

Расширенная настройка DNS и DHCP в роутерах Mikrotik

 interface31.ru/tech_it/2019/05/rasshirennaya-nastroyka-dns-i-dhcp-v-routerah-mikrotik.html

Записки IT специалиста

Технический блог специалистов ООО "Интерфейс"

- [Главная](#)
- Расширенная настройка DNS и DHCP в роутерах Mikrotik

Со службами DHCP и DNS знакомы, пожалуй все, ведь это базовые службы для сетей любого размера. Но их настройка обычно сводится к установке базовых параметров и затем о них забывают. Действительно, ну что может быть в них интересного, особенно если это неполноценные сервера, а службы роутера. Это так, если говорить о бытовых роутерах, где пользователю стараются не давать в руки лишних инструментов, но RouterOS рассчитана на иную аудиторию и предоставляет широкий спектр возможностей, которыми глупо не воспользоваться.



Онлайн-курс по MikroTik

Научиться настраивать MikroTik с нуля или систематизировать уже имеющиеся знания можно на [углубленном курсе по администрированию MikroTik](#). Автор курса, сертифицированный тренер MikroTik Дмитрий Скоромнов, лично проверяет лабораторные работы и контролирует прогресс каждого своего студента. В три раза больше информации, чем в вендорской программе MTCNA, более 20 часов практики и доступ навсегда.

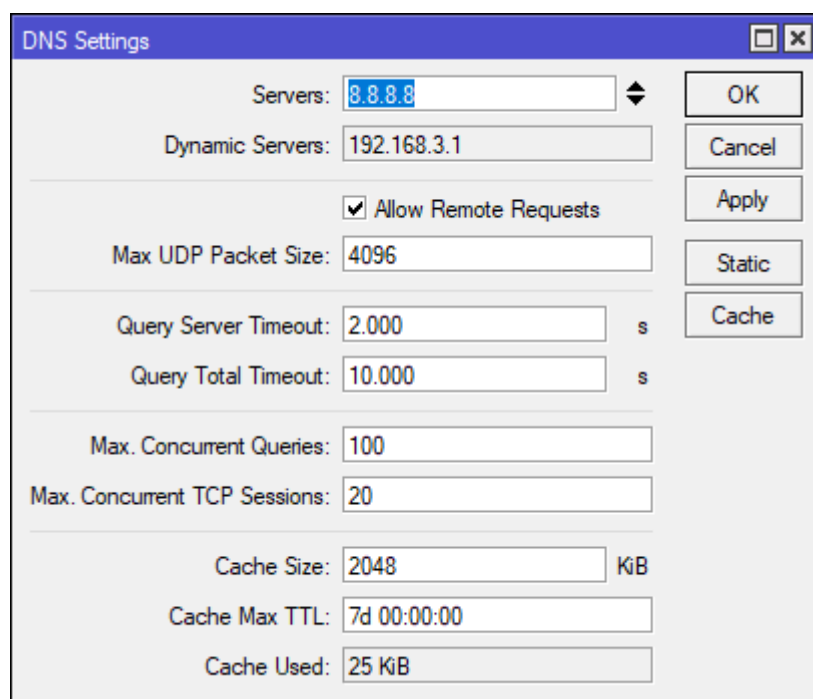
DNS (Domain Name System)

В отличие от DHCP считается, что свой DNS - это удел крупных сетей, а небольшие организации вполне могут жить и без нее, используя сервера провайдера или публичные сервера. Отчасти это так, но упуская из своих рук одну из ключевых сетевых служб администратор теряет многие инструменты контроля и управления в собственной сети.

Как работает система DNS мы рассказывали в [одной из наших статей](#), рекомендуем ее к прочтению, особенно если вы не до конца разбираетесь в вопросе. Основное, на что нужно обратить внимание - это иерархичность системы. Данные зон хранятся на собственных серверах и для получения информации из них следует прибегать к рекурсии. Это достаточно затратный процесс, поэтому многие сервера кешируют запросы и отвечают на повторные обращения самостоятельно.

Скорость ответа на DNS-запрос очень важна для комфортного использования интернета. Внешне это может проявляться как "задумчивость" браузера, который некоторое время "думает", а только потом начинает грузить страницу. При этом сама скорость канала может быть высокой, а пинг к требуемому серверу небольшим. Неопытного админа такая ситуация может сильно озадачить, но все просто - это долго отвечает используемый DNS-сервер. Ведь прежде, чем начать взаимодействие с сервером, браузер должен получить от DNS-клиента его адрес, передав последнему доменное имя.

Mikrotik дает нам возможность использовать кеширующий DNS-сервер прямо на роутере и если вы настроили использование роутера в качестве DNS - то вы его уже используете. Еще раз перейдем в **IP - DNS** и внимательно посмотрим на настройки:



Servers:	8.8.8.8	OK
Dynamic Servers:	192.168.3.1	Cancel
<input checked="" type="checkbox"/> Allow Remote Requests		Apply
Max UDP Packet Size:	4096	Static
Query Server Timeout:	2.000 s	Cache
Query Total Timeout:	10.000 s	
Max. Concurrent Queries:	100	
Max. Concurrent TCP Sessions:	20	
Cache Size:	2048 KiB	
Cache Max TTL:	7d 00:00:00	
Cache Used:	25 KiB	

В самом низу расположены настройки кеша: его размер (**Cache Size**) и максимальное время хранения записей (**Cache Max TTL**). Еще ниже располагается показатель использования кеша - **Cache Used** - который показывает его текущий размер. Если он начинает приближаться к размеру кеша, то последний следует увеличить. Просмотреть кеш можно нажав на кнопку **Cache**.

Глядя на значение TTL, можно подумать, что это очень много, целая неделя. Но это - максимальное время хранения записи, реальное время хранения определяется временем TTL в SOA-записи домена, причем Mikrotik использует минимальное значение - Minimum TTL. В этом несложно убедиться, мы очистили кеш и посетили один из своих сайтов, минимальный TTL которого равен 4 часам:

Name	Type	Data	TTL
kazak31.disqus.com	CNAME	prod.disqus.map.fastlyb....	05:58:51
kazak31.ru	A	91.250.98.27	03:59:19
kazak31.ru	NS	dns1.yandex.net	00:13:52
kazak31.ru	NS	dns2.yandex.net	00:13:52

Как видим, Mikrotik использовал значение TTL из SOA-записей, ни о каких 7 днях речи не идет. Тогда для чего нужна эта настройка? Вы можете обновлять кеш чаще, чем это указано в TTL-домена. Если значение максимального TTL Mikrotik будет меньше, чем указанное в SOA-домена, то будет использоваться именно оно.

Это может быть полезным, если вы внесли какие-либо изменения во внешнюю зону и желаете быстрее обновить кеш. Как показывает практика, публичные сервера, такие как Google, OpenDNS или Яндекс тоже часто игнорируют TTL из SOA и обновляют свой кеш чаще.

Очистить кеш можно кнопкой **Flush Cache** в окне **Cache** или командой в терминале:

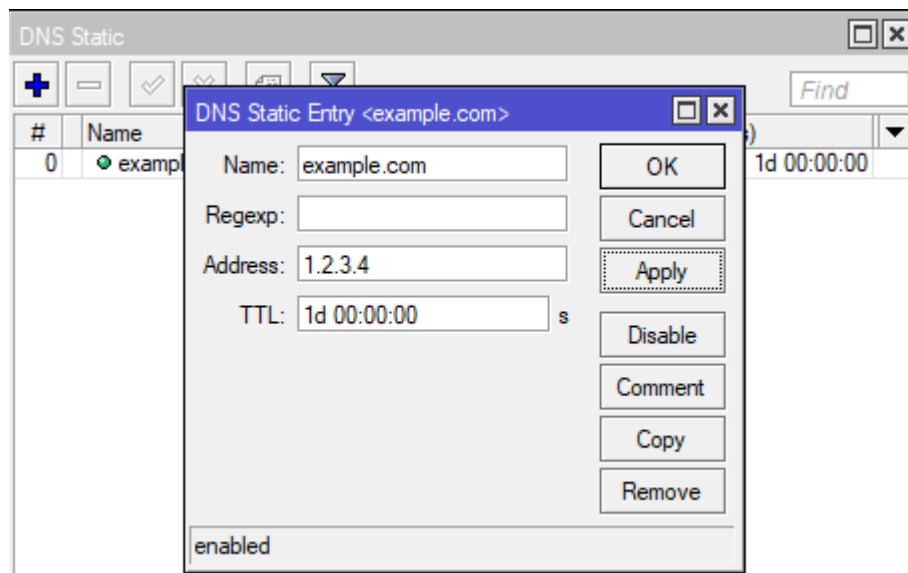
```
ip dns cache flush
```

Но это еще не всё, изучение кеша может быть полезным для изучения сетевой активности пользователей, если они используют в качестве DNS ваш роутер - то вся их сетевая активность отразится в кеше. Говорите, никто не сидит в соцсетях?

Name	Type	Data	TTL
us-west-2.elb.amazonaws.com	SOA	ns-332.awsdns-41.com	00:00:06
v10.events.data.microsoft.com	CNAME	v10.events.data.microsoft.com	00:52:12
vk.com	SOA	ns1.vkontakte.ru	00:09:43
vk.com	A	87.240.190.67	00:10:50
vk.com	A	93.186.225.193	00:10:50
vk.com	A	93.186.225.197	00:10:50
vk.com	A	87.240.129.133	00:10:50
vk.com	A	87.240.182.224	00:10:50
wac.phicdn.net	SOA	ns1.edgecastcdn.net	00:09:02
wd-prod-cp.trafficmanager.net	CNAME	wd-prod-cp-eu-west-1-fe...	00:02:22

Поэтому, если у вас нет иных инструментов логирования и статистики, то изучение записей кеша вполне позволит получить картину сетевой активности ваших пользователей.

На этом закончим с кешем и перейдем к другому разделу - **Static**. Он позволяет создавать нам собственные записи типа A (и больше ничего кроме них). Не густо, но основную часть потребностей это перекрывает. Перенесли сайт на новый сервер? Не нужно ждать пока обновятся DNS-записи, заходим в раздел **Static** и создаем собственную запись:



Проверим, как это работает. Выполним разрешение имени на клиенте:

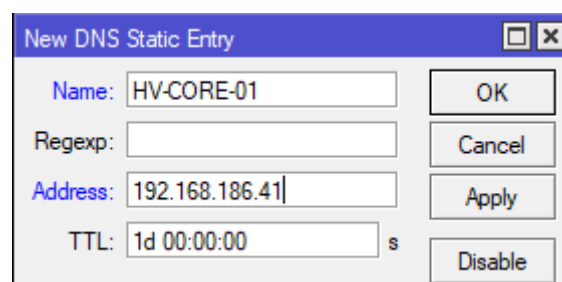
```
andrey@andrey-mint ~ $ nslookup example.com
Server:      127.0.0.53
Address:     127.0.0.53#53

Non-authoritative answer:
Name:   example.com
Address: 1.2.3.4
```

Как видим - все работает отлично.

Отдельный разговор - плоские имена. В одноранговой сети часто бывает нужно указать имя узла, который не поддерживает NetBIOS, скажем ноду Hyper-V Server или машину с Linux. Аналогично создаем запись:

Но имейте ввиду, работать такое разрешение имен будет только на Windows машинах, на Linux вы получите ошибку:



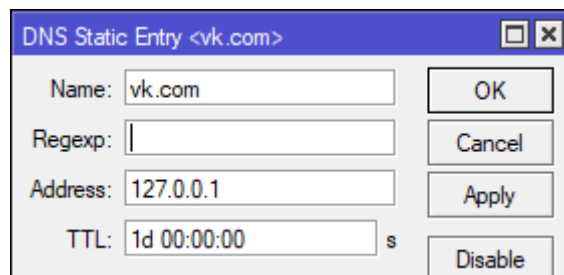
```
andrey@andrey-mint ~ $ nslookup hv-core-01
Server:      127.0.0.53
Address:     127.0.0.53#53

** server can't find hv-core-01: SERVFAIL
```

Но так как большинство сетей имеет преимущественно Windows ПК, то особых проблем плоские имена не доставят, и вы можете смело добавлять их записи на DNS Mikrotik вместо того, чтобы прописывать в **hosts** на каждой машине.

Так так, скажет внимательный читатель, это же можно использовать для блокировки нежелательных ресурсов и будет прав. Если мы не хотим, чтобы пользователи сидели в соцсетях, то добавим на сервер записи, который будут разрешать такие запросы в 127.0.0.1:

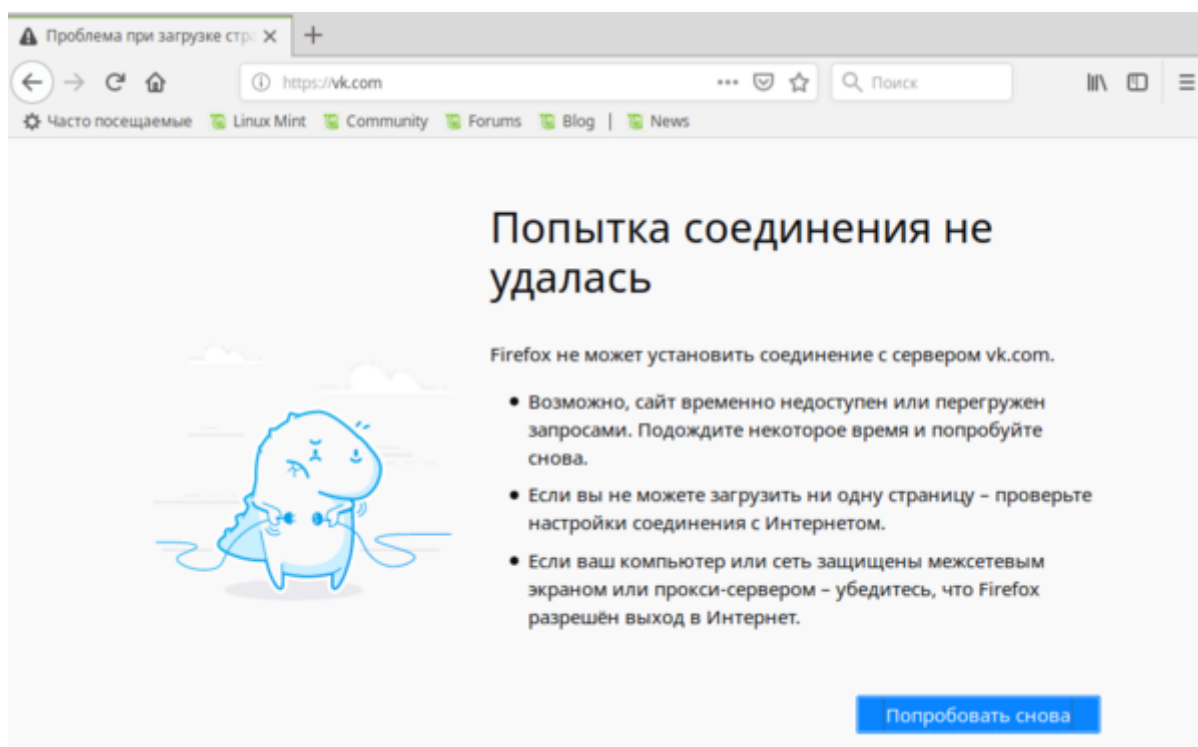
Проверим?



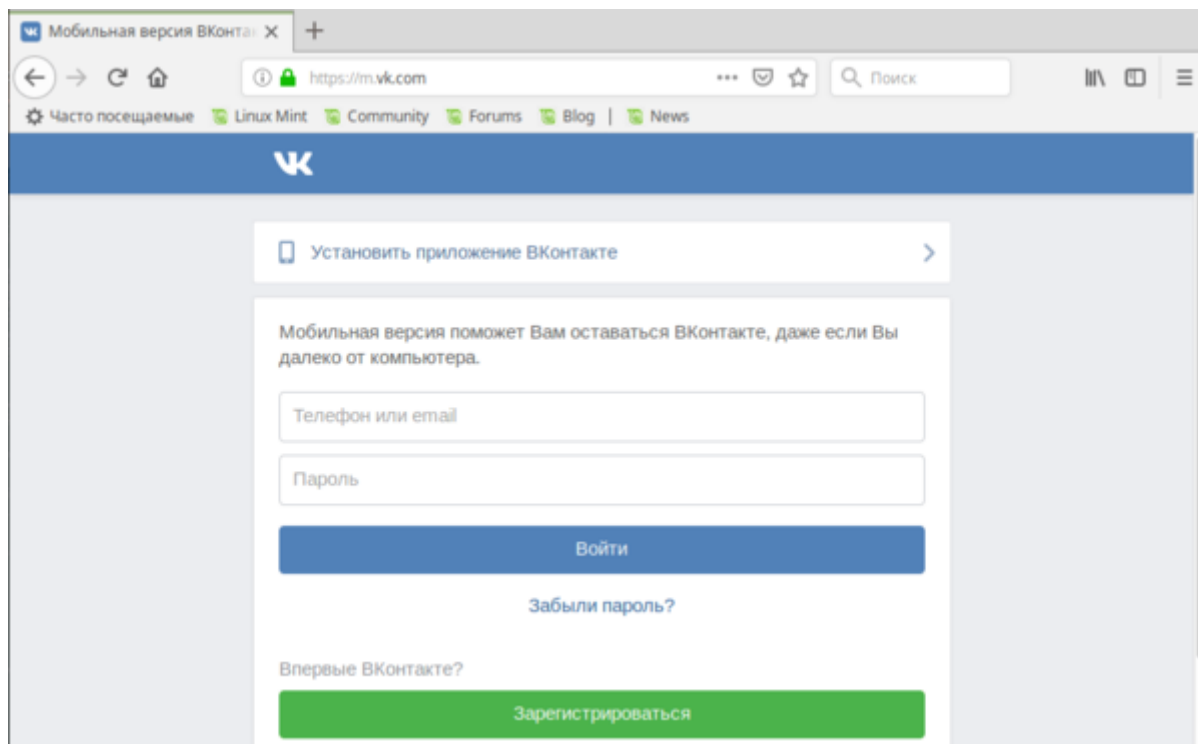
The image shows a 'DNS Static Entry' dialog box for the domain 'vk.com'. It contains the following fields and buttons:

Field	Value
Name	vk.com
Regex	
Address	127.0.0.1
TTL	1d 00:00:00 s

Buttons: OK, Cancel, Apply, Disable.



Вроде бы работает, но недостаток такого метода, что мы заблокировали только основной домен и пользователь легко сможет зайти через мобильный поддомен:



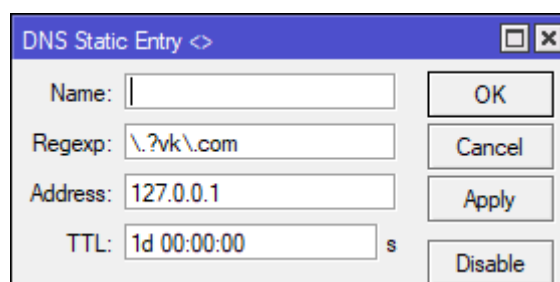
Чтобы этого избежать следует использовать регулярные выражения. К сожалению, в большинстве инструкций в интернете приводятся неправильные выражения, с которыми фильтр работать не будет, поэтому мы советуем использовать для создания и проверки регулярных выражений ресурс regex101.com. Это избавит вас от ошибок и вопросов в стиле "я сделал все как написано в статье, но ничего не работает".

Скажем, чтобы заблокировать домен vk.com со всеми поддоменами нам потребуется выражение:

```
\.?vk\.com
```

Внесем его в соответствующее поле Mikrotik:

И проверим, теперь любое, даже заведомо не существующее имя в домене vk.com будет разрешаться в 127.0.0.1, что нам и требовалось.



```

andrey@andrey-mint ~ $ nslookup m.vk.com
Server:      127.0.0.53
Address:     127.0.0.53#53

Non-authoritative answer:
Name:   m.vk.com
Address: 127.0.0.1

andrey@andrey-mint ~ $ nslookup qwerty.vk.com
Server:      127.0.0.53
Address:     127.0.0.53#53

Non-authoritative answer:
Name:   qwerty.vk.com
Address: 127.0.0.1

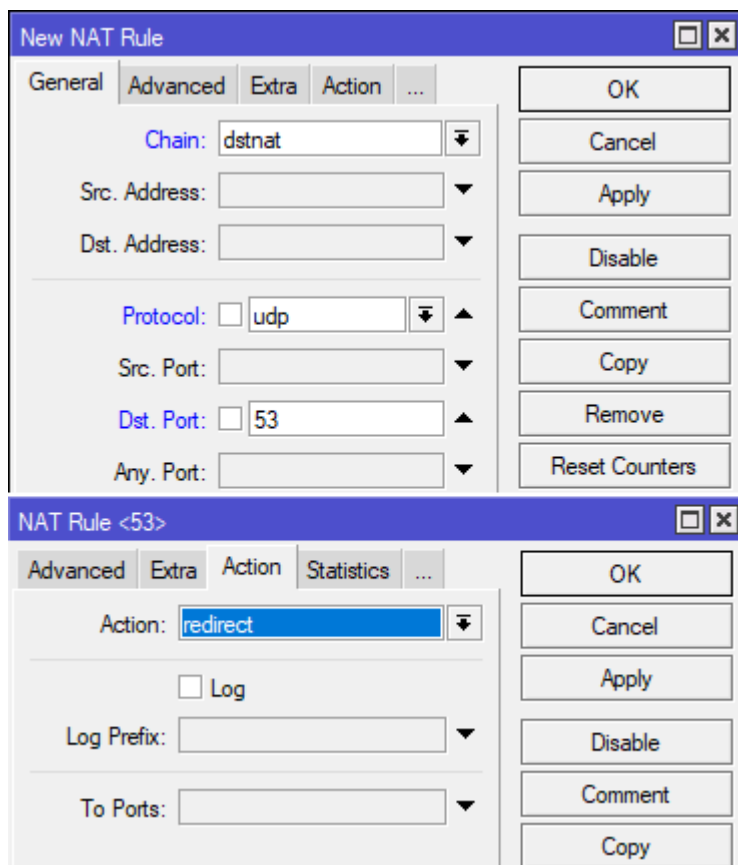
```

Но это правило заблокирует также любые ресурсы, которые заканчиваются на vk.com, для того, чтобы этого избежать, нам потребуется более сложное выражение:

```
^([A-Za-z0-9.-]*\.)?vk\.com
```

В небольших сетях, где пользователи имеют достаточно прав, всю эту идилию можно быстро перечеркнуть, вручную прописав собственные DNS. Как с этим бороться? Решение в лоб - заблокировать прохождение DNS-запросов в цепочке FORWARD, но тогда у "продвинутого" пользователя вообще перестанет работать интернет, поэтому мы поступим по-другому.

Откроем **IP - Firewall - NAT** и добавим правило: **Chain: dstnat, protocol: udp, Dst. Port: 53** и на закладке **Action** выберем действие **redirect**.



Эти же самые действия можно быстро выполнить в терминале:

```
ip firewall nat
add chain=dstnat protocol=udp dst-port=53 action=redirect
```

Теперь любые DNS-запросы от пользователей сети будут перенаправляться на наш DNS-сервер на роутере, что сделает любые попытки обойти локальный DNS бессмысленными.

Как видим, DNS-сервер роутера Mikrotik, несмотря на кажущуюся простоту, в общем не так уж и прост и дает в руки администратора достаточно широкие возможности.

DHCP (Dynamic Host Configuration Protocol)

Когда речь заходит о DHCP, то обычно имеют ввиду автоматическое присвоение сетевых параметров, таких как IP-адрес, маска, шлюз и DNS-сервера. Но на самом деле возможности протокола намного шире и позволяют настроить очень многие сетевые параметры. Все возможности протокола описаны в [RFC 2132](#), но мы не будем забираться столь глубоко, а рассмотрим в качестве примера только несколько наиболее популярных опций.

Прежде всего это **Option 15 (DNS Domain Name)** - которая позволяет автоматически настроить DNS-суффикс подключения, что позволяет снять определенный ряд проблем, связанный с использованием плоских имен.

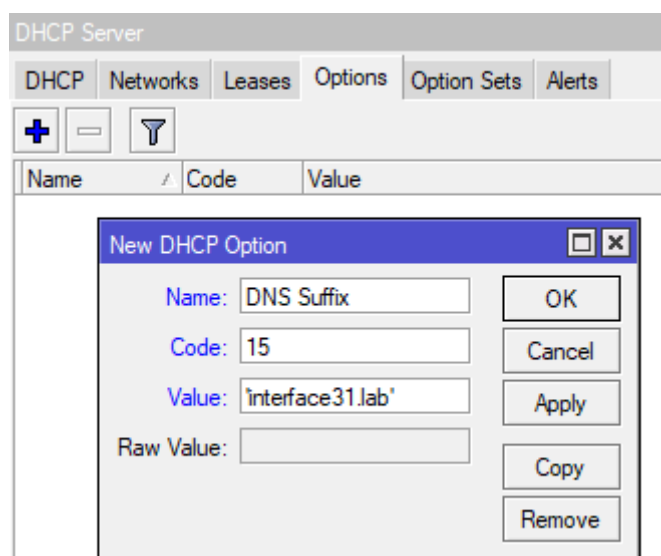
Чтобы создать любую DHCP опцию потребуется перейти в **IP - DHCP Server - Options** и добавить там новую запись:

Поле **Name** содержит имя опции, можем задать его произвольно, так чтобы нам потом было понятно, что это такое и для чего нужно. **Code** - код опции, в нашем случае **15**, **Value** - значение, в нашем случае это строка, поэтому обрамляем ее одиночной кавычкой. Тоже самое можно быстро сделать в терминале:

```
ip dhcp-server option
add name="DNS Suffix" code=15
value="'interface31.lab'"
```

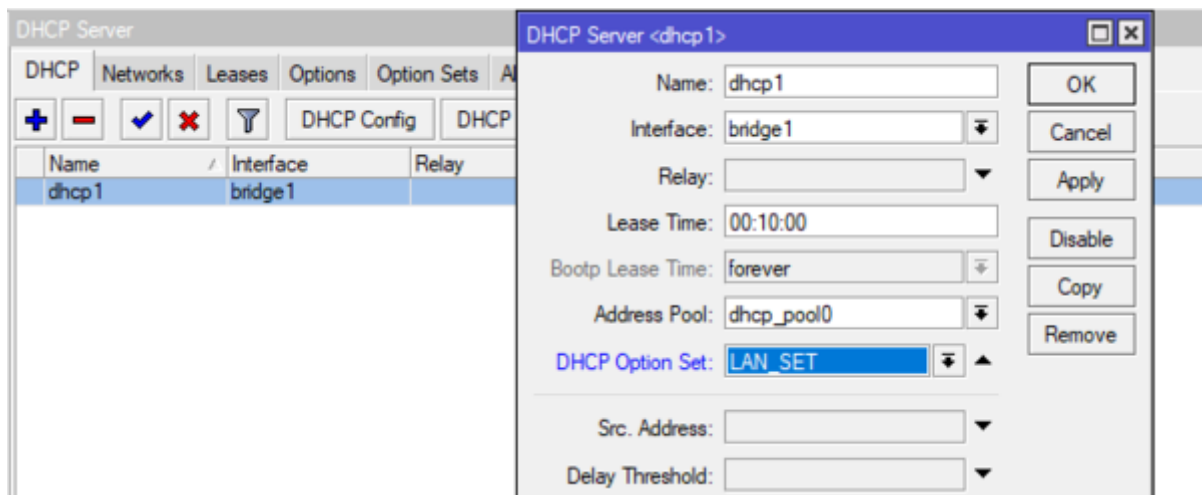
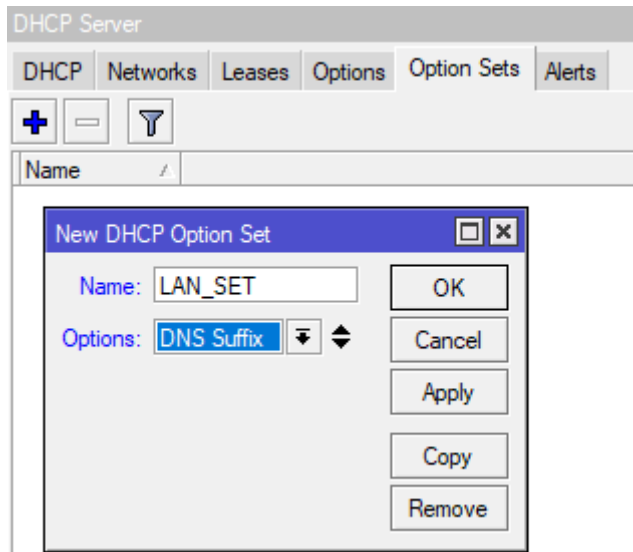
Обратите внимание, что значение **value** берется в кавычки два раза, в двойные и одинарные.

Опции можно (и нужно) объединять в наборы - **Option Set**, даже несмотря на то, что во многих сценариях их можно указывать непосредственно. Если в будущем вы надумаете что-то поменять, то достаточно будет просто изменить состав набора, в противном случае вам нужно будет вспомнить все места, где вы использовали



некую опцию и заменить ее на новую (или удалить / добавить). Перейдем на одноименную закладку и создадим новый набор. Пока в него будет входить только одна опция:

Наборам желательно давать осмысленные названия, чтобы впоследствии вы легко могли понять для чего он предназначен. Теперь назначим его для применения на всю область DHCP-сервера. Перейдем на закладку **DHCP** и откроем запись нашего сервера, в поле **DHCP Option Set** укажем имя созданного нами набора.



Теперь обновим параметры DHCP и сразу увидим полученный DNS-суффикс:

```
PS C:\Users\Andrey> ipconfig /renew

Настройка протокола IP для Windows

Невозможно выполнять операции над Ethernet, пока отключена сеть.
Невозможно выполнять операции над Ethernet 2, пока отключена сеть.
Невозможно выполнять операции над Ethernet 3, пока отключена сеть.

Адаптер Ethernet Ethernet0:

    DNS-суффикс подключения . . . . . : interface31.lab
    Локальный IPv6-адрес канала . . . . : fe80::8da2:e458:de95:730%14
    IPv4-адрес. . . . . : 192.168.186.198
    Маска подсети . . . . . : 255.255.255.0
    Основной шлюз. . . . . : 192.168.186.1
```

После этого все плоские имена, которые вы добавили на DNS-сервер следует дополнить до FQDN, т.е. вместо **HV-CORE-01** написать **hv-core-01.interface31.lab** (регистр записи значение не имеет).

DNS Static				
#	Name	Regexp	Address	TTL
0	example.com		1.2.3.4	1
1	HV-CORE-01.interface31.lab		192.168.186.41	1
2		\.?vk\.com	127.0.0.1	1
3		ok.ru	127.0.0.1	1

Проверим:

```
andrey@andrey-mint ~ $ nslookup hv-core-01
Server:      127.0.0.53
Address:     127.0.0.53#53

Non-authoritative answer:
Name:   hv-core-01.interface31.lab
Address: 192.168.186.41
```

Как видим, одной проблемой стало меньше, плоские имена нормально дополняются до FQDN и нормально разрешаются на нашем DNS вне зависимости от используемой ОС.

Также довольно часто используются опции: **42 (NTP Servers)**, **60, 66 (TFTP Server Name)**, **67 (Bootfile-Name)**. Имейте ввиду, что ОС Windows не запрашивает у DHCP-сервера опцию 42 и для нее установить таким образом сервер времени не удастся.

Отдельно коснемся того, как указывать IP-адреса. Протокол предусматривает передачу значений в шестнадцатеричном (Hex) формате, но RouterOS позволяет использовать и строковые значения, для этого значение Value должно содержать привычное написание адреса, взятое в одинарные кавычки:

```
'192.168.186.1'
```

или его шестнадцатеричное значение, которое должно начинаться с префикса 0x:

```
0xc0a8ba01
```

Если адресов несколько, то указываем каждый, взяв в одинарные кавычки, без пробелов между ними:

```
'192.168.186.1' '192.168.186.2'
```

В шестнадцатеричном виде мы добавляем второе значение в конец строки, также без пробелов:

```
0xc0a8ba01c0a8ba02
```

В терминале это будет выглядеть так:

```
ip dhcp-server option
add name="NTP1" code=42 value="'192.168.186.1'"
```

или

```
add name="NTP1" code=42 value="0xc0a8ba01"
```

Для перевода значений IP-адреса в шестнадцатеричное значение, можно использовать любой онлайн-калькулятор, мы при подготовке данного материала использовали эTOT.

Еще одна интересная возможность открывается в выдаче отдельным узлам своего набора опций. Следующий сценарий подойдет домашним пользователям, как достаточно простой и эффективный способ обеспечить безопасность ребенка в интернет.

Суть ее состоит в следующем: мы выборочно изменяем DNS-сервера детских сетевых устройств на безопасные DNS, например, **Яндекс Семейный**, **SkyDNS**, **AdGuard** и т.д. Тем, кто захочет реализовать этот сценарий в сети предприятия следует иметь ввиду, что в этом случае таким клиентам будут недоступны возможности собственного DNS-сервера, т.е. все то, о чем мы говорили в первой части статьи.

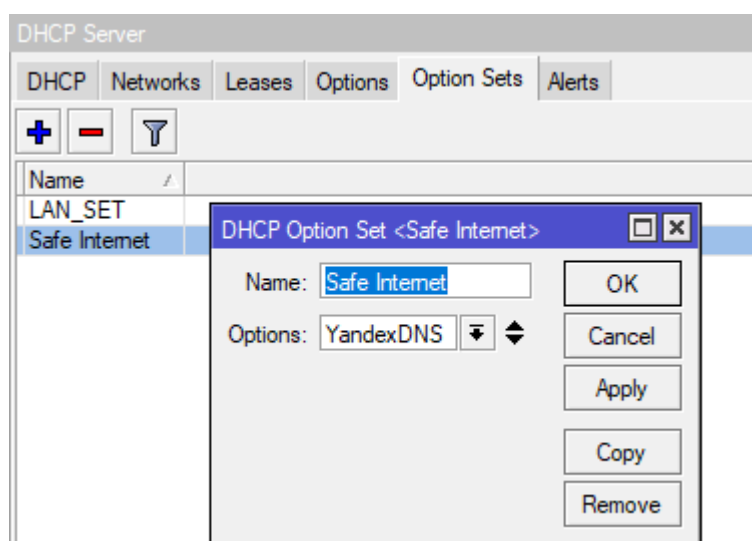
Итак, сначала создадим новую **DHCP опцию с кодом 6** и укажем в значении сервера Яндекс Семейного:

```
ip dhcp-server option  
add name="YandexDNS" code=6 value="'77.88.8.7' '77.88.8.3'"
```

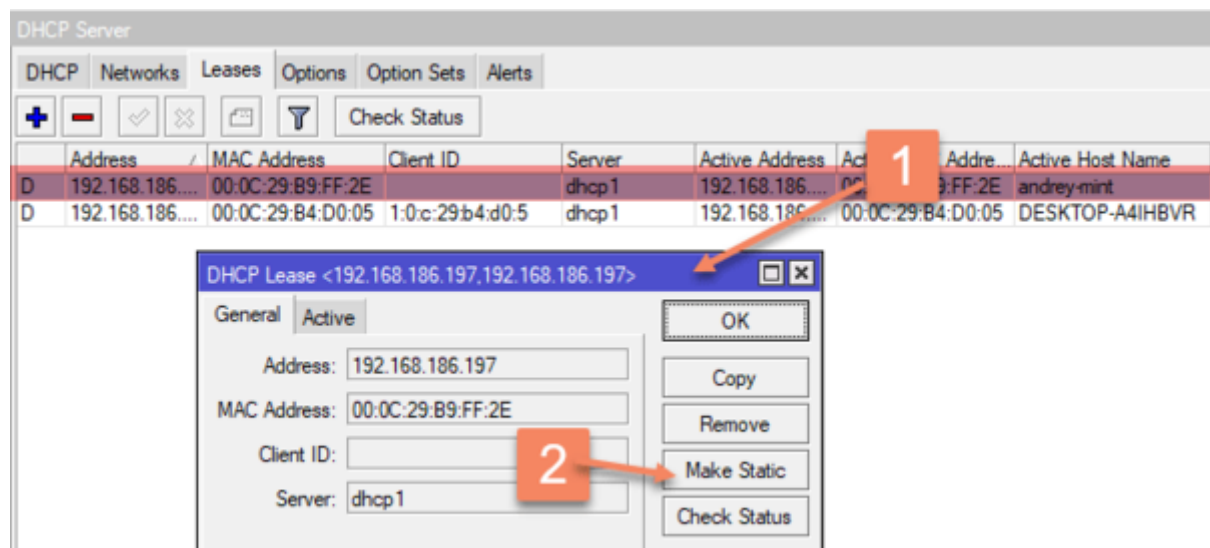
или

```
add name="YandexDNS" code=6 value="0x4d5808034d580807"
```

Теперь создадим новый набор опций и добавим туда опцию **YandexDNS**.

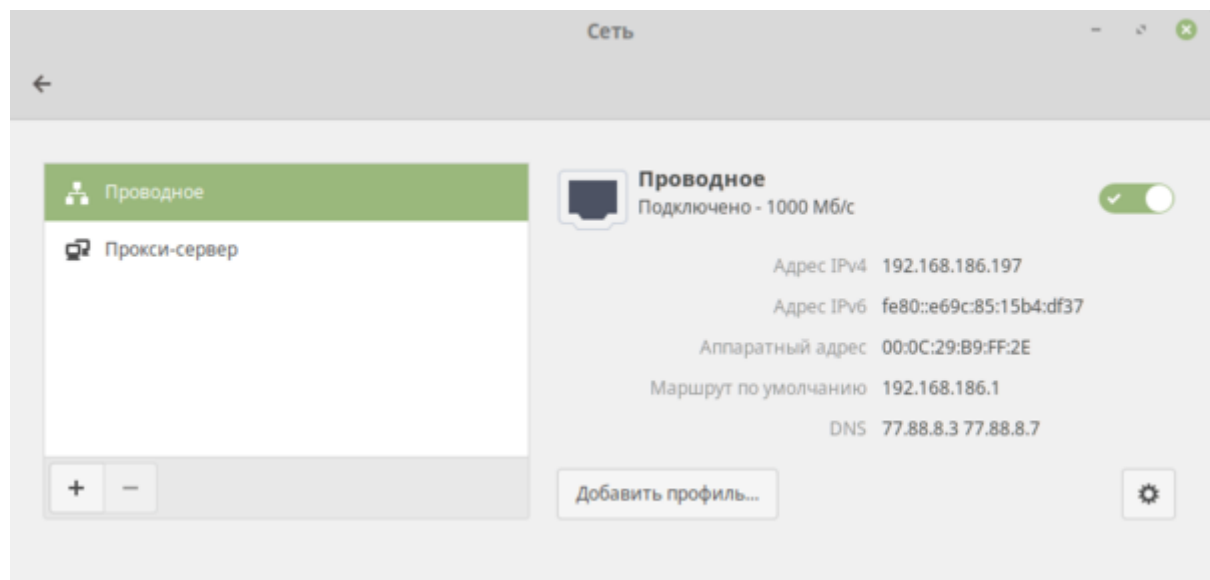
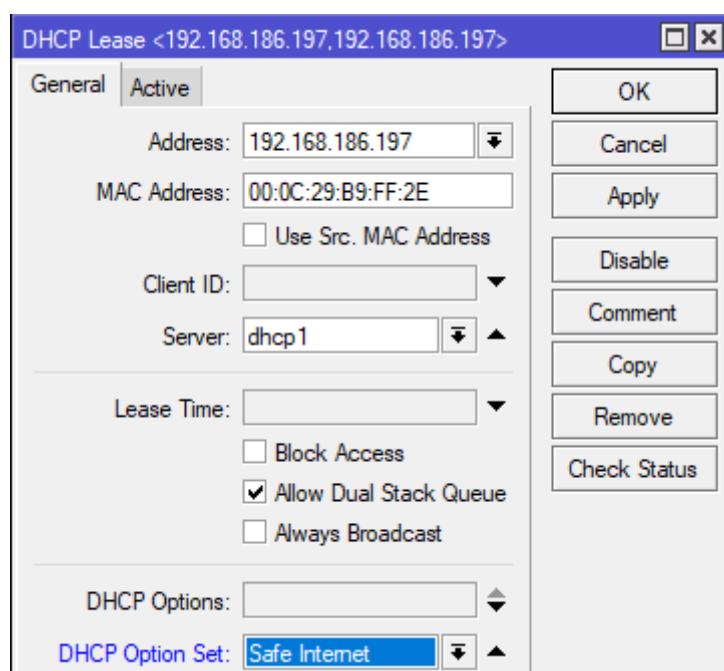


Теперь перейдем на закладку Leases, где находится список выданных в аренду адресов и найдем там детское устройство, после чего откроем запись и выполним резервирование адреса, нажав **Make Static**:

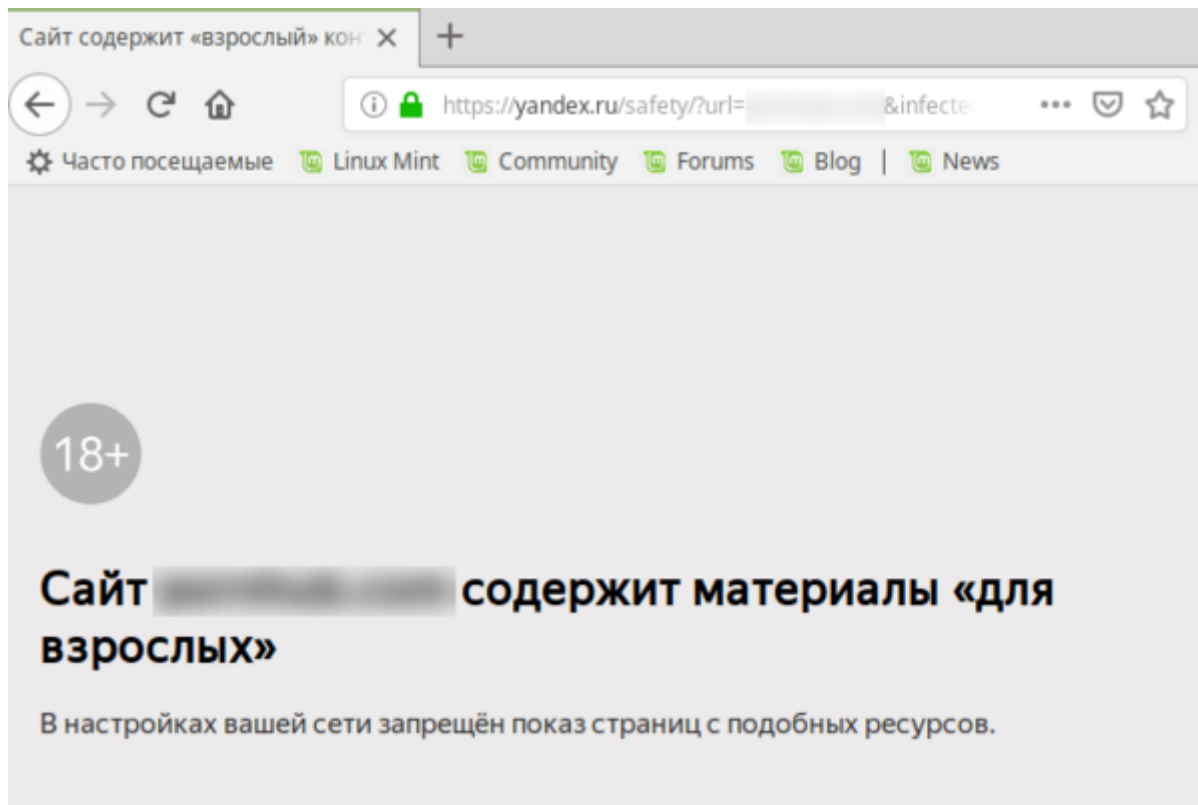


Закроем и заново откроем эту запись и в поле DHCP Option Set укажем созданный нами набор с безопасными серверами:

Теперь проверим на клиенте, какие DNS-сервера он получил:



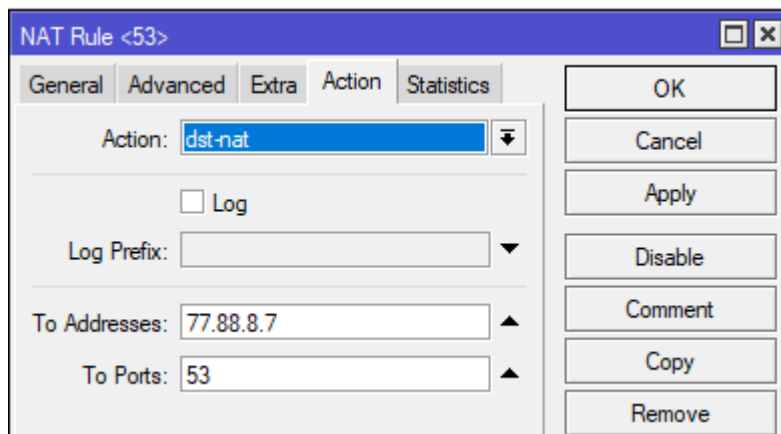
Все верно, это семейные сервера Яндекса. Попробуем посетить какой-нибудь сайт "для взрослых":



Отлично, фильтрация работает, теперь можно гораздо меньше переживать, что ребенок увидит неподобающий контент, в тоже время взрослые члены семьи могут использовать интернет без ограничений.

В прошлой части статьи мы рассказывали, как предотвратить подмену DNS на клиентском ПК, данное решение совместно с этими правилами работать не будет. Можно, конечно, добавить исключение, но мы подойдем с другой стороны. Наша основная цель в этом сценарии - это оградить ребенка от посещения ненадлежащих ресурсов, поэтому при попытке подмены DNS мы должны направлять все запросы не на роутер, а на безопасные DNS, в противном случае попытка обхода фильтрации достигнет своей цели.

Поэтому заменим в правилах действие **redirect** на действие **dst-nat**, в поле **To Addresses** указываем один из серверов семейного Яндекса, а в поле **To Ports** - порт 53.



Также можно быстро добавить нужные правила командой:

```
ip firewall nat
add chain=dstnat protocol=udp dst-port=53 action=dst-nat to-addresses=77.88.8.7
to-ports=53
```

Напоследок рассмотрим опции **121 (Classless Static Routes)** и **249 (MS Routes)**, которые предназначены для передачи статических маршрутов. Первая опция предусмотрена RFC 2132, вторая является "художественной самодеятельностью" Microsoft, поэтому следует указывать обе из них с одинаковым содержимым.

Во избежание ошибок мы советуем задавать маршруты в HEX-формате, синтаксис предусмотрен следующий:

[маска сети назначения][сеть назначения][шлюз]

Если маршрутов несколько - добавляем значения к конец строки, без пробелов. Допустим, мы хотим добавить маршрут в сеть 192.168.4.0/22 через 192.168.186.92 и в сеть 10.8.0.0/24 через 192.168.186.94. Чтобы правильно получить шестнадцатеричное значение маски следует использовать такое представление: 0.0.0.22 или 0.0.0.24, забьем все значения в онлайн калькулятор и получим:

Convert IP Address to Hexadecimal Format

Use this tool to Convert Quad IP Address (Human Readable) to Hexadecimal Format (IP to Hexa)

192.168.186.94

Convert

c0a8ba5e

Converted IP

Quad	Hexadecimal
192.168.186.94	c0a8ba5e
10.8.0.0	a080000
0.0.0.24	18
192.168.186.92	c0a8ba5c
192.168.4.0	c0a80400
0.0.0.22	16

А вот теперь начнется небольшая магия, прежде всего обратим внимание на то, что количество символов в шестнадцатеричном числе всегда должно быть четным, но в строке, соответствующей 10.8.0.0 - нечетное количество символов, так как калькулятор отбросил ведущий ноль, поэтому вместо **a080000** мы должны использовать **0a08000**. Имеем это ввиду, так как разные калькуляторы могут по-разному обрабатывать ведущий ноль.

Затем от адреса сети, мы должны отбросить столько нулевых октетов, сколько нулей содержится в маске, в HEX-значении это по два нуля. Проще говоря для сетей /24 - /17 мы должны убрать в HEX-значении сзади два нуля, для сетей /16 - /9 - четыре нуля, для сетей /8 - /1 - шесть нулей. Таким образом **0a08000** должно превратиться в **0a0800**, а **c0a80400** в **c0a804**.

Таким образом первый маршрут должен выглядеть так:

16c0a804c0a8ba5c

а второй так:

180a0800c0a8ba5e

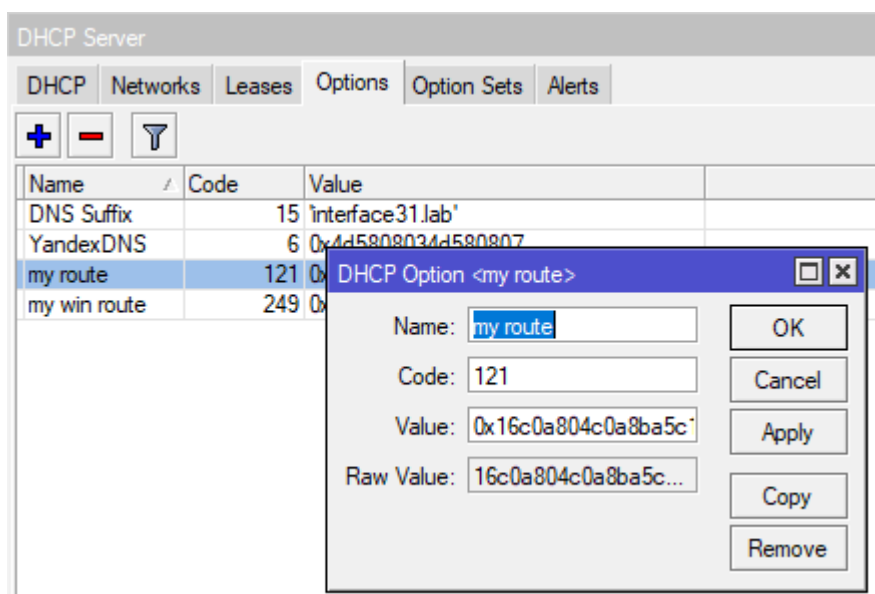
Итоговое значение будет (не забываем про 0x вначале):

0x16c0a804c0a8ba5c180a0800c0a8ba5e

Добавим опции командами:

```
ip dhcp-server option
add name="my route" code=121 value="0x16c0a804c0a8ba5c180a0800c0a8ba5e"
add name="my win route" code=249 value="0x16c0a804c0a8ba5c180a0800c0a8ba5e"
```

или через графический интерфейс:



Обновим параметры DHCP и проверим таблицы маршрутизации на клиентах.
Windows-клиент:

IPv4 таблица маршрута

=====

Активные маршруты:

Сетевой адрес	Маска сети	Адрес шлюза	Интерфейс	Метрика
0.0.0.0	0.0.0.0	192.168.186.1	192.168.186.198	25
10.8.0.0	255.255.255.0	192.168.186.94	192.168.186.198	26
127.0.0.0	255.0.0.0	On-link	127.0.0.1	331
127.0.0.1	255.255.255.255	On-link	127.0.0.1	331
127.255.255.255	255.255.255.255	On-link	127.0.0.1	331
192.168.4.0	255.255.252.0	192.168.186.92	192.168.186.198	26
192.168.186.0	255.255.255.0	On-link	192.168.186.198	281
192.168.186.198	255.255.255.255	On-link	192.168.186.198	281
192.168.186.255	255.255.255.255	On-link	192.168.186.198	281
224.0.0.0	240.0.0.0	On-link	127.0.0.1	331
224.0.0.0	240.0.0.0	On-link	192.168.186.198	281
255.255.255.255	255.255.255.255	On-link	127.0.0.1	331
255.255.255.255	255.255.255.255	On-link	192.168.186.198	281

=====

Linux-клиент:

```
andrey@andrey-mint ~ $ ip route
default via 192.168.186.1 dev eth0 proto dhcp metric 100
10.8.0.0/24 via 192.168.186.94 dev eth0 proto dhcp metric 100
169.254.0.0/16 dev eth0 scope link metric 1000
192.168.4.0/22 via 192.168.186.92 dev eth0 proto dhcp metric 100
192.168.186.0/24 dev eth0 proto kernel scope link src 192.168.186.197 metric 100
```

Как видим - все работает, маршруты добавились и еще одной заботой у администратора стало меньше.

В заключение хочется добавить, что данная статья не должна рассматриваться вами как догма, все пункты которой требуется обязательно применить. Данный материал имеет цель показать те скрытые возможности привычных сервисов, которые обычно не затрагиваются в руководствах по базовой настройке. После чего уже сам администратор должен решать, что он будет применять в своей сети и зачем. К сожалению, объем статьи не позволяет нам охватить все возможности DHCP, что-то из них мы отразим в будущих статьях, а что-то вам придется изучать самостоятельно. Но надеемся, что данный материал окажется вам полезен и даст дополнительный стимул к изучению всех возможностей используемых протоколов.

Онлайн-курс по MikroTik

Научиться настраивать MikroTik с нуля или систематизировать уже имеющиеся знания можно на [углубленном курсе по администрированию MikroTik](#). Автор курса, сертифицированный тренер MikroTik Дмитрий Скоромнов, лично проверяет лабораторные работы и контролирует прогресс каждого своего студента. В три раза больше информации, чем в вендорской программе MTCNA, более 20 часов практики и доступ навсегда.