за определённую плату предоставляет клиентам доступ к их домашнему каталогу по SSH.

Это может быть удобно как для работы в командной строке, так и для удалённого запуска программ (в том числе

графических приложений).

SSH — это протокол прикладного уровня. SSH-сервер обычно прослушивает соединения на TCP-порту 22. Спецификация протокола

SSH-2 содержится в RFC 4251. Для аутентификации сервера в SSH используется протокол аутентификации сторон на основе алгоритмов

электронно-цифровой подписи RSA или DSA, но допускается также аутентификация при помощи пароля (режим обратной совместимости с

Telnet) и даже ip-адреса хоста (режим обратной совместимости с rlogin).

Аутентификация по паролю наиболее распространена. При каждом подключении подобно https вырабатывается общий секретный ключ для

шифрования трафика.

При аутентификации по ключевой паре предварительно генерируется пара открытого и закрытого ключей для определённого пользователя.

На машине, с которой требуется произвести подключение, хранится закрытый ключ, а на удалённой машине — открытый. Эти файлы

не передаются

при аутентификации, система лишь проверяет, что владелец открытого ключа также владеет и закрытым. При данном подходе,

как правило,

настраивается автоматический вход от имени конкретного пользователя в ОС.

Аутентификация по ip-адресу небезопасна, эту возможность чаще всего отключают.

Для создания общего секрета (сеансового ключа) используется алгоритм Диффи — Хеллмана (DH). Для шифрования передаваемых

данных используется

симметричное шифрование, алгоритмы AES, Blowfish или 3DES. Целостность передачи данных проверяется с помощью CRC32 в

SSH1 или HMAC-SHA1/HMAC-MD5 в SSH2.

vim /etc/bind/zone/revershq.db

$TTL 1D

@ IN SOA hq.work. root.hq.work. (

2024021400 ;serial

12H ;refresh

1H ;retry

1W ;expire

1H ;ncache

)

IN NS hq.work.

1 IN PTR hq-r.hq.work.

2 IN PTR hq-srv.hq.work.

SSH (англ. Secure Shell — «безопасная оболочка»[1]) — сетевой протокол прикладного уровня,

позволяющий производить удалённое управление операционной системой и туннелирование TCP-соединений

(например, для передачи файлов). Схож по функциональности с протоколами Telnet и rlogin, но, в отличие

от них, шифрует весь трафик, включая и передаваемые пароли. SSH допускает выбор различных алгоритмов шифрования.

SSH-клиенты и SSH-серверы доступны для большинства сетевых операционных систем.

SSH позволяет безопасно передавать в незащищённой среде практически любой другой сетевой протокол. Таким образом,

можно не только удалённо работать на компьютере через командную оболочку, но и передавать по шифрованному каналу

звуковой поток или видео (например, с веб-камеры)[2]. Также SSH может использовать сжатие передаваемых данных для

последующего их шифрования, что удобно, например, для удалённого запуска клиентов X Window System.

Большинство хостинг-провайдеров за определённую плату предоставляет клиентам доступ к их домашнему каталогу по SSH.

Это может быть удобно как для работы в командной строке, так и для удалённого запуска программ (в том числе

графических приложений).

SSH — это протокол прикладного уровня. SSH-сервер обычно прослушивает соединения на TCP-порту 22. Спецификация протокола

SSH-2 содержится в RFC 4251. Для аутентификации сервера в SSH используется протокол аутентификации сторон на основе алгоритмов

электронно-цифровой подписи RSA или DSA, но допускается также аутентификация при помощи пароля (режим обратной совместимости с

Telnet) и даже ip-адреса хоста (режим обратной совместимости с rlogin).

Аутентификация по паролю наиболее распространена. При каждом подключении подобно https вырабатывается общий секретный ключ для

шифрования трафика.

При аутентификации по ключевой паре предварительно генерируется пара открытого и закрытого ключей для определённого пользователя.

На машине, с которой требуется произвести подключение, хранится закрытый ключ, а на удалённой машине — открытый. Эти файлы

не передаются

при аутентификации, система лишь проверяет, что владелец открытого ключа также владеет и закрытым. При данном подходе,

как правило,

настраивается автоматический вход от имени конкретного пользователя в ОС.

Аутентификация по ip-адресу небезопасна, эту возможность чаще всего отключают.

Для создания общего секрета (сеансового ключа) используется алгоритм Диффи — Хеллмана (DH). Для шифрования передаваемых

данных используется

симметричное шифрование, алгоритмы AES, Blowfish или 3DES. Целостность передачи данных проверяется с помощью CRC32 в

SSH1 или HMAC-SHA1/HMAC-MD5 в SSH2.

vim /etc/bind/zone/reversbr.db

$TTL 1D

@ IN SOA branch.work. root.branch.work. (

2024021400 ;serial

12H ;refresh

1H ;retry

1W ;expire

1H ;ncache

)

IN NS branch.work.

1 IN PTR br-r.branch.work.

(проверить named-checkconf -z, host hq-r.work, host 172.16.100.1