

навигация

- [Заглавная страница](#)
- [Свежие правки](#)
- [Случайная статья](#)
- [Справка](#)

поиск

Перейти

Найти

инструменты

- [Ссылки сюда](#)
- [Связанные правки](#)
- [Спецстраницы](#)
- [Версия для печати](#)
- [Постоянная ссылка](#)

на других языках

- [English Wikipedia](#)
- [Deutsche Wikipedia](#)
- [Русская Википедия](#)

статья

Xgu.ru теперь в Контакте — приходите и подключайтесь.

Пока мы работаем над следующими видео, вы можете подключиться в Контакте. Познакомимся и обсудим новые страницы и ролики.



VLAN

Автор: [Наташа Самойленко](#)

Автор: [Игорь Чубин](#)

VLAN (Virtual Local Area Network) — группа устройств, имеющих возможность взаимодействовать между собой напрямую на канальном уровне, хотя физически при этом они могут быть подключены к разным сетевым коммутаторам. И наоборот, устройства, находящиеся в разных VLAN'ах, невидимы друг для друга на канальном уровне, даже если они подключены к одному коммутатору, и связь между этими устройствами возможна только на сетевом и более высоких уровнях.

В современных сетях VLAN — главный механизм для создания логической топологии сети, не зависящей от её физической топологии. VLAN'ы используются для сокращения широковещательного трафика в сети. Имеют большое значение с точки зрения безопасности, в частности как средство борьбы с [ARP-spoofing](#)'ом.

Содержание [\[убрать\]](#)

- 1 Зачем нужен VLAN?
- 2 Тегирование трафика VLAN
- 3 Коммутатор и VLAN'ы
 - 3.1 Принципы работы коммутатора
 - 3.1.1 Механизмы передачи фреймов
 - 3.1.2 Пример сети для демонстрации использования механизмов передачи фреймов
 - 3.2 Хосты в одном VLAN на одном коммутаторе
 - 3.3 Хосты в разных VLAN на одном коммутаторе
 - 3.4 Хосты в разных VLAN на разных коммутаторах (объяснение тегированных портов)
 - 3.4.1 Добавлен второй коммутатор и хосты в VLAN 2
 - 3.4.2 Ко второму коммутатору добавлены хосты в VLAN 10
 - 3.4.3 Создание тегированного порта между коммутаторами
- 4 Принадлежность VLAN
- 5 Настройка VLAN на коммутаторах
 - 5.1 Настройка VLAN на коммутаторах Cisco под управлением IOS
 - 5.1.1 Настройка access портов
 - 5.1.2 Настройка транка (trunk)
 - 5.1.2.1 Настройка статического транка
 - 5.1.2.2 Динамическое создание транков (DTP)
 - 5.1.2.3 Разрешенные VLAN'ы
 - 5.1.2.4 Native VLAN
 - 5.1.3 Настройка маршрутизации между VLAN
 - 5.1.4 Перевод интерфейса в режим 3го уровня
 - 5.1.5 Просмотр информации
 - 5.1.6 Диапазоны VLAN
 - 5.1.7 Пример настройки
 - 5.1.7.1 Пример базовой настройки VLAN, без настройки маршрутизации
 - 5.1.7.2 Пример конфигураций с настройкой маршрутизации между VLAN
 - 5.2 Настройка VLAN на коммутаторах HP ProCurve
 - 5.2.1 Настройка маршрутизации между VLAN
 - 5.2.2 Пример конфигурации
 - 5.3 Настройка VLAN на коммутаторах D-LINK
- 6 Настройка VLAN на маршрутизаторах
 - 6.1 Настройка VLAN на маршрутизаторах Cisco
 - 6.1.1 Пример настройки
 - 6.1.2 Настройка native VLAN
 - 6.1.3 Проверка настройки
 - 6.2 Настройка VLAN на маршрутизаторах ProCurve
- 7 Настройка VLAN в операционных системах
 - 7.1 Настройка VLAN в Linux
 - 7.1.1 Настройка VLAN при загрузке в Debian GNU/Linux
 - 7.1.2 Настройка VLAN при загрузке в CentOS
 - 7.1.3 QinQ-инкапсуляция в Linux
 - 7.2 Настройка VLAN во FreeBSD
 - 7.3 Настройка VLAN в (Open)Solaris
 - 7.4 Настройка VLAN в Oracle Solaris 11 Express
 - 7.5 Настройка VLAN в Windows
- 8 Дополнительные вопросы
 - 8.1 Автоматизированное создание VLAN
 - 8.2 Динамическая настройка VLAN

Зачем нужен VLAN?

[\[править\]](#)

Гибкое разделение устройств на группы

Как правило, одному VLAN соответствует одна подсеть. Устройства, находящиеся в разных VLAN, будут находиться в разных подсетях. Но в то же время VLAN не привязан к местоположению устройств и поэтому устройства, находящиеся на расстоянии друг от друга, все равно могут быть в одном VLAN независимо от местоположения

Уменьшение количества широковещательного трафика в сети

Каждый VLAN — это отдельный широковещательный домен. Например, коммутатор — это устройство 2 уровня модели OSI. Все порты на коммутаторе с лишь одним VLAN находятся в одном широковещательном домене. Создание дополнительных VLAN на коммутаторе означает разбиение коммутатора на несколько широковещательных доменов. Если один и тот же VLAN настроен на разных коммутаторах, то порты разных коммутаторов будут образовывать один широковещательный домен.

Увеличение безопасности и управляемости сети

Когда сеть разбита на VLAN, упрощается задача применения политик и правил безопасности. С VLAN политики можно применять к целым подсетям, а не к отдельному устройству. Кроме того, переход из одного VLAN в другой предполагает прохождение через устройство 3 уровня, на котором, как правило, применяются политики, разрешающие или запрещающие доступ из VLAN в VLAN.

Тегирование трафика VLAN

[\[править\]](#)

Компьютер при отправке трафика в сеть даже не догадывается, в каком VLAN'е он размещён. Об этом думает коммутатор. Коммутатор знает, что компьютер, который подключен к определённому порту, находится в соответствующем VLAN'е. Трафик, приходящий на порт определённого VLAN'а, ничем особенным не отличается от трафика другого VLAN'а. Другими словами, никакой информации о принадлежности трафика определённому VLAN'у в нём нет.

Однако, если через порт может прийти трафик разных VLAN'ов, коммутатор должен его как-то различать. Для этого каждый кадр (frame) трафика должен быть помечен каким-то особым образом. Пометка должна говорить о том, какому VLAN'у трафик принадлежит.

Наиболее распространённый сейчас способ ставить такую пометку описан в открытом стандарте [IEEE 802.1Q](#). Существуют проприетарные протоколы, решающие похожие задачи, например, протокол [ISL](#) от Cisco Systems, но их популярность значительно ниже (и снижается).

Коммутатор и VLAN'ы

[\[править\]](#)

VLAN'ы могут быть настроены на коммутаторах, маршрутизаторах, других сетевых устройствах и на хостах. Однако, для объяснения VLAN лучше всего подойдет коммутатор.

Коммутатор — устройство 2го уровня и изначально все порты коммутатора находятся, как правило, в VLAN 1 и, следовательно, в одном широковещательном сегменте.

Это значит, что если одно из устройств, которое подключено к порту коммутатора, отправит широковещательный фрейм, то коммутатор перенаправит этот фрейм на все остальные порты, к которым подключены устройства, и они получат этот фрейм.

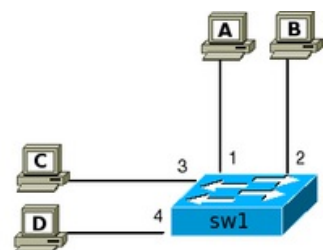
Принципы работы коммутатора

[\[править\]](#)

В этом разделе рассматривается коммутатор с настройками по умолчанию, то есть все его порты находятся в VLAN 1.

Для того чтобы передавать фреймы, коммутатор использует таблицу коммутации. Изначально, после включения коммутатора таблица пуста. Заполняет её коммутатор автоматически, при получении фреймов от хостов. Когда коммутатор получает фрейм от хоста, он сначала передает его в соответствии со своими правилами (описаны ниже), а затем запоминает MAC-адрес отправителя во фрейме и ставит его в соответствие порту на котором он был получен.

Например, для изображенной схемы, итоговая таблица коммутации будет иметь такой вид (после того как все хосты передавали какой-то трафик):



Порт коммутатора	MAC-адрес хоста
1	A
2	B
3	C
4	D

Когда таблица заполнена, коммутатор знает на каких портах у него находятся какие хосты и передает фреймы на соответствующие порты.

Unicast фрейм с MAC-адресом получателя для которого у коммутатора нет записи в таблице коммутации, называется **unknown unicast**.

Механизмы передачи фреймов

[\[править\]](#)

Для того, чтобы передавать фреймы, коммутатор использует три базовых механизма:

- **Flooding** — фрейм, полученный на один из портов, передается на остальные порты коммутатора. Коммутатор выполняет эту операцию в двух случаях:
 - при получении широковещательного или multicast (если не настроена поддержка multicast) фрейма,
 - при получении unknown unicast фрейма. Это позволяет коммутатору доставить фрейм хосту (при условии, что хост достижим и существует), даже когда он не знает, где хост находится.
- **Forwarding** — передача фрейма, полученного на одном порту, через другой порт в соответствии с записью в таблице коммутации.
- **Filtering** — если коммутатор получает фрейм через определенный порт, и MAC-адрес получателя доступен через этот же порт (это указано в таблице коммутации), то коммутатор отбрасывает фрейм. То есть, коммутатор считает, что в этом случае хост уже получил этот фрейм, и не дублирует его.

Пример сети для демонстрации использования механизмов передачи фреймов

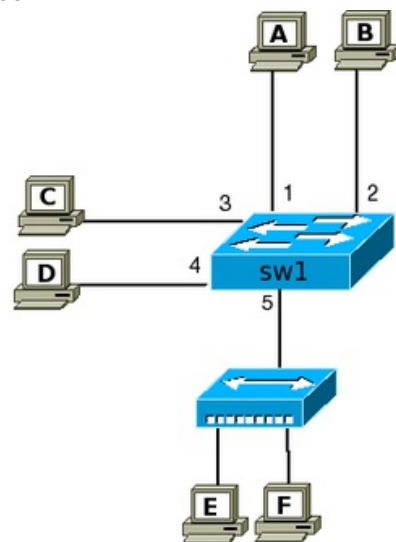
[\[править\]](#)

На рисунке изображен коммутатор sw1 и повторитель (hub) к которому подключены два хоста.

Изначально к коммутатору были подключены три хоста А, В и С.

Соответственно у коммутатора такая таблица коммутации:

Порт коммутатора	MAC-адрес хоста
1	A
2	B
3	C



Когда хост А отправляет фрейм хосту В, коммутатор использует механизм **forwarding**, так как ему известно где находятся оба хоста и хосты находятся на разных портах коммутатора.

Далее к коммутатору подключили хост D. Если хост А отправляет фрейм хосту D, то для коммутатора это unknown unicast фрейм, так как в таблице коммутации нет записи о MAC-адресе D. В соответствии со своими правилами коммутатор выполняет **flooding** и передает фрейм на все порты, кроме 1 (с которого фрейм был получен).

После того как коммутатор получит фрейм от хоста D, он запомнит его адрес и создаст соответствующую запись в таблице коммутации.

К коммутатору подключили повторитель с двумя хостами и коммутатор выучил их адреса.

Соответствующая таблица коммутации:

Порт коммутатора	MAC-адрес хоста
1	A
2	B
3	C
4	D
5	E
5	F

Если после этого хост Е будет передавать фрейм хосту F, то коммутатор получит его, но не будет передавать далее. В этой ситуации коммутатор использует механизм **filtering**, так как MAC-адрес получателя доступен через тот же порт, что и отправитель.

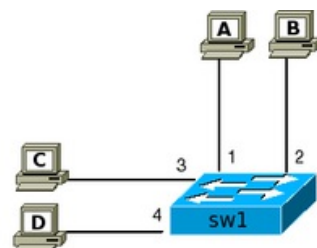
Хосты в одном VLAN на одном коммутаторе

[\[править\]](#)

К коммутатору подключены 4 хоста. Для упрощения будем считать, что А, В, С и D это соответствующие MAC-адреса хостов.

Соответствующая таблица коммутации:

Порт коммутатора	MAC-адрес хоста
1	A
2	B
3	C
4	D



Хосты в разных VLAN на одном коммутаторе

[\[править\]](#)

Обычно, по умолчанию все порты коммутатора считаются нетегированными членами VLAN 1. В процессе настройки или работы коммутатора они могут перемещаться в другие VLAN'ы.

На коммутаторе, который изображен на рисунке, настроены два VLAN'а, все порты в соответствующих VLAN настроены как нетегированные, т. е. не используют теги IEEE 802.1Q при передаче фреймов (access-порты в терминологии Cisco).

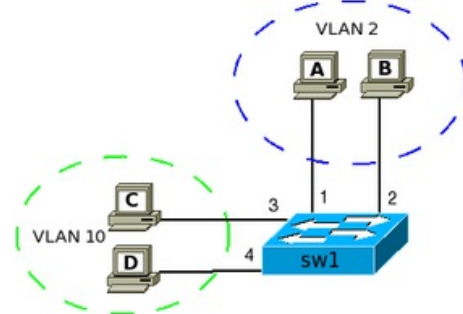
После этого на коммутаторе существуют две таблицы коммутации.

Для VLAN 2:

Порт коммутатора	MAC-адрес хоста
1	A
2	B

Для VLAN 10:

Порт коммутатора	MAC-адрес хоста
3	C
4	D



Все базовые механизмы коммутатора остаются точно такими же как и до разделения на VLAN, но они используются только в пределах соответствующего VLAN.

Например, если хост из VLAN 10 отправляет широковещательный фрейм, то он будет отправлен только на порты в этом VLAN.

Получается, что нетегированные порты это "обычные" порты коммутатора. Это просто возможность сообщить коммутатору о том, какому VLAN принадлежат порты. Затем коммутатор использует эту информацию при передаче фреймов.

Как правило, реально в таблице коммутации в коммутаторах указывается порт, MAC-адрес и VLAN. То есть, для указанного примера таблица коммутации будет такая:

Порт коммутатора	VLAN	MAC-адрес хоста
1	2	A
2	2	B
3	10	C
4	10	D

Однако, далее для упрощения используется запись таблицы коммутации в виде соответствия между портами и MAC-адресами.

Хосты в разных VLAN на разных коммутаторах (объяснение тегированных портов) [\[править\]](#)

К используемому примеру добавляется ещё один коммутатор.



В ходе объяснения используются промежуточные схемы подключения устройств и настройки коммутаторов, которые обычно не используются в реальной сети. Но они необходимы для объяснения.

Добавлен второй коммутатор и хосты в VLAN 2 [\[править\]](#)

Для начала добавлен коммутатор sw2 и два хоста E и F в VLAN 2. Если рассматривать два коммутатора отдельно, то получается, что на коммутаторе sw1 осталась прежняя таблица коммутации, а на коммутаторе sw2 таблица такая (пока что коммутаторы не соединены):

Порт коммутатора	MAC-адрес хоста
7	E
8	F

Теперь необходимо чтобы хосты A, B, E, F "увидели" друг друга. Они должны находиться в одном VLAN. То есть, необходимо каким-то образом указать коммутатору, что ещё на одном порту есть хосты в соответствующем VLAN'е.

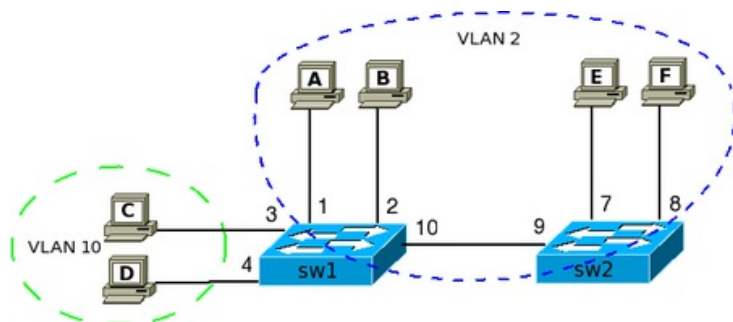
Для указанного примера достаточно добавить на коммутаторе sw1 порт 10 в VLAN 2, а на коммутаторе sw2 порт 9 в VLAN 2. Принадлежность к VLAN указывается настройкой порта нетегированным в VLAN 2 (пока что). После этого на коммутаторах в таблицах коммутации добавятся новые порты и соответствующие MAC-адреса хостов. Теперь четыре хоста на разных коммутаторах находятся в одном широковещательном сегменте.

Таблица коммутации **sw1** для VLAN 2:

Порт коммутатора	MAC-адрес хоста
1	A
2	B
10	E
10	F

Таблица коммутации **sw2** для VLAN 2:

Порт коммутатора	MAC-адрес хоста
7	E
8	F
9	A
9	B



Ко второму коммутатору добавлены хосты в VLAN 10

[править]

К коммутатору sw2 добавлены два хоста G и H в VLAN 10.

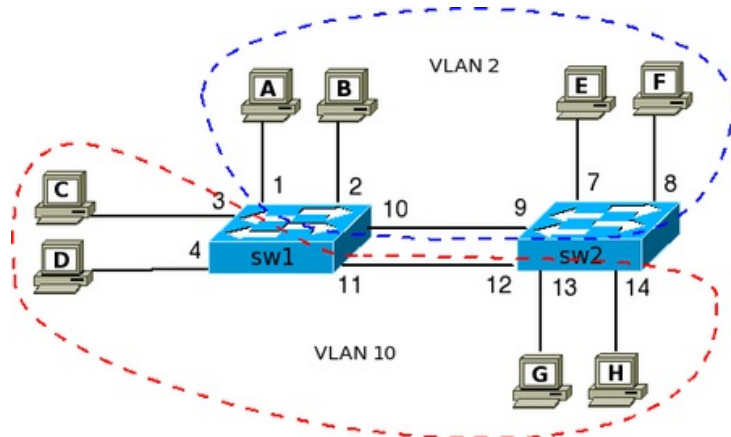
Для того чтобы хосты C и D в VLAN 10 на коммутаторе sw1, могли обмениваться информацией с хостами VLAN 10 на коммутаторе sw2 добавлен линк между коммутаторами. Логика аналогична добавлению хостов в VLAN 2.

Таблица коммутации **sw1** для VLAN 10:

Порт коммутатора	MAC-адрес хоста
3	C
4	D
11	G
11	H

Таблица коммутации **sw2** для VLAN 10:

Порт коммутатора	MAC-адрес хоста
13	G
14	H
12	C
12	D



Создание тегированного порта между коммутаторами

[править]

Когда необходимо передать трафик одного-двух VLAN'ов между коммутаторами, то схема, которая использовалась выше, выглядит нормально. Однако, когда количество VLAN возрастает, то схема явно становится очень неудобной, так как для каждого VLAN надо будет добавлять линк между коммутаторами для того, чтобы объединить хосты в один широкоэвещательный сегмент.

Для решения этой проблемы используются **тегированные порты**.

Тегированный порт позволяет коммутатору передать трафик нескольких VLAN'ов через один порт и сохранить при этом информацию о том, в пределах какого именно VLAN'a передается фрейм.

На коммутаторах sw1 и sw2 порты 21 и 22, соответственно, это тегированные порты.

Для того, чтобы коммутаторы понимали какому VLAN принадлежит пришедший фрейм и использовали соответствующую таблицу коммутации для его обработки, выполняется тегирование фрейма.

Например, если хост E передает фрейм хосту A, то коммутатор sw2 проверяет свою таблицу и видит, что хост A доступен через порт 22. Так как порт настроен как тегированный, то когда фрейм выходит с порта 22 в нём проставляется тег, который указывает какому VLAN'у принадлежит этот фрейм. В данном случае проставляется тег с VLAN'ом 2.

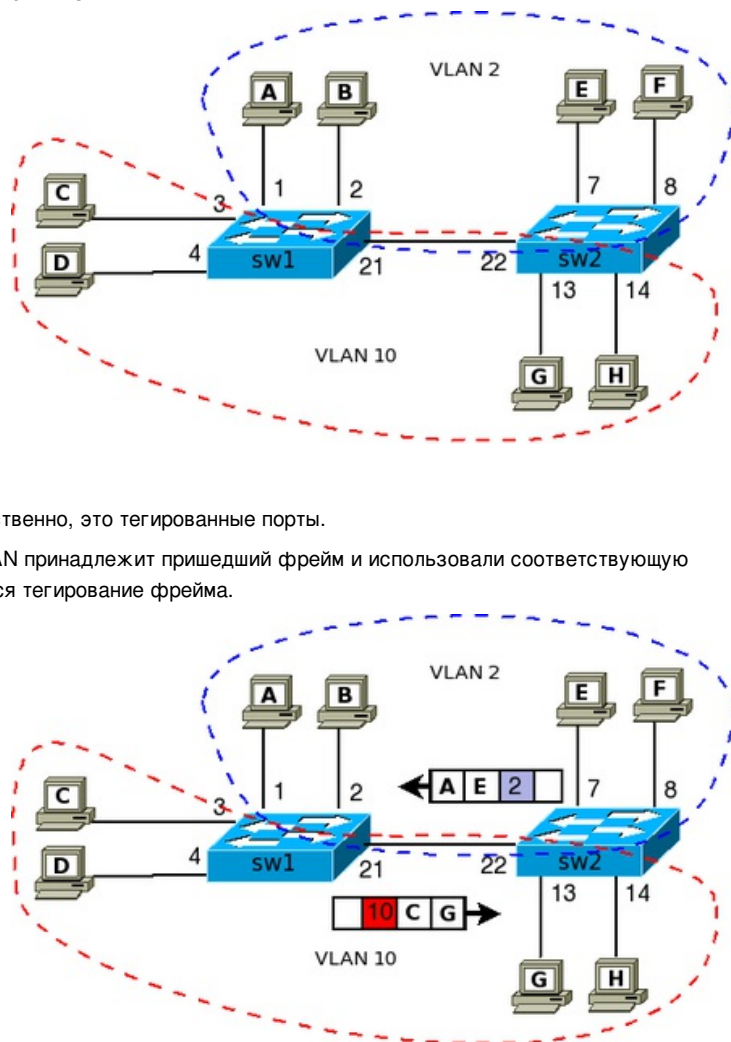


Реальная структура фрейма с тегом описана на странице [802.1Q](#).

Коммутатор sw1 получает тегированный фрейм через тегированный порт 21. Для того чтобы определить на какой порт его передавать далее sw1 использует таблицу коммутации для VLAN 2 (так как этот VLAN был указан в теге). На коммутаторе sw1 порт 21 должен быть настроен как тегированный для того чтобы коммутатор не отбрасывал тегированные фреймы, а считывал информацию тега. И соответственно чтобы он также помечал фрейм тегом, когда будет передаваться трафик коммутатору sw2.

Остальные порты коммутатора остаются нетегированными. И для хостов операция тегирования, которую выполняют коммутаторы абсолютно прозрачна. Хосты ничего не знают о теге и получают обычные фреймы.

Аналогичные действия выполняются, например, при передаче фрейма от хоста C хосту G.



Принадлежность VLAN

[править]

Порты коммутатора, поддерживающие VLAN'ы, (с некоторыми допущениями) можно разделить на два множества:

1. Тегированные порты (или транковые порты, *trunk-порты* в терминологии Cisco).
2. Нетегированные порты (или порты доступа, *access-порты* в терминологии Cisco);

Тегированные порты нужны для того, чтобы через один порт была возможность передавать данные принадлежащие к различным VLAN и, соответственно, получать трафик нескольких VLAN на один порт. Информация о принадлежности трафика к конкретному VLAN, как было сказано выше, указывается в специальном теге. Без тега коммутатор не сможет различать трафик различных VLAN.

Если порт нетегированный и принадлежит к какому-либо VLAN, то трафик для этого VLAN передается без тега. На Cisco нетегированный порт (access-порт) может быть только в одном VLAN, на некоторых других свитчах (например, ZyXEL, D-Link и Planet) реализация иная.

Если порт тегирован для нескольких VLAN, то в этом случае весь нетегированный трафик будет приниматься специальным *родным* VLAN (native VLAN). С этим параметром (native, PVID, port VID) возникает много путаницы. Например, свитчи Planet для правильной работы нетегированного порта требуют поместить порт в VLAN, задать режим порта untagged (нетегированный), и прописать номер этого VLAN в PVID этого порта. На коммутаторах HP ProCurve тегированный порт начинает работать как тегированный только если поставить его PVID в "None".

Если порт принадлежит только одному VLAN как нетегированный, то тегированный трафик, приходящий через такой порт, должен удаляться. На деле это поведение обычно настраивается.

Проще всего это понять, если "забыть" всю внутреннюю структуру коммутатора и отталкиваться только от портов. Допустим, есть VLAN с номером 111, есть два порта которые принадлежат к VLAN 111. Они общаются только между собой, с untagged/access-порта выходит нетегированный трафик, с tagged/trunk-порта выходит трафик тегированный в VLAN 111. Все необходимые преобразования прозрачно внутри себя делает коммутатор.

Обычно, по умолчанию все порты коммутатора считаются нетегированными членами VLAN 1. В процессе настройки или работы коммутатора они могут быть перемещены в другие VLAN.

Существуют два подхода к назначению порта в определённый VLAN:

1. *Статическое назначение* — когда принадлежность порта VLAN'у задаётся администратором в процессе настройки;
2. *Динамическое назначение* — когда принадлежность порта VLAN'у определяется в ходе работы коммутатора с помощью процедур, описанных в специальных стандартах, таких, например, как 802.1X. При использовании 802.1X для того чтобы получить доступ к порту коммутатора, пользователь проходит аутентификацию на RADIUS-сервере. По результатам аутентификации порт коммутатора размещается в том или ином VLANе (подробнее: 802.1X и RADIUS).

Настройка VLAN на коммутаторах

[\[править\]](#)

В качестве примера выбрана такая топология, на которой легче объяснить различия в настройках VLAN на коммутаторах различных производителей. В соответствующих разделах объясняется как настраивать VLAN'ы, а также приведены примеры конфигурации.

Настройка VLAN на коммутаторах Cisco под управлением IOS

[\[править\]](#)



На странице [VLAN в Cisco/Lab](#) находятся лабораторные, которые можно сделать для того чтобы на практике попробовать настройки, которые описываются в этом разделе. Лабораторные подготовлены в Packet Tracer, но аналогично могут быть выполнены и на реальном оборудовании.

Основная страница: [VLAN в Cisco](#)

Терминология Cisco:

- access port — порт принадлежащий одному VLAN'у и передающий нетегированный трафик
- trunk port — порт передающий тегированный трафик одного или нескольких VLAN'ов

Коммутаторы Cisco ранее поддерживали два протокола 802.1Q и ISL. ISL — проприетарный протокол использующийся в оборудовании Cisco. ISL полностью инкапсулирует фрейм для передачи информации о принадлежности к VLAN'у.

В современных моделях коммутаторов Cisco ISL не поддерживается.

Создание VLAN'а и задание имени:

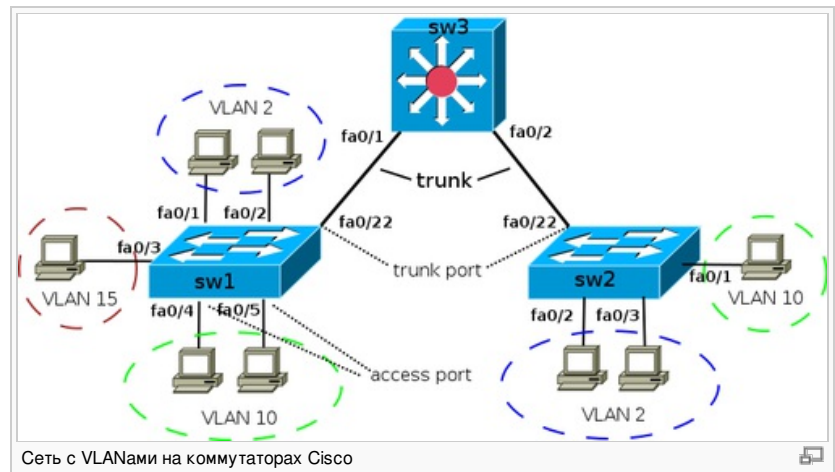
```
sw1(config)# vlan 2
sw1(config-vlan)# name test
```

Настройка access портов

[\[править\]](#)

Назначение портов 1 и 2 коммутатора в VLAN:

```
sw1(config)# interface range fa0/1 - 2
sw1(config-if)# switchport mode access
sw1(config-if)# switchport access vlan 2
```



Назначение диапазона портов с fa0/4 до fa0/5 в vlan 10:

```
sw1(config)# interface range fa0/4 - 5
sw1(config-if-range)# switchport mode access
sw1(config-if-range)# switchport access vlan 10
```

Просмотр информации о VLAN'ах:

```
sw1# show vlan brief
VLAN Name      Status  Ports
-----
1  default      active  Fa0/6, Fa0/7, Fa0/8, Fa0/9,
                        Fa0/10, Fa0/11, Fa0/12, Fa0/13,
                        Fa0/14, Fa0/15, Fa0/16, Fa0/17,
                        Fa0/18, Fa0/19, Fa0/20, Fa0/21,
                        Fa0/22, Fa0/23, Fa0/24
2  test         active  Fa0/1, Fa0/2
10  VLAN0010     active  Fa0/4, Fa0/5
15  VLAN0015     active  Fa0/3
```

Настройка транка (trunk)

[\[править\]](#)

Для того чтобы передать через порт трафик нескольких VLAN, порт переводится в режим транка.

Режимы интерфейса (режим по умолчанию зависит от модели коммутатора):

- **auto** — Порт находится в автоматическом режиме и будет переведён в состояние trunk, только если порт на другом конце находится в режиме on или desirable. Т.е. если порты на обоих концах находятся в режиме "auto", то trunk применяться не будет.
- **desirable** — Порт находится в режиме "готов перейти в состояние trunk"; периодически передает DTP-кадры порту на другом конце, запрашивая удаленный порт перейти в состояние trunk (состояние trunk будет установлено, если порт на другом конце находится в режиме on, desirable, или auto).
- **trunk** — Порт постоянно находится в состоянии trunk, даже если порт на другом конце не поддерживает этот режим.
- **nonegotiate** — Порт готов перейти в режим trunk, но при этом не передает DTP-кадры порту на другом конце. Этот режим используется для предотвращения конфликтов с другим "не-cisco" оборудованием. В этом случае коммутатор на другом конце должен быть вручную настроен на использование trunk'a.

По умолчанию в транке разрешены все VLAN. Для того чтобы через соответствующий VLAN в транке передавались данные, как минимум, необходимо чтобы VLAN был активным. Активным VLAN становится тогда, когда он создан на коммутаторе и в нём есть хотя бы один порт в состоянии up/up.

VLAN можно создать на коммутаторе с помощью команды vlan. Кроме того, VLAN автоматически создается на коммутаторе в момент добавления в него интерфейсов в режиме access.

В схеме, которая используется для демонстрации настроек, на коммутаторах sw1 и sw2, нужные VLAN будут созданы в момент добавления access-портов в соответствующие VLAN:

```
sw1(config)# interface fa0/3
sw1(config-if)# switchport mode access
sw1(config-if)# switchport access vlan 15
% Access VLAN does not exist. Creating vlan 15
```

На коммутаторе sw3 access-портов нет. Поэтому необходимо явно создать все необходимые VLAN:

```
sw3(config)# vlan 2,10,15
```

Для автоматического создания VLAN на коммутаторах, может использоваться протокол VTP.

Настройка статического транка

[\[править\]](#)

Создание статического транка:

```
sw1(config)# interface fa0/22
sw1(config-if)# switchport mode trunk
```

На некоторых моделях коммутаторов (на которых поддерживается ISL), после попытки перевести интерфейс в режим статического транка, может появиться такая ошибка:

```
sw1(config-if)# switchport mode trunk
Command rejected: An interface whose trunk encapsulation is "Auto" can not be configured to "trunk" mode.
```

Это происходит из-за того, что динамическое определение инкапсуляции (ISL или 802.1Q) работает только с динамическими режимами транка. И для того чтобы настроить статический транк, необходимо инкапсуляцию также настроить статически.

Для таких коммутаторов необходимо явно указать тип инкапсуляции для интерфейса:

```
sw1(config-if)# switchport trunk encapsulation dot1q
```

И после этого снова повторить команду настройки статического транка (switchport mode trunk).

Динамическое создание транков (DTP)

[\[править\]](#)

Dynamic Trunk Protocol (DTP) — проприетарный протокол Cisco, который позволяет коммутаторам динамически распознавать настроен ли соседний коммутатор для поднятия транка и какой протокол использовать (802.1Q или ISL).

Включен по умолчанию.

Режимы DTP на интерфейсе:

- **auto** — Порт находится в автоматическом режиме и будет переведён в состояние trunk, только если порт на другом конце находится в режиме on или desirable. Т.е. если порты на обоих концах находятся в режиме "auto", то trunk применяться не будет.
- **desirable** — Порт находится в режиме "готов перейти в состояние trunk"; периодически передает DTP-кадры порту на другом конце, запрашивая удаленный порт перейти в состояние trunk (состояние trunk будет установлено, если порт на другом конце находится в режиме on, desirable, или auto).
- **nonegotiate** — Порт готов перейти в режим trunk, но при этом не передает DTP-кадры порту на другом конце. Этот режим используется для предотвращения конфликтов с другим "не-cisco" оборудованием. В этом случае коммутатор на другом конце должен быть вручную настроен на использование trunk'a.

Перевести интерфейс в режим auto:

```
sw1 (config-if)# switchport mode dynamic auto
```

Перевести интерфейс в режим desirable:

```
sw1 (config-if)# switchport mode dynamic desirable
```

Перевести интерфейс в режим nonegotiate:

```
sw1 (config-if)# switchport nonegotiate
```

Проверить текущий режим DTP:

```
sw# show dtp interface
```

Разрешенные VLAN'ы

[править]

По умолчанию в транке разрешены все VLAN. Можно ограничить перечень VLAN, которые могут передаваться через конкретный транк.

Указать перечень разрешенных VLAN для транкового порта fa0/22:

```
sw1 (config)# interface fa0/22
sw1 (config-if)# switchport trunk allowed vlan 1-2,10,15
```

Добавление ещё одного разрешенного VLAN:

```
sw1 (config)# interface fa0/22
sw1 (config-if)# switchport trunk allowed vlan add 160
```

Удаление VLAN из списка разрешенных:

```
sw1 (config)# interface fa0/22
sw1 (config-if)# switchport trunk allowed vlan remove 160
```

Native VLAN

[править]

Основная страница: [Native VLAN](#)

В стандарте 802.1Q существует понятие native VLAN. Трафик этого VLAN передается нетегированным. По умолчанию это VLAN 1. Однако можно изменить это и указать другой VLAN как native.

Настройка VLAN 5 как native:

```
sw1 (config-if)# switchport trunk native vlan 5
```

Теперь весь трафик принадлежащий VLAN'у 5 будет передаваться через транковый интерфейс нетегированным, а весь пришедший на транковый интерфейс нетегированный трафик будет промаркирован как принадлежащий VLAN'у 5 (по умолчанию VLAN 1).

Настройка маршрутизации между VLAN

[править]

Все настройки по назначению портов в VLAN, сделанные ранее для sw1, sw2 и sw3, сохраняются. Дальнейшие настройки подразумевают использование sw3 как коммутатора 3 уровня.

При такой схеме работы никаких дополнительных настроек на маршрутизаторе не требуется. Коммутатор осуществляет маршрутизацию между сетями разных VLAN, а на маршрутизатор отправляет трафик предназначенный в другие сети.

Настройки на коммутаторе sw3:

VLAN / интерфейс 3го уровня	IP-адрес
VLAN 2	10.0.2.1 /24
VLAN 10	10.0.10.1 /24
VLAN 15	10.0.15.1 /24
Fa 0/10	192.168.1.2 /24

Включение маршрутизации на коммутаторе:

```
sw3(config)# ip routing
```

Задание адреса в VLAN. Этот адрес будет маршрутом по умолчанию для компьютеров в VLAN 2:

```
sw3(config)# interface Vlan2
sw3(config-if)# ip address 10.0.2.1 255.255.255.0
sw3(config-if)# no shutdown
```

Задание адреса в VLAN 10:

```
sw3(config)# interface Vlan10
sw3(config-if)# ip address 10.0.10.1 255.255.255.0
sw3(config-if)# no shutdown
```

Перевод интерфейса в режим 3го уровня [\[править\]](#)

Интерфейс fa0/10 соединен с маршрутизатором. Этот интерфейс можно перевести в режим 3 уровня.

Перевод fa0/10 в режим интерфейса 3 уровня и задание IP-адреса:

```
sw3(config)#interface FastEthernet 0/10
sw3(config-if)# no switchport
sw3(config-if)# ip address 192.168.1.2 255.255.255.0
sw3(config-if)# no shutdown
```

Коммутатор sw3 использует R1 как [шлюз по умолчанию](#) для рассматриваемой сети. Трафик не предназначенный сетям VLAN'ов будет передаваться на R1.

Настройка маршрута по умолчанию:

```
sw3(config) ip route 0.0.0.0 0.0.0.0 192.168.1.1
```

Просмотр информации [\[править\]](#)

Просмотр информации о транке:

```
sw1# show interface fa0/22 trunk
```

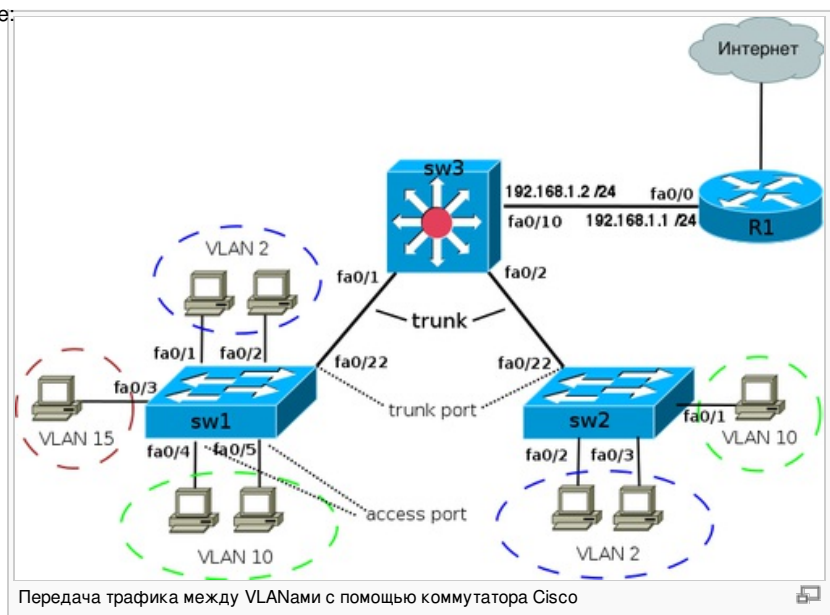
Port	Mode	Encapsulation	Status	Native vlan
Fa0/22	on	802.1q	trunking	1
Port	Vlans allowed on trunk			
Fa0/22	1-2,10,15			
Port	Vlans allowed and active in management domain			
Fa0/22	1-2,10,15			
Port	Vlans in spanning tree forwarding state and not pruned			
Fa0/22	1-2,10,15			

Просмотр информации о настройках интерфейса (о транке):

```
sw1# show interface fa0/22 switchport
Name: Fa0/22
Switchport: Enabled
Administrative Mode: trunk
Operational Mode: trunk
Administrative Trunking Encapsulation: dot1q
Operational Trunking Encapsulation: dot1q
Operational Dot1q Ethertype: 0x8100
Negotiation of Trunking: On
Access Mode VLAN: 1 (default)
Trunking Native Mode VLAN: 1 (VLAN_1)
Administrative Native VLAN tagging: enabled
Operational Native VLAN tagging: disabled
Voice VLAN: none
Administrative private-vlan host-association: none
Administrative private-vlan mapping: none
Operational private-vlan: none
Trunking VLANs Enabled: ALL
Pruning VLANs Enabled: 2-1001
Capture Mode Disabled
Capture VLANs Allowed: ALL
```

Просмотр информации о настройках интерфейса (об access-интерфейсе):

```
sw1# show interface fa0/3 switchport
Name: Fa0/3
Switchport: Enabled
Administrative Mode: static access
Operational Mode: static access
Administrative Trunking Encapsulation: negotiate
Operational Trunking Encapsulation: native
Operational Dot1q Ethertype: 0x8100
Negotiation of Trunking: Off
Access Mode VLAN: 15 (VLAN0015)
Trunking Native Mode VLAN: 1 (default)
Administrative Native VLAN tagging: enabled
Operational Native VLAN tagging: disabled
Voice VLAN: none
Administrative private-vlan host-association: none
Administrative private-vlan mapping: none
```



Operational private-vlan: none
Trunking VLANs Enabled: ALL
Pruning VLANs Enabled: 2-1001
Capture Mode Disabled
Capture VLANs Allowed: ALL

Просмотр информации о VLAN'ах:

```
sw1# show vlan brief
VLAN Name      Status Ports
-----
1  default      active Fa0/6, Fa0/7, Fa0/8, Fa0/9,
                Fa0/10, Fa0/11, Fa0/12, Fa0/13,
                Fa0/14, Fa0/15, Fa0/16, Fa0/17,
                Fa0/18, Fa0/19, Fa0/20, Fa0/21,
                Fa0/22, Fa0/23, Fa0/24
2  test         active Fa0/1, Fa0/2
10  VLAN0010     active Fa0/4, Fa0/5
15  VLAN0015     active Fa0/3
```

Диапазоны VLAN

[\[править\]](#)

VLANs	Диапазон	Использование	Передается VTP
0, 4095	Reserved	Только для системного использования.	--
1	Normal	VLAN по умолчанию. Можно использовать, но нельзя удалить.	Да
2-1001	Normal	Для VLANов Ethernet. Можно создавать, удалять и использовать.	Да
1002-1005	Normal	Для FDDI и Token Ring. Нельзя удалить.	Да
1006-4094	Extended	Только для VLANов Ethernet.	Версия 1 и 2 нет, версия 3 да

Пример настройки

[\[править\]](#)

Пример базовой настройки VLAN, без настройки маршрутизации

[\[править\]](#)

В этом разделе приведены конфигурационные файлы коммутаторов для изображенной схемы. На коммутаторе sw3 не настроена маршрутизация между VLAN, поэтому в данной схеме hosts могут общаться только в пределах одного VLAN.

Например, hosts на коммутаторе sw1 в VLAN 2 могут взаимодействовать между собой и с hosts в VLAN 2 на коммутаторе sw2. Однако, они не могут взаимодействовать с hosts в других VLAN на коммутаторах sw1 и sw2.

Не все настройки являются обязательными. Например, перечисление разрешенных VLAN в транке не является обязательным для работы транка, однако, рекомендуется настраивать разрешенные VLAN явно.

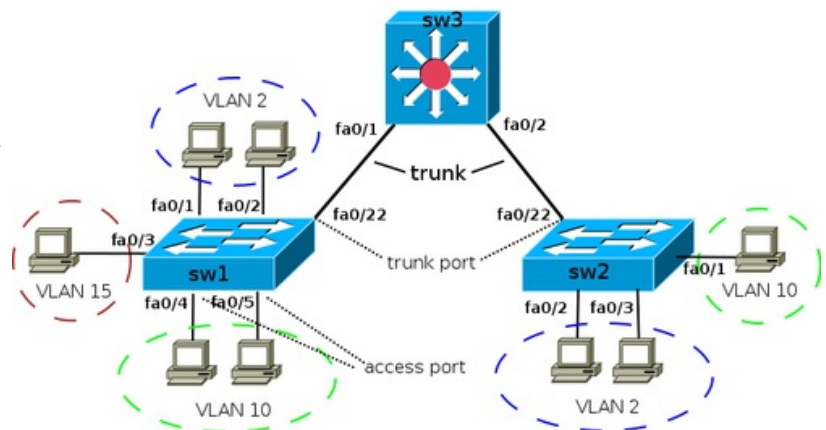
Настройки транка на sw1 и sw2 немного отличаются от sw3. На sw3 не задается инкапсуляция для транка (команда `switchport trunk encapsulation dot1q`), так как в используемой модели коммутатора поддерживается только режим 802.1Q.

Конфигурация sw1:

```
!
interface FastEthernet0/1
 switchport mode access
 switchport access vlan 2
!
interface FastEthernet0/2
 switchport mode access
 switchport access vlan 2
!
interface FastEthernet0/3
 switchport mode access
 switchport access vlan 15
!
interface FastEthernet0/4
 switchport mode access
 switchport access vlan 10
!
interface FastEthernet0/5
 switchport mode access
 switchport access vlan 10
!
interface FastEthernet0/22
 switchport trunk encapsulation dot1q
 switchport mode trunk
 switchport trunk allowed vlan 1,2,10,15
!
```

Конфигурация sw2:

```
!
interface FastEthernet0/1
```



```

!
switchport mode access
switchport access vlan 10
!
interface FastEthernet0/2
switchport mode access
switchport access vlan 2
!
interface FastEthernet0/3
switchport mode access
switchport access vlan 2
!
interface FastEthernet0/22
switchport trunk encapsulation dot1q
switchport mode trunk
switchport trunk allowed vlan 1,2,10
!

```

Конфигурация sw3:

```

!
vlan 2,10,15
!
interface FastEthernet0/1
switchport mode trunk
switchport trunk allowed vlan 1,2,10,15
!
interface FastEthernet0/2
switchport mode trunk
switchport trunk allowed vlan 1,2,10
!

```

Пример конфигураций с настройкой маршрутизации между VLAN

[\[править\]](#)

В этом разделе приведены конфигурационные файлы коммутаторов для изображенной схемы. На коммутаторе sw3 настроена маршрутизация между VLAN, поэтому в данной схеме hosts могут общаться как в пределах одного VLAN, так и между различными VLAN.

Например, hosts на коммутаторе sw1 в VLAN 2 могут взаимодействовать между собой и с hosts в VLAN 2 на коммутаторе sw2. Кроме того, они могут взаимодействовать с hosts в других VLAN на коммутаторах sw1 и sw2.

Настройки коммутаторов sw1 и sw2 остались точно такими же, как и в предыдущем разделе. Добавились дополнительные настройки только на коммутаторе sw3.

Конфигурация sw1:

```

!
interface FastEthernet0/1
switchport mode access
switchport access vlan 2
!
interface FastEthernet0/2
switchport mode access
switchport access vlan 2
!
interface FastEthernet0/3
switchport mode access
switchport access vlan 15
!
interface FastEthernet0/4
switchport mode access
switchport access vlan 10
!
interface FastEthernet0/5
switchport mode access
switchport access vlan 10
!
interface FastEthernet0/22
switchport trunk encapsulation dot1q
switchport mode trunk
switchport trunk allowed vlan 1,2,10,15
!

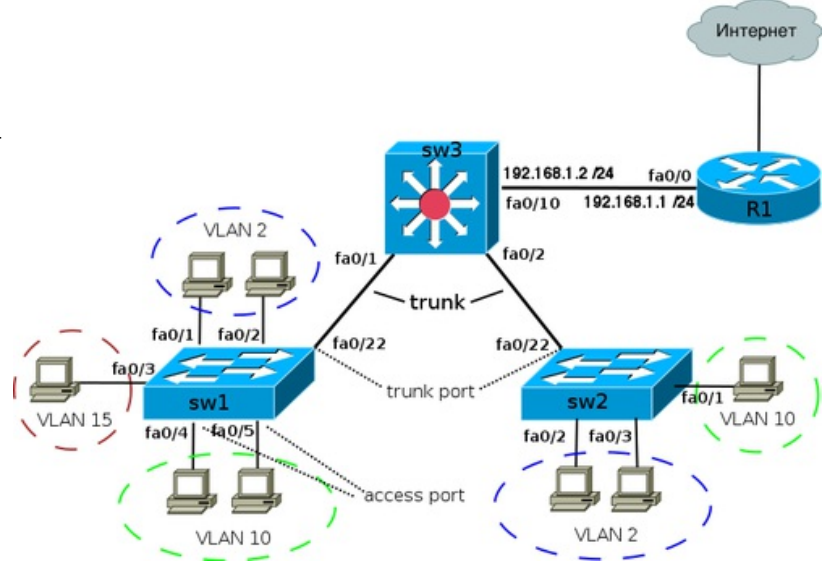
```

Конфигурация sw2:

```

!
interface FastEthernet0/1
switchport mode access
switchport access vlan 10
!
interface FastEthernet0/2
switchport mode access
switchport access vlan 2
!
interface FastEthernet0/3
switchport mode access
switchport access vlan 2
!
interface FastEthernet0/22
switchport trunk encapsulation dot1q
switchport mode trunk
switchport trunk allowed vlan 1,2,10
!

```



Конфигурация sw3:

```
!
ip routing
!
vlan 2,10,15
!
interface FastEthernet0/1
switchport mode trunk
switchport trunk allowed vlan 1,2,10,15
!
interface FastEthernet0/2
switchport mode trunk
switchport trunk allowed vlan 1,2,10
!
interface FastEthernet0/10
no switchport
ip address 192.168.1.2 255.255.255.0
!
interface Vlan2
ip address 10.0.2.1 255.255.255.0
!
interface Vlan10
ip address 10.0.10.1 255.255.255.0
!
interface Vlan15
ip address 10.0.15.1 255.255.255.0
!
ip route 0.0.0.0 0.0.0.0 192.168.1.1
```

Настройка VLAN на коммутаторах HP ProCurve

[\[править\]](#)

Основная страница: [VLAN в HP ProCurve](#)

По умолчанию на коммутаторах HP ProCurve доступно не максимальное количество VLAN. Как правило, можно создать 8 VLAN. Это можно изменить с помощью команды `max-vlans`. Указание максимального количества VLAN (максимальное значение варьируется в зависимости от модели коммутатора):

```
sw1 (config)# max-vlans <количество VLAN>
```

Для применения этой команды необходимо сохранить конфигурацию и перезагрузить коммутатор.

Создание VLAN'а, задание имени и назначение портов:

```
sw1 (config)# vlan 2
sw1 (vlan-2)# name test
sw1 (vlan-2)# untagged 1,2
sw1 (vlan-2)# tagged 22
```

```
sw1 (config)# vlan 10
sw1 (vlan-10)# untagged 4-5
sw1 (vlan-10)# tagged 22
```

Так как на коммутаторе sw2 нет VLAN'а 15, то мы не добавляем тегированный порт 22 в этот VLAN:

```
sw1 (config)# vlan 15
sw1 (vlan-15)# untagged 3
```

Просмотр информации о существующих VLAN'ах:

```
sw1# sh vlan

Status and Counters - VLAN Information

Maximum VLANs to support : 256
Primary VLAN : DEFAULT_VLAN
Management VLAN :

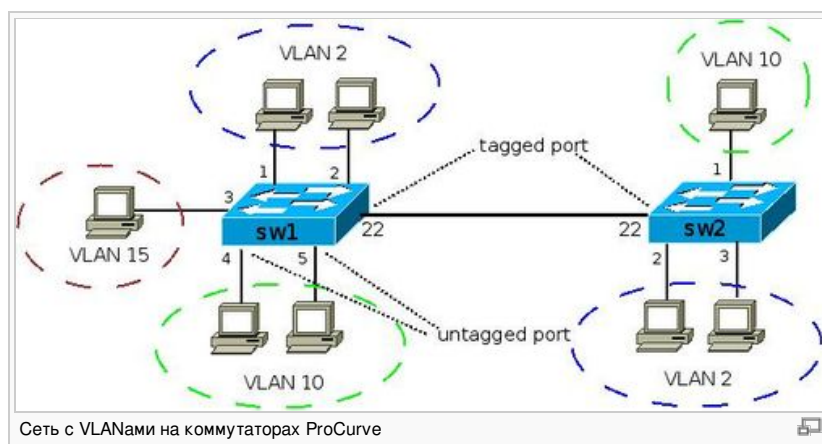
VLAN ID Name          | Status  Voice Jumbo
-----+-----
1  DEFAULT_VLAN       | Port-based No  No
2  test                | Port-based No  No
10 VLAN10              | Port-based No  No
15 VLAN15              | Port-based No  No
```

Посмотреть, какие порты принадлежат VLAN'у 2 и какой статус у порта:

```
sw1# sh vlan 2

Status and Counters - VLAN Information - Ports - VLAN 2

VLAN ID : 2
Name : test
Status : Port-based
Voice : No
```



Сеть с VLANами на коммутаторах ProCurve

Jumbo : No

Port	Information Mode	Unknown VLAN Status
------	------------------	---------------------

1	Untagged	Learn	Up
2	Untagged	Learn	Up
22	Tagged	Learn	Up

Посмотреть, каким VLAN'ам принадлежит порт 22:

```
sw1# sh vlan port 22
```

Status and Counters - VLAN Information - for ports 22

VLAN ID	Name	Status	Voice	Jumbo
1	DEFAULT_VLAN	Port-based	No	No
2	test	Port-based	No	No
10	VLAN10	Port-based	No	No

Конфигурация sw1:

```
vlan 1
 name "DEFAULT_VLAN"
 untagged 6-24
 ip address dhcp-bootp
 no untagged 1-5
 exit
vlan 2
 name "test"
 untagged 1-2
 tagged 22
 no ip address
 exit
vlan 10
 name "VLAN10"
 untagged 4-5
 tagged 22
 no ip address
 exit
vlan 15
 name "VLAN15"
 untagged 3
 no ip address
 exit
```

Настройка маршрутизации между VLAN

[\[править\]](#)

Дальнейшие настройки подразумевают использование коммутатора 3 уровня.

Для того чтобы настроить маршрутизацию между сетями разных VLAN на коммутаторе необходимо:

1. Включить ip routing
2. Назначить IP-адреса соответствующим VLAN

Кроме того, необходимо чтобы IP-адреса соответствующих VLAN были указаны как маршруты по умолчанию на хостах.

Включение маршрутизации на коммутаторе:

```
sw(config)# ip routing
```

Задание адреса в VLAN. Этот адрес должен быть прописан как маршрут по умолчанию для компьютеров в VLAN 2:

```
sw(config)# vlan 2
sw(vlan-2)# ip address 10.0.2.1/24
```

Или, другой формат задания IP-адреса в VLAN:

```
sw(config)# vlan 2 ip address 10.0.2.1/24
```

Просмотр информации о заданных IP-адресах и включен ли ip routing:

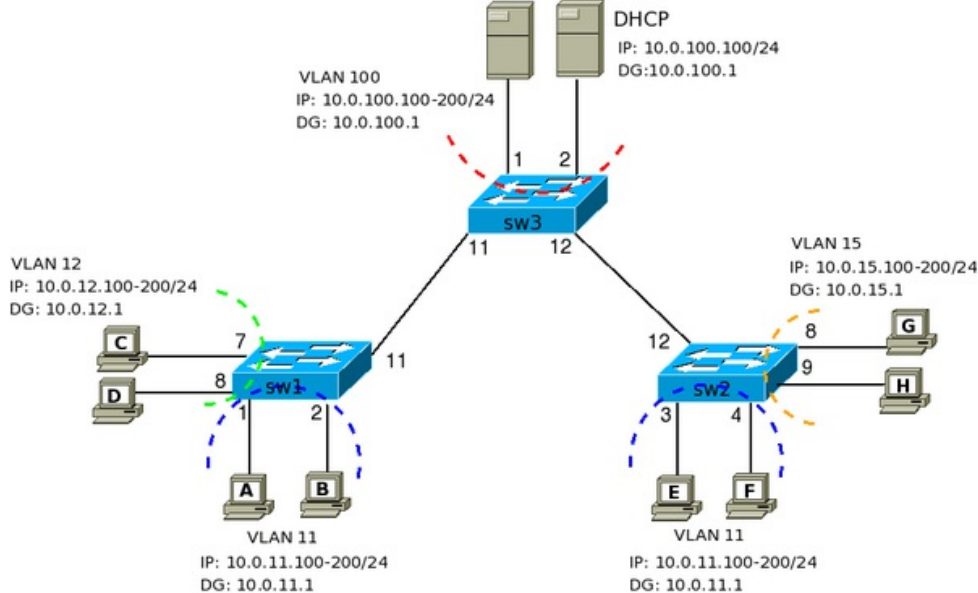
```
sw(config)# show ip
```

Таблица маршрутизации:

```
sw(config)# show ip route
```

Пример конфигурации

[\[править\]](#)



Для сети изображенной на схеме приведены примеры конфигураций коммутаторов.

Коммутаторы sw1 и sw2 работают как коммутаторы 2 уровня. Коммутатор sw3 выполняет маршрутизацию между VLAN'ами хостов и VLAN'ом серверов. Кроме того, на коммутаторе sw3 настроен DHCP relay agent, для того чтобы хосты могли получить IP-адреса и маршрут по умолчанию от DHCP-сервера (IP-адрес DHCP-сервера указан на схеме).

Конфигурация sw1:

```
hostname "sw1"
vlan 11
  name "marketing"
  untagged 1-2
  tagged 11
  exit
vlan 12
  name "managers"
  untagged 7-8
  tagged 11
  exit
```

Конфигурация sw2:

```
hostname "sw2"
vlan 11
  name "marketing"
  untagged 3-4
  tagged 12
  exit
vlan 15
  name "top-managers"
  untagged 8-9
  tagged 12
  exit
```

Конфигурация sw3:

```
hostname "sw3"
ip routing
vlan 11
  name "marketing"
  ip address 10.0.11.1 255.255.255.0
  ip helper-address 10.0.100.100
  tagged 11-12
  exit
vlan 12
  name "managers"
  ip address 10.0.12.1 255.255.255.0
  ip helper-address 10.0.100.100
  tagged 11
  exit
vlan 15
  name "top-managers"
  ip address 10.0.15.1 255.255.255.0
  ip helper-address 10.0.100.100
  tagged 12
  exit
vlan 100
  name "servers"
  ip address 10.0.100.1 255.255.255.0
  untagged 1-2
  exit
```

Настройка VLAN на коммутаторах D-LINK

[\[править\]](#)

Основная страница: [VLAN в D-LINK](#)

- create vlan *vlan_name* tag *vlan_id*
- config vlan *vlan_name* delete *port_list*
- config vlan *vlan_name* add <untagged|tagged|forbidden> *port_list*

Создание VLAN'а:

```
# create vlan v6 tag 6
```

или

```
# create vlan tag 6 desc v6
```

Удаление и добавление портов в VLAN:

```
# config vlan default delete 1-16
# config vlan v6 add untagged 1-16
# config vlan v6 add tagged 21-22
```

Просмотр информации о VLAN'ах:

```
# show vlan
VID      : 1      VLAN Name   : default
VLAN TYPE : static  Advertisement : Enabled
Member ports : 17-22,24-26
Static ports  : 17-22,24-26
Current Untagged ports : 17-22,24-26
Static Untagged ports : 17-22,24-26
Forbidden ports :
...

VID      : 6      VLAN Name   : v6
VLAN TYPE : static  Advertisement : Disabled
Member ports : 1-16,21-22
Static ports  : 1-16,21-22
Current Untagged ports : 1-16
Static Untagged ports : 1-16
Forbidden ports :
...

Total Entries : 6
```

Сохранение изменений в NVRAM:

```
# save
```

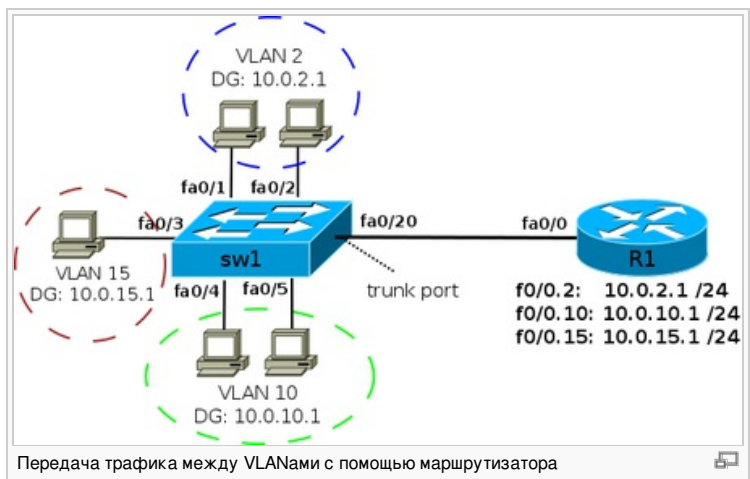
Настройка VLAN на маршрутизаторах

[\[править\]](#)

Передача трафика между VLAN может осуществляться с помощью маршрутизатора. Для того чтобы маршрутизатор мог передавать трафик из одного VLAN в другой (из одной сети в другую), необходимо чтобы в каждой сети у него был интерфейс. Для того чтобы не выделять под сеть каждого VLAN отдельный физический интерфейс, создаются логические подинтерфейсы^[1] на физическом интерфейсе для каждого VLAN.

На коммутаторе порт, ведущий к маршрутизатору, должен быть настроен как тегированный порт (в терминах Cisco — трунк).

Изображенная схема, в которой маршрутизация между VLAN выполняется на маршрутизаторе, часто называется **router on a stick**.



Передача трафика между VLANами с помощью маршрутизатора

Настройка VLAN на маршрутизаторах Cisco

[\[править\]](#)

На странице [VLAN в Cisco/Lab](#) находятся лабораторные, которые можно сделать для того чтобы на практике попробовать настройки, которые описываются в этом разделе. Лабораторные подготовлены в Packet Tracer, но аналогично могут быть выполнены и на реальном оборудовании.

IP-адреса шлюза по умолчанию для VLAN (эти адреса назначаются на подинтерфейсах маршрутизатора R1):

VLAN	IP-адрес
VLAN 2	10.0.2.1 /24
VLAN 10	10.0.10.1 /24
VLAN 15	10.0.15.1 /24

Для логических подинтерфейсов^[1] необходимо указывать то, что интерфейс будет получать тегированный трафик и указывать номер VLAN соответствующий этому интерфейсу. Это задается командой в режиме настройки подинтерфейса:

```
R1(config-if)# encapsulation dot1q <vlan-id>
```

Создание логического подинтерфейса для VLAN 2:

```
R1(config)# interface fa0/0.2
R1(config-subif)# encapsulation dot1q 2
R1(config-subif)# ip address 10.0.2.1 255.255.255.0
```

Создание логического подинтерфейса для VLAN 10:

```
R1(config)# interface fa0/0.10
R1(config-subif)# encapsulation dot1q 10
R1(config-subif)# ip address 10.0.10.1 255.255.255.0
```



Соответствие номера подинтерфейса и номера VLAN не является обязательным условием. Однако обычно номера подинтерфейсов задаются именно таким образом, чтобы упростить администрирование.

На коммутаторе порт, ведущий к маршрутизатору, должен быть настроен как статический транк:

```
interface FastEthernet0/20
switchport trunk encapsulation dot1q
switchport mode trunk
switchport trunk allowed vlan 2,10,15
```

Пример настройки

[\[править\]](#)

Конфигурационные файлы устройств для схемы изображенной в начале раздела.

Конфигурация sw1:

```
!
interface FastEthernet0/1
switchport mode access
switchport access vlan 2
!
interface FastEthernet0/2
switchport mode access
switchport access vlan 2
!
interface FastEthernet0/3
switchport mode access
switchport access vlan 15
!
interface FastEthernet0/4
switchport mode access
switchport access vlan 10
!
interface FastEthernet0/5
switchport mode access
switchport access vlan 10
!
interface FastEthernet0/20
switchport trunk encapsulation dot1q
switchport mode trunk
switchport trunk allowed vlan 2,10,15
!
```

Конфигурация R1:

```
!
interface fa0/0.2
encapsulation dot1q 2
ip address 10.0.2.1 255.255.255.0
!
interface fa0/0.10
encapsulation dot1q 10
ip address 10.0.10.1 255.255.255.0
!
interface fa0/0.15
encapsulation dot1q 15
ip address 10.0.15.1 255.255.255.0
!
```

Настройка native VLAN

[\[править\]](#)

По умолчанию трафик VLAN'a 1 передается не тегированным (то есть, VLAN 1 используется как native), поэтому на физическом интерфейсе маршрутизатора задается адрес из сети VLAN 1.

Задание адреса на физическом интерфейсе:

```
R1(config)# interface fa0/0
R1(config-if)# ip address 10.0.1.1 255.255.255.0
```

Если необходимо создать подинтерфейс для передачи не тегированного трафика, то в этом подинтерфейсе явно указывается, что он принадлежит native VLAN. Например, если native VLAN 99:

```
R1(config)# interface fa0/0.99
R1(config-subif)# encapsulation dot1q 99 native
R1(config-subif)# ip address 10.0.99.1 255.255.255.0
```

Проверка настройки

[\[править\]](#)

Для проверки настроек маршрутизации между VLAN, можно использовать такие команды:

- sh ip route -- для проверки, что присоединенные сети видны в таблице маршрутизации
- sh int -- для проверки какой VLAN соответствует подынтерфейсу
- sh vlans -- для проверки соответствия влан-подынтерфейс

Пример вывода команды sh ip route:

```
R1#sh ip route
C    10.0.2.0/24 is directly connected, FastEthernet0/0.2
C    10.0.10.0/24 is directly connected, FastEthernet0/0.10
C    10.0.15.0/24 is directly connected, FastEthernet0/0.15
```

Пример вывода sh int fa0/0.2

```
R1#sh int e0/0.10
FastEthernet0/0.10 is up, line protocol is up
Hardware is A, address is aaaa.bbbb.cccc (bia aaaa.bbbb.cccc)
Internet address is 10.0.10.1/24
MTU 1500 bytes, BW 10000 Kbit/sec, DLY 1000 usec,
  reliability 255/255, txload 1/255, rxload 1/255
Encapsulation 802.1Q Virtual LAN, Vlan ID 10.
ARP type: ARPA, ARP Timeout 04:00:00
Keepalive set (10 sec)
Last clearing of "show interface" counters never
```

Вывод sh vlans:

```
R1#sh vlans

Virtual LAN ID: 1 (IEEE 802.1Q Encapsulation)

  vLAN Trunk Interface: FastEthernet0/0

This is configured as native Vlan for the following interface(s) :
FastEthernet0/0

  Protocols Configured:  Address:      Received:  Transmitted:
    Other                0          779

335 packets, 21528 bytes input
779 packets, 52943 bytes output

Virtual LAN ID: 2 (IEEE 802.1Q Encapsulation)

  vLAN Trunk Interface:  FastEthernet0/0.2

  Protocols Configured:  Address:      Received:  Transmitted:
    IP                  10.0.2.1      21         18
    Other                0             17

2222 packets, 152116 bytes input
35 packets, 3195 bytes output

Virtual LAN ID: 10 (IEEE 802.1Q Encapsulation)

  vLAN Trunk Interface: FastEthernet0/0.10

  Protocols Configured:  Address:      Received:  Transmitted:
    IP                  10.0.10.1     0          0

329 packets, 22385 bytes input
0 packets, 0 bytes output

Virtual LAN ID: 15 (IEEE 802.1Q Encapsulation)

  vLAN Trunk Interface:  FastEthernet0/0.15

  Protocols Configured:  Address:      Received:  Transmitted:
    IP                  10.0.15.1     21         18
    Other                0             17

2222 packets, 152116 bytes input
35 packets, 3195 bytes output
```

Настройка VLAN на маршрутизаторах ProCurve

[\[править\]](#)

Настройка физического интерфейса:

```
R1(config)#interface eth 0/1
R1(config-eth 0/1)#encapsulation 802.1q
```

После задания на физическом интерфейсе инкапсуляции 802.1Q на нем нельзя задать IP-адрес (адреса задаются на подинтерфейсах^[1]).

На логических подинтерфейсах необходимо указать номер VLAN и адрес. Если подинтерфейс должен передавать нетегированный трафик, то необходимо использовать опцию native (по умолчанию native VLAN 1).

Создание логического подинтерфейса для VLAN 1:

```
R1(config)#int eth 0/1.1
R1(config-eth 0/1.1)#vlan-id 1 native
R1(config-eth 0/1.1)#ip address 10.0.1.1 /24
```

Создание логического подинтерфейса для VLAN 2:

```
R1(config)#int eth 0/1.2
R1(config-eth 0/1.2)#vlan-id 2
R1(config-eth 0/1.2)#ip address 10.0.2.1 /24
```

Пример конфигурации R1:

```
!
interface eth 0/1
 encapsulation 802.1q
 no shutdown
!
interface eth 0/1.1
 vlan-id 1 native
 no shutdown
 ip address 10.0.1.1 255.255.255.0

interface eth 0/1.2
 vlan-id 2
 no shutdown
 ip address 10.0.2.1 255.255.255.0

interface eth 0/1.10
 vlan-id 10
 no shutdown
 ip address 10.0.10.1 255.255.255.0

interface eth 0/1.15
 vlan-id 15
```

Настройка VLAN в операционных системах

[\[править\]](#)

Когда хост подключен к сети через нетегированный порт, никакой особой поддержки VLAN со стороны операционной системы не требуется. Она работает как обычно. В большинстве настольных систем именно так и происходит. Операционная система должна понимать, что такое VLAN, только тогда, когда она получает через сетевой интерфейс тегированный трафик.

Такая задача чаще всего возникает на маршрутизаторах, которые должны передавать трафик между несколькими VLAN'ами, или на нагруженных серверах, которые должны присутствовать одновременно в нескольких VLAN'ах и получать трафик из них непосредственно, а не через маршрутизатор.

В подавляющем большинстве современных UNIX/Linux-систем поддержка VLAN'ов есть. Поддержка VLAN'ов в Windows доступна при инсталляции дополнительного программного обеспечения.


Настройка VLAN в операционной системе состоит из нескольких шагов:

- Создать подинтерфейс^[1] VLAN'a;
- Привязать его к какому-либо сетевому интерфейсу как родительскому;
- Назначить IP-адрес для подинтерфейса;
- Задать дополнительные параметры подинтерфейса, если необходимо;
- Если система используется для маршрутизации трафика, разрешить прохождение трафика между интерфейсами (форвардинг) и описать правила фильтрации трафика.

Настройка VLAN в Linux

[\[править\]](#)

Основная страница: [VLAN в Linux](#)

Написана на основе [\[2\]](#) 

Ниже описывается процедура поднятия тегированного интерфейса в Linux-системе. Может применяться, например, в тех случаях, когда нужно маршрутизировать трафик между несколькими VLAN'ами или обеспечить присутствие сервера в нескольких VLAN'ах одновременно.

Необходима поддержка 802.1Q ядром Linux.

```
# modprobe 8021q
```

Если модуль не найден, необходимо переконфигурировать ядро, включив поддержку модуля, а потом пересобрать модули ядра.

Модуль включается в Network options / 802.1Q VLAN Support.

```
# make menuconfig
# make modules; make modules_install
```

Выключить интерфейс:

```
# /sbin/ifconfig eth0 down
```

Теперь поднимем его, но уже без адреса:

```
# /sbin/ifconfig eth0 0.0.0.0 up
```

Затем говорим какие VID будем пропускать. Для этого используем программу vconfig (пакет *vlan* в Debian) В результате будут созданы виртуальные интерфейсы с именами, содержащими VLAN ID.

```
интерфейс.VLAN-ID
eth0.2 (пример).
```

Имена интерфейсов могут отличаться (список возможных имён и от чего это зависит см. в [man:vconfig](#)).

```
# /sbin/vconfig add eth0 2
# /sbin/vconfig add eth0 3
# /sbin/vconfig add eth0 4
# /sbin/vconfig add eth0 5
```

Далее, назначаем каждому интерфейсу свой IP-адрес.

```
# /sbin/ifconfig eth0.2 10.x.x.x netmask 255.255.255.0 up
# /sbin/ifconfig eth0.3 192.168.0.x netmask 255.255.255.0 up
# /sbin/ifconfig eth0.4 192.168.1.x netmask 255.255.255.0 up
# /sbin/ifconfig eth0.5 x.x.x.x netmask 255.255.255.224 up
```

Если маршрут по умолчанию смотрит в один из VLAN'ов, нужно его задать:

```
# /sbin/route add default gw 192.168.1.1
```

Лучше сразу запретить пересылку трафика между интерфейсами (forwarding). Если пересылку разрешить, весь трафик между VLAN'ами может пересылаться через эту систему. Если для этого она и настраивалась, пересылку разрешайте, но помните, что вы можете ограничить прохождение трафика и регулировать его другим способом.

Для управления прохождением трафика между VLAN'ами можно использовать стандартные механизмы ядра Linux, такие как маршрутизация, iptables и QoS.

Просматривать информацию о VLAN-подынтерфейсах^[1] в Linux можно через /proc:

```
%# cat /proc/net/vlan/eth0.2
eth0.2 VID: 2 REORDER_HDR: 1 dev->priv_flags: 1
total frames received: 53973265
total bytes received: 1075877000
Broadcast/Multicast Rcvd: 397878

total frames transmitted: 41904604
total bytes transmitted: 2333267429
total headroom inc: 0
total encap on xmit: 41904604
Device: eth0
INGRESS priority mappings: 0:0 1:0 2:0 3:0 4:0 5:0 6:0 7:0
EGRESS priority Mappings:
```

Настройка VLAN при загрузке в Debian GNU/Linux

[\[править\]](#)

Основная страница: [VLAN в Debian](#)

Для работы описанного ниже способа необходимо чтобы в системе был установлен пакет **vlan**

Для того чтобы информация о созданных VLAN'ах сохранилась после перезагрузки, необходимо добавить её в файл /etc/network/interfaces. Например:

```
auto vlan1400
iface vlan1400 inet static
    address 192.168.1.1
    netmask 255.255.255.0
    vlan_raw_device eth0
```

Настройка VLAN при загрузке в CentOS

[\[править\]](#)

Основная страница: [VLAN в CentOS](#)

Для того чтобы информация о созданных VLAN'ах сохранилась после перезагрузки, необходимо создать файлы с описанием подынтерфейсов^[1] VLAN. Например, для создания подынтерфейса eth0.10 необходимо создать файл

/etc/sysconfig/network-scripts/ifcfg-eth0.10, содержимое которого будет похожем на традиционное содержимое файлов network-scripts/ifcfg-*

```
VLAN=yes
DEVICE=eth0.10
BOOTPROTO=static
ONBOOT=yes
TYPE=Ethernet
IPADDR=10.10.10.2
NETMASK=255.255.255.252
```

QinQ-инкапсуляция в Linux

[\[править\]](#)

Q-in-Q инкапсуляция позволяет создавать дважды тегированный трафик. Для каждого уровня вложенности создаётся свой собственный интерфейс.

Подробнее о процедуре настройки:

- <http://www.opennet.ru/tips/info/1381.shtml> [↗](#)

Настройка VLAN во FreeBSD

[\[править\]](#)

Основная страница: [VLAN в FreeBSD](#)

Для создания и управления VLAN во FreeBSD не используются какие-то специальные программы. Всё делается с помощью программы **ifconfig**.

Создать интерфейс vlan4:

```
# ifconfig vlan4 create
```

Указать, что созданный интерфейс соответствует трафику, тегированному VLAN 4 и приходящему через физический интерфейс fxp0:

```
# ifconfig vlan4 vlan 4 vlandev fxp0
```

Теперь можно работать с vlan4 как с обычным интерфейсом. Этому интерфейсу нужно назначить адрес. Машина будет видна в VLAN 4 по этому адресу:

```
# ifconfig vlan4 192.168.16.14/30
```

Предыдущие команды можно было бы совместить в одну.

Просмотреть информацию об интерфейсе:

```
# ifconfig vlan4
vlan4: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> mtu 1500
    inet 192.168.16.14 netmask 0xfffffc broadcast 192.168.16.15
    inet6 fe80::204:79ff:fe67:9671%vlan4 prefixlen 64 scopeid 0x6
    ether 00:07:e9:45:0d:2c
    media: Ethernet autoselect (100baseTX <full-duplex>)
    status: active
    vlan: 4 parent interface: fxp0
```

Дополнительная информация:

- http://people.freebsd.org/~arved/vlan/vlan_en.html [↗](#) (англ.)

Для создания и управления VLAN в Solaris так же, как и во FreeBSD, не используются какие-то специальные программы. Всё делается с помощью программы **ifconfig**, но с некоторой спецификой

Создать интерфейс vlan4 на интерфейсе nge0 (именование сетевых интерфейсов таково же, как и во FreeBSD):

```
# ifconfig nge4000 plumb
```

То есть новый интерфейс обзывается в виде {имя ethernet-девайса-без_цифр}{(id вилана X 1000)+цифра_от_девайса}. Тем самым новый интерфейс уже привязывается к физическому интерфейсу.

```
Примеры:

nge0 включенный в 20й влан - nge20000
nge1 включенный в 20й влан - nge20001
```

Теперь можно работать с nge4000 как с обычным интерфейсом. Этому интерфейсу нужно назначить адрес. Машина будет видна в VLAN 4 по этому адресу:

```
# ifconfig nge4000 192.168.16.14/30
```

Просмотреть информацию об интерфейсах:

```
# ifconfig -a
# ifconfig nge4000
```

Настройка VLAN в Oracle Solaris 11 Express

[\[править\]](#)

В данной версии настройка VLAN может осуществляться с использованием утилит администрирования уровня соединения (Data Link) и транспортного уровня (IP).

Провести инвентаризацию физических интерфейсов

```
# dladm show-phys
LINK      MEDIA      STATE    SPEED  DUPLEX  DEVICE
e1000g0   Ethernet    up       100    full   e1000g0

# dladm show-link
LINK      CLASS  MTU  STATE  BRIDGE  OVER
e1000g0   phys   1500 up     --     --
```

Создать виртуальный адаптер на базе физического

```
# dladm create-vlan -l e1000g0 -v 100 vlan100
```

Создать интерфейс на базе адаптера

```
# ipadm create-if vlan100
```

Назначить IPv4 статический адрес для интерфейса

```
# ipadm create-addr -T static -a 192.168.100.2/24 vlan100/v4
```

Провести инвентаризацию интерфейсов

```
# dladm show-link
# dladm show-link
LINK      CLASS  MTU  STATE  BRIDGE  OVER
e1000g0   phys   1500 up     --     --
vlan100   vlan   1500 up     --     e1000g0

# ipadm show-if
IFNAME    STATE  CURRENT  PERSISTENT
lo0       ok     -m-v-----46 ---
vlan100   ok     bm-----46 -46

# ipadm show-addr
ADDROBJ   TYPE  STATE  ADDR
lo0/v4    static ok     127.0.0.1/8
vlan100/v4 static ok     192.168.100.2/24
```

Настройка VLAN в Windows

[\[править\]](#)

Основная страница: [VLAN в Windows](#)

В Windows нет встроенной поддержки VLAN. Нельзя создать интерфейс, который бы соответствовал отдельному VLANу, за исключением случаев, когда в системе есть специальный драйвер.

В случае использования простых сетевых карт, например, Realtek RTL8139, трафик передаётся в неизменном виде, с тегами, в операционную систему и приложениям. Например, если запустить **wireshark** и направить его на соответствующий интерфейс, он увидит трафик с тегами.

Более мощные сетевые карты обрабатывают теги 802.1Q. Они могут по-разному поступить с тегами: удалить теги, удалить тегированные фреймы или сделать что-нибудь ещё.

Как правило, в этом случае существуют механизм, позволяющий отключить обработку тегов сетевой картой, сделать её более "тупой" и доставлять трафик вместе с тегами. Обычно, это делается через registry, указанием соответствующего ключа.

Например, для карт Intel в ветке

HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Control\Class\{4D36E972-E325-11CE-BFC1-08002BE10318}\00xx

нужно установить ключ

MonitorModeEnabled= 1

Здесь, xx — это номер сетевого адаптера в системе (подробнее [3] [↗](#)).

Ещё существуют специализированные драйверы от Intel, Broadcom, 3Com и SysKonnect, которые добавляют поддержку VLANов (виртуальные интерфейсы), агрегированных каналов, failover'a и многие другие функции. Каким образом настраивать их, нужно смотреть в документации для этих драйверов.

Примеры таких драйверов:

- 3com DynamicAccess
- Broadcom Advanced Server Program (BASP)
- Intel Advanced Networking Services (iANS)

Дополнительные вопросы

[\[править\]](#)

Автоматизированное создание VLAN

[\[править\]](#)

Статическое размещение портов коммутатора в отдельных VLAN'ах становится утомительным занятием уже со второго порта. Решение этой задачи можно автоматизировать одним из нескольких способов:

1. Использовать специальное программное обеспечение, такое как, например, [ProCurve Manager](#);
2. Использовать SNMP;
3. Использовать скрипты.

Первый вариант достаточно прост и удобен, но требует специального программного обеспечения. Кроме того, он не обладает достаточной гибкостью, в том смысле, что это программное обеспечение позволяет решать типовые распространённые задачи, но даже небольшой уход от них в сторону уже трудноосуществим или вовсе невозможен.

Специальное программное обеспечение выполняет размещение портов по VLAN'ам, фактически, отправляя соответствующие команды по SNMP. Это означает, что системный администратор может сделать это и сам с помощью таких инструментов, например, как `snmp-utils`. Однако, для этого он должен знать процедуру назначения настроек VLAN по SNMP. Для различного сетевого оборудования она варьируется.

Самый простой способ — использовать известные уже команды по настройке оборудования, только автоматизировать их ввод.

К сожалению, простейший удалённый вызов команды в стиле UNIX

```
ssh switch 'ena; conf t; vlan 10; tagged 20'
```

невозможен. Аналогичные команды где используется перевод строки вместо символа `;` тоже не дадут желаемого эффекта.

Единственный рабочий способ — сформировать поток команд, которые должны быть выполнены на коммутаторе, и каким-то образом передать их ему. Самый простой (но не единственный) способ — передать команды через SSH/Telnet или консольный интерфейс.

Однако, напрямую передать не получится. Понадобится несколько небольших ухищрений:

1. Мы не можем передать команды для активного оборудования на стандартный поток ввода (то есть `"| ssh switch sh -s"`, очевидно, не работает). Необходимо использовать программу, взаимодействующую через псевдотерминал. С этой задачей может справиться **socat** (но не только он);
2. Если отправить всем скопом большой поток команд на терминал коммутатора через **socat** (или другое аналогичное средство), он может их просто не успеть обработать, и часть команд будет потеряна. Поэтому, команды вводятся не сплошным потоком, а разделяются паузами.

Для обхода указанных ограничений типичных командных оболочек и упрощения автоматизированной работы чаще всего используются средства более развитых скриптовых языков. Для Perl существует модуль **Net::Telnet** [Описание на `www.cpan.org`](#) [↗](#) Для Python существует библиотека **telnetlib** [Описание на `www.python.org`](#) [↗](#) Также существуют аналогичные модули для работы с FTP, SSH и другими сетевыми протоколами.

Рабочий пример настройки коммутатора с использованием **socat** ниже. Пример приведён для коммутатора ProCurve, но аналогичным образом можно выполнять настройку любого коммутатора.

Генерируется последовательность команд, которая отправляется на вход **socat**. Программа **socat** вызывает программу **ssh**, которой предоставляет терминал. Через этот терминал вводятся необходимые команды, исполняющиеся на коммутаторе. Скрипт `make-many-vlans.pl` генерирует последовательность команд, которые передаются на коммутатор. В этом примере команды генерируются и передаются скопом, без задержек между ними. Что плохо и может работать не всегда.

Код скрипта ниже.

```
%# (sleep 5; echo ${PASSWORD}); sleep 2; echo ; perl make-many-vlans.pl; sleep 2) \  
| socat - EXEC:"ssh ${SWITCH}",setsid,pty,ctty
```

Здесь:

- `PASSWORD` — пароль для доступа к коммутатору
- `switch.host.net` — доменное имя или IP-адрес коммутатора, на котором необходимо выполнить создание VLAN'ов.

Пример скрипта `make-many-vlans.pl`, который создаёт 5 различных VLAN'ов (VLAN25--VLAN30) и помещает в них отдельные порты коммутатора: (25--30)

```
<perl/>
#!/usr/bin/perl
#

my $i=0;

print "conf t\n";
for (my $i=25; $i<30; $i++) {
    print <<EOF
vlan $i
tagged $i
exit
EOF
}
print "exit\nexit\nexit\n";
```

Скрипт просто генерирует последовательность команд, которые должны вводиться в командную строку коммутатора:

```
conf t
vlan 25
tagged 25
exit
vlan 26
tagged 26
exit
vlan 27
tagged 27
exit
vlan 28
tagged 28
exit
vlan 29
tagged 29
exit
exit
exit
y
```

Аналогичный пример, но команды вводятся в коммутатор через небольшие интервалы времени.

```
<sh/>
usleep()
{
    perl -mTime::HiRes -e "Time::HiRes::usleep($1);"
}

z ()
{
    ( sleep 5;
    echo ${PASSWORD};
    sleep 2;
    echo;
    sleep 3;
    echo;
    perl make-many-vlans.pl | while read line; do
        echo $line;
        usleep 100000;
        done;
    sleep 2 ) | socat - EXEC:"ssh ${SWITCH}",setsid,pty,ctty
}
```

Пример скрипта автоматизированного создания VLAN'ов для коммутатора Cisco:

```
<sh/>
#!/bin/sh

USER=admin
PASSWORD=cisco
SWITCH=192.168.90.5

(
sleep 2
echo $PASSWORD
sleep 1
echo ena
sleep 1
echo $PASSWORD
echo conf t
for port in `seq 17 24`
do
    vlan=[101-17+$port]
    cat <<EOF
int fa0/$port
switchport mode access
switchport access vlan $vlan
exit
EOF
sleep 1
done
echo end
echo exit
echo exit
) | socat - EXEC:"ssh ${USER}@${SWITCH}",setsid,pty,ctty
```

Ещё вместо **socat** может использоваться **empty**.

- [empty - run processes and applications under pseudo-terminal \(PTY\) sessions and replace TCL/Expect with a simple shell-tool](#) [\[править\]](#) (англ.)

Динамическая настройка VLAN

[\[править\]](#)

Выше рассмотрен способ статического размещения, когда порты коммутатора заранее размещаются в отдельных VLAN'ах.

Настройка может быть в некоторых случаях выполнена динамически, на основе:

- MAC-адреса компьютера, который выполняет подключение к сети;
- Результатов аутентификации пользователя, который подключается к сети.

Для динамической настройки VLAN по MAC-адресу используется протокол VQP (VLAN Query Protocol) и специальный сервер

VMPS (VLAN Management Policy Server). Коммутатор выступает в роли клиента. Он должен поддерживать протокол VQP.

Подробнее о VMPS:

- [Troubleshooting the Catalyst VMPS Switch](#) (англ.)

Для динамической настройки VLAN в зависимости от пользователя, который подключается к сети используется стандарт 802.1X.

Основная страница: [802.1X и RADIUS](#)

(несколько слов о том, как организовать динамическое назначение VLAN с помощью 802.1x)

Проброска тегированного трафика через туннель поверх IP-сети

[\[править\]](#)

Основная страница: [OpenVPN Bridge](#)

Существует две сети, соединённые на IP-уровне между собой. Необходимо сделать так чтобы между этими сетями мог прозрачно передаваться трафик, тегированный с помощью тега 802.1Q так как если бы это была большая коммутируемая сеть.

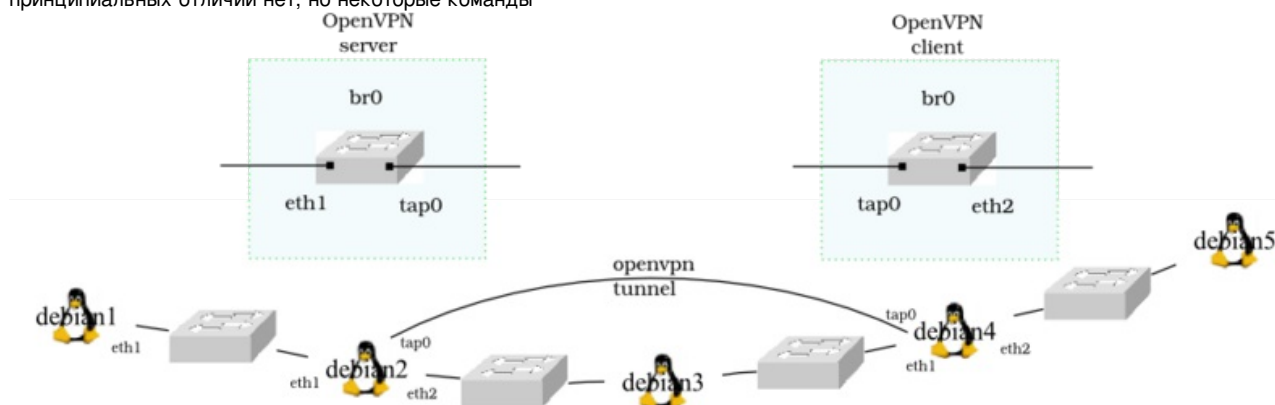
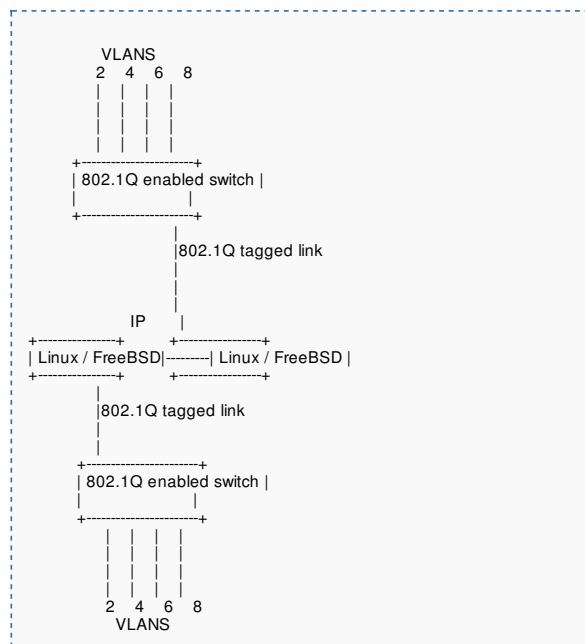
Указанную задачу можно решить с помощью различных средств построения туннелей, в частности *vtun* и *OpenVPN*. Ниже описывается решение задачи с помощью OpenVPN.

Рассмотрим топологию:

Здесь есть несколько машин, соединённых в цепочку с помощью коммутаторов.

- **debian1** — машина, находящаяся в одной локальной сети
- **debian5** — машина, находящаяся в другой локальной сети
- **debian2** — шлюз в Интернет с одной стороны
- **debian4** — шлюз в Интернет с другой стороны.
- **debian3** — условный Интернет; соединяет между собой шлюзы **debian2** и **debian4**

В качестве операционной системы используется Debian GNU/Linux. Рассматриваемая последовательность действий будет работать и в других дистрибутивах Linux. В FreeBSD принципиальных отличий нет, но некоторые команды



отличаются (настройка мостов и VLAN'ов).

Необходимо сделать так чтобы машина **debian1** видела машину **debian5** так как будто они находятся с ней в одной коммутируемой сети. Если трафик, передающийся в этой сети тегирован, теги должны сохраняться. Другими словами, если вместо машины **debian1** будет коммутатор с настроенными на нём VLAN'ами, который будет смотреть на **debian2** тегированным портом (и точно такая же ситуация будет с противоположной стороны), то компьютеры находящиеся в одном VLAN'е будут прекрасно видеть друг друга, независимо от того, в какой части сети они находятся.

Для выполнения процедуры потребуется установить программное обеспечение.

На **debian2** и **debian4**:

- `openvpn`
- `bridge-utils`
- `iproute2` (не обязательно; но тогда нужно использовать команду `ifconfig` вместо команды `ip`)

На **debian1** и **debian5**:

- `vlan`
- `tcpdump`

Настройки OpenVPN:

На **debian2** (сервер):

```
debian2%# cat /etc/openvpn/server.conf
dev tap0
secret static.key
```

На **debian4** (клиент):


```
remote 192.168.2.2
dev tap
secret static.key
```

(здесь предполагается, что 192.168.2.2 — IP-адрес машины debian2 в условном Интернете)

Как и следовало ожидать, для поднятия туннеля и его использования в режиме моста, IP-адреса не нужны.

После того как туннель поднят

```
debian2 %# /etc/init.d/openvpn start
debian4 %# /etc/init.d/openvpn start
```

на обеих машинах появляются tap-интерфейсы, однако они находятся в погашенном состоянии (ifconfig -a).

Необходимо создать мосты на обеих машинах (debian2 и debian4), и включить все необходимые интерфейсы.

На debian2:

```
debian2 %# brctl addbr br0
debian2 %# brctl addif br0 tap0
debian2 %# brctl addif br0 eth1
debian2 %# ip link set tap0 up
debian2 %# ip link set br0 up
```

На debian4:

```
debian4 %# brctl addbr br0
debian4 %# brctl addif br0 tap0
debian4 %# brctl addif br0 eth2
debian4 %# ip link set tap0 up
debian4 %# ip link set br0 up
```

Теперь машины debian1 и debian5 видят друг друга, так как если бы они находились в одной сети.

```
debian1%# ifconfig eth1
eth1  Link encap:Ethernet  HWaddr 00:16:3e:01:00:c2
       inet addr:192.168.4.1  Bcast:192.168.4.255  Mask:255.255.255.0
       UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
       RX packets:335 errors:0 dropped:0 overruns:0 frame:0
       TX packets:346 errors:0 dropped:0 overruns:0 carrier:0
       collisions:0 txqueuelen:1000
       RX bytes:24490 (23.9 KiB)  TX bytes:24736 (24.1 KiB)

debian5%# ifconfig eth1
eth1  Link encap:Ethernet  HWaddr 00:16:3e:01:04:c2
       inet addr:192.168.4.5  Bcast:192.168.4.255  Mask:255.255.255.0
       UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
       RX packets:144 errors:0 dropped:0 overruns:0 frame:0
       TX packets:132 errors:0 dropped:0 overruns:0 carrier:0
       collisions:0 txqueuelen:1000
       RX bytes:13492 (13.1 KiB)  TX bytes:12556 (12.2 KiB)
```

С машины debian1 можно пингануть машину debian5 и проследить трассу:

```
debian1%# ping 192.168.4.5
PING 192.168.4.5 (192.168.4.5) 56(84) bytes of data.
64 bytes from 192.168.4.5: icmp_seq=1 ttl=64 time=5.56 ms

--- 192.168.4.5 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 5.566/5.566/5.566/0.000 ms

debian1%# traceroute 192.168.4.5
traceroute to 192.168.4.5 (192.168.4.5), 30 hops max, 40 byte packets
1  (192.168.4.5) 108.408 ms 108.389 ms 108.382 ms
```

Трассировка показала, что debian5 находится с debian1 в одной коммутируемой сети.

Мы также можем передавать тегированный трафик.

На машине debian1:

```
debian1%# vconfig add eth1 100
debian1%# ifconfig eth1.100 192.168.100.1
```

На машине debian5:

```
debian5%# vconfig add eth1 100
debian5%# ifconfig eth1.100 192.168.100.5
```

Проверяем:

```
debian1:~# ifconfig eth1.100
eth1.100  Link encap:Ethernet  HWaddr 00:16:3e:01:00:c2
          inet addr:192.168.100.1  Bcast:192.168.100.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:34 errors:0 dropped:0 overruns:0 frame:0
          TX packets:34 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:2712 (2.6 KiB)  TX bytes:3188 (3.1 KiB)

debian1%# ping 192.168.100.5
PING 192.168.100.5 (192.168.100.5) 56(84) bytes of data.
64 bytes from 192.168.100.5: icmp_seq=1 ttl=64 time=1.34 ms
--- 192.168.100.5 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 1.346/1.346/1.346/0.000 ms
```

```
debian1%# traceroute 192.168.100.5
traceroute to 192.168.100.5 (192.168.100.5), 30 hops max, 40 byte packets
1  (192.168.100.5)  1.960 ms  1.938 ms  1.930 ms
```

В это время на debian5:

```
debian5# tcpdump -i eth1 -n
device eth1 entered promiscuous mode
audit(1203642136.881:2): dev=eth1 prom=256 old_prom=0 auid=4294967295
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth1, link-type EN10MB (Ethernet), capture size 96 bytes
01:02:29.644256 vlan 100, p 0, IP 192.168.100.1 > 192.168.100.5: ICMP echo request, id 19801, seq 1, length 64
01:02:29.644373 vlan 100, p 0, IP 192.168.100.5 > 192.168.100.1: ICMP echo reply, id 19801, seq 1, length 64
...
```

Дополнительная информация:

- [OpenVPN: Static Key Mini-HOWTO](#) ^(англ.)
- [OpenVPN: Ethernet Bridging](#) ^(англ.)

Протоколы автоматической настройки VLAN [править]

- [GVRP](#) (Generic VLAN Registration Protocol)
- [VTP](#) (VLAN Trunking Protocol)

Анализ тегированного трафика [править]

Тегированный трафик можно анализировать точно также как и обычный. Как правило, все современные анализаторы распознают теги 802.1Q, показывают их и поддерживают фильтрацию по ним.

Например, если запустить **tcpdump** на интерфейсе, по которому передаётся тегированный трафик, он его покажет.

```
%# tcpdump -i eth0 -e -n
```

Для фильтрации нужно использовать ключевое слово `vlan`. Например, если нужно увидеть трафик, тегированный тегом VLAN 100, передающийся через интерфейс `eth0` (и только его) нужно использовать команду:

```
%# tcpdump -i eth0 -n vlan 100
```

Безопасность VLAN [править]

Основная страница: [Безопасность VLAN](#)

Принято считать, что достаточно изолировать трафик внутри отдельного VLAN и становится абсолютно невозможно ни просмотреть, ни, тем более, модифицировать его другим участникам сети, которые не имеют прямого доступа к этому VLANу. В действительности, это правда только отчасти. Существует большое количество различных способов в случае некорректной настройки коммутатора заставить его направлять на порт вместо трафика одного VLAN'a тегированный трафик множества VLAN'ов.

Дополнительная информация [править]

- [IEEE 802.1Q](#) ^(англ.) — Стандарт 802.1Q
- [Ethernet frame format](#) ^(англ.) — Описание форматов фрейма Ethernet
- [VLAN Information](#) ^(англ.) — древняя статья о том, что такое VLANы и зачем они нужны
- [Tutorial on VLANs: Part 1](#) ^(англ.) — Статья о VLAN, в том числе о protocol-based VLAN.
- [VLAN Overview](#) ^(англ.) — VLAN Overview

Примечания [править]

- ↑ [1,0](#) [1,1](#) [1,2](#) [1,3](#) [1,4](#) [1,5](#) Хотя глазу приятнее видеть написание слова как *подинтерфейс*, правильное написание всё же через букву *ы*; подробнее: [\[1\]](#)

VLAN - Virtual Local Area Network [скрыть]	
Стандарты, протоколы и основные понятия	802.1Q • VLAN ID • ISL • VTP • GVRP • Native VLAN
В операционных системах	 Linux (Debian , Ubuntu , CentOS) •  FreeBSD •  Windows
В сетевом оборудовании	 Cisco •  ProCurve Networking • HP ProCurve • D-LINK • Allied Telesis • Asotel • Juniper •  ExtremeXOS
Разное	man vconfig • Безопасность VLAN • 802.1X и RADIUS • Cisco Private VLAN

Канальный уровень [показать]

Категории: [Автор Наташа Самойленко](#) | [Автор Игорь Чубин](#) | [VLAN](#) | [Канальный уровень](#)

