

# Настраиваем Port Knocking в Mikrotik

 [interface31.ru/tech\\_it/2020/04/nastraivaem-port-knocking-v-mikrotik.html](https://interface31.ru/tech_it/2020/04/nastraivaem-port-knocking-v-mikrotik.html)

## Записки IT специалиста

Технический блог специалистов ООО "Интерфейс"

- [Главная](#)
- Настраиваем Port Knocking в Mikrotik

Достаточно часто системные администраторы сталкиваются с дилеммой: необходимо иметь возможность подключиться к собственным системам из любой точки мира, но при этом не выставлять открытыми наружу служебные порты. Отчасти это решается использованием VPN, но такая возможность присутствует не всегда, а в некоторых ситуациях единственной возможностью может оказаться только прямое подключение к системе. Но есть и другой способ, называемый Port Knocking, дословно обозначающий "постучать в порт", но не просто постучать, а особым образом, после чего вы сможете получить доступ, стучите - и вам откроют.



### 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.

Но сначала хочется предупредить вас о некоторых заблуждениях относительно этой технологии. Являясь инструментом, обеспечивающим дополнительную безопасность, Port Knocking никак не повышает безопасность защищаемого им сервиса. Это всего лишь способ получить доступ, причем далеко не самый надежный.

Если вы хотите закрыть таким образом, скажем, терминальный сервер со слабой политикой паролей или приложение, работающее по незащищенному протоколу - то это плохая идея. Лучше займитесь обеспечением безопасности основных сервисов.

Кроме того, стучащийся клиент может находиться в публичной сети, за VPN-сервером или прокси, да даже за NAT провайдера, в этом случае доступ к сервису, хоть и кратковременно, могут получить все пользователи, находящиеся с ним на одном IP-адресе.

Также трафик может быть перехвачен и проанализирован, что позволит вычислить правильную последовательность "стука" и чем чаще вы будете использовать Port Knocking, тем быстрее можно будет это сделать.

Поэтому мы не рекомендуем этот способ для активно используемых сетевых сервисов, кроме дополнительных сложностей вы получите достаточно призрачную защиту, а вот как инструмент сокрытия служебных портов и получения доступа к ним только в случаях крайней нужды данная технология очень даже подходит.

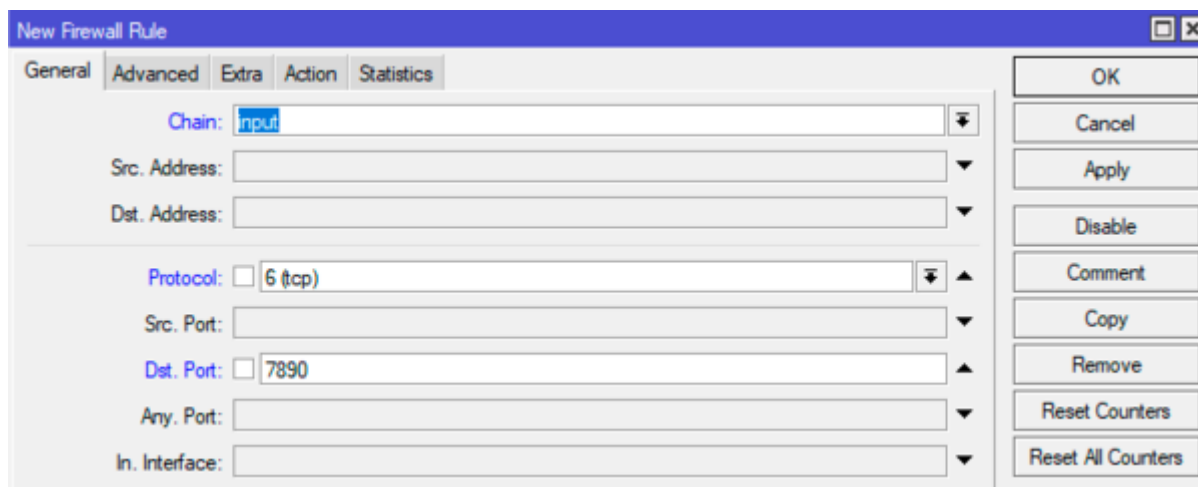
## Port Knocking с использованием портов

Этот метод можно назвать классическим, его суть сводится к обращению в определенном порядке к ряду закрытых портов за ограниченный период времени. Если мы выполнили необходимые условия - получаем доступ.

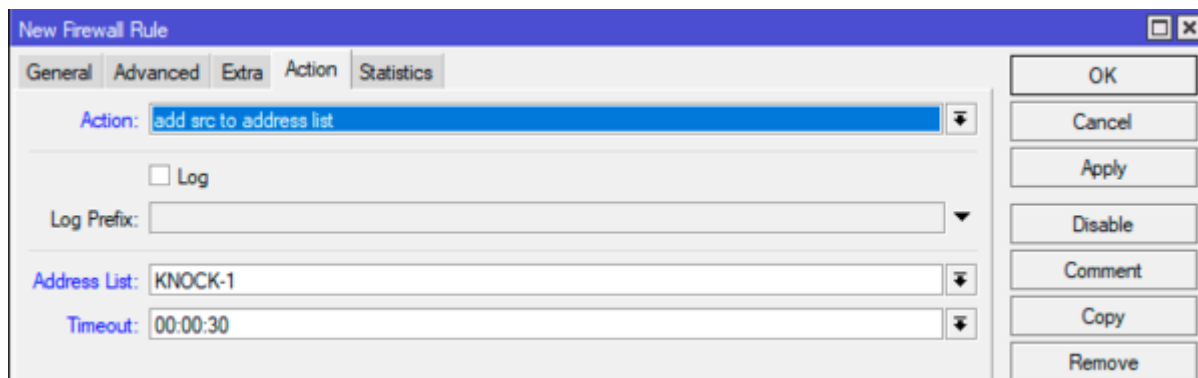
При реализации данного метода следует соблюдать правильную последовательность правил в цепочке INPUT межсетевого экрана и основным правилом, вокруг которого будет строиться наша система будет разрешение уже установленных и связанных соединений (ESTABLISHED, RELATED) и если вы настраивали роутер по нашей инструкции, то данное правило будет стоять вторым в цепочке, после "правила-пустышки" разрешающего доступ из локальной сети.

В нашем примере для получения доступа мы выберем последовательное обращение к портам 7890 и 9870, данные значения выбраны исключительно из удобочитаемости, на практике желательно разносить порты подальше друг от друга, так как такое сочетание достаточно легко "простучать" любым сканером сети, но к этому мы вернемся позже.

А пока добавим новое правило. На закладке **General** укажем: **Chain - input, Protocol - tcp, Dst. Port - 7890**.

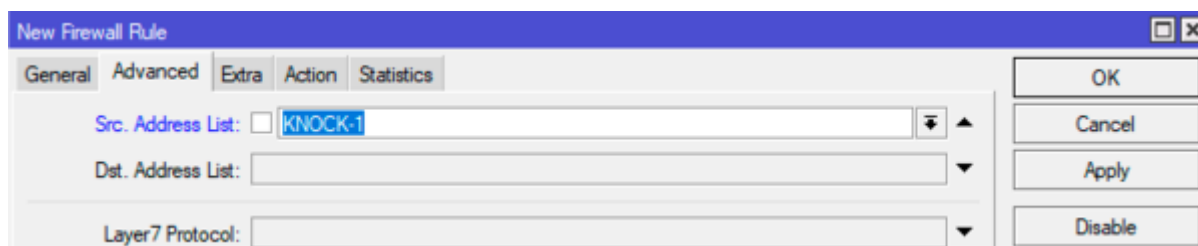


На закладке **Action** добавим действия: **Action - add src to address list, Address List - KNOCK-1, Timeout - 00:00:30**



Данное правило при обращении к порту 7890 добавит адрес источника в список KNOCK-1 на 30 секунд, в течении этого времени мы должны успеть обратиться к следующему порту, чтобы получить доступ. Поэтому не следует указывать слишком высокие значения, на наш взгляд 30 секунд вполне достаточно для выполнения нужных команд в терминале, в ряде случаев можно незначительно увеличить таймаут.

Теперь добавим еще одно правило. Закладка **General: Chain - input, Protocol - tcp, Dst. Port - 9870**. Закладка **Advanced: Src. Address List - KNOCK-1**.



Закладка **Action: Action - add src to address list, Address List - KNOCK-ACCEPT, Timeout - 00:01:00**.

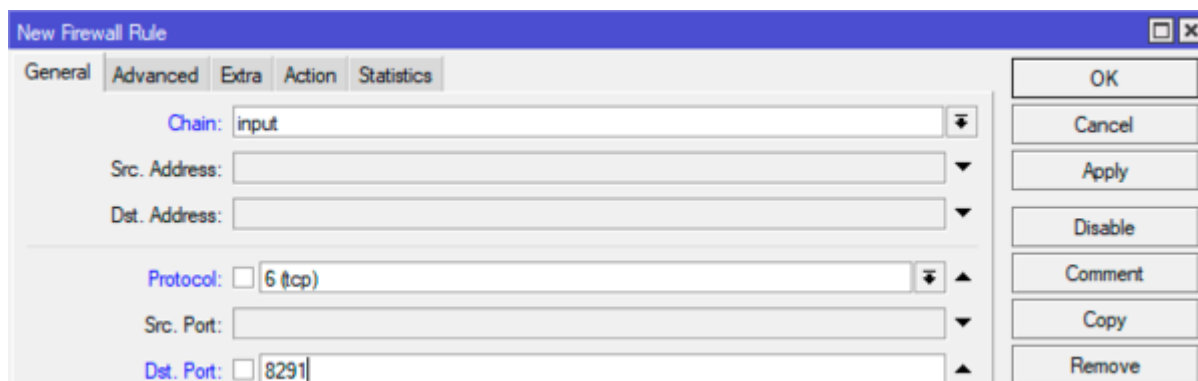
Данное правило добавит в лист KNOCK-ACCEPT адрес того источника, который обратится к порту 9870 и при этом находится в списке KNOCK-1, время жизни записи в новом листе - 1 минута, за это время мы должны будем установить соединение с устройством.

Тоже самое в терминале:

```
/ip firewall filter
add action=add-src-to-address-list address-list=KNOCK-1 address-list-timeout=30s
chain=input dst-port=7890 protocol=tcp
add action=add-src-to-address-list address-list=KNOCK-ACCEPT address-list-
timeout=1m chain=input dst-port=9870 protocol=tcp src-address-list=KNOCK-1
```

Оба этих правила следует расположить **выше** правила, разрешающего уже установленные соединения. Ниже этого правила добавим разрешающее для подключения к целевому порту, в нашем случае это будет порт Winbox - 8291.

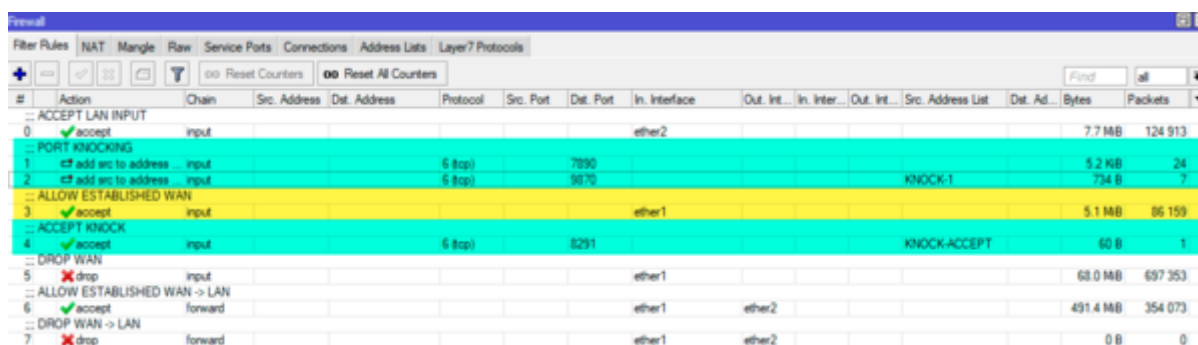
На закладке **General** укажем: **Chain - input, Protocol - tcp, Dst. Port - 8291**, на **Advanced: Src. Address List - KNOCK-ACCEPT**, вкладку **Action** не трогаем, так как **accept** действие по умолчанию.



В терминале для этого выполните:

```
/ip firewall filter
add action=accept chain=input dst-port=8291 protocol=tcp src-address-list=KNOCK-ACCEPT
```

Данное правило следует расположить **ниже** разрешающего установленные и связанные подключения.



#	Action	Chain	Src. Address	Dst. Address	Protocol	Src. Port	Dst. Port	In. Interface	Out. Int.	In. Inter.	Out. Int.	Src. Address List	Dst. Ad.	Bytes	Packets
0	✓ accept	input						ether2						7.7 MB	124 913
1	✗ add src to address	input			6 (tcp)	7890								5.2 KB	24
2	✗ add src to address	input			6 (tcp)	9870						KNOCK-1		734 B	7
3	✓ accept	input						ether1						5.1 MB	86 159
4	✓ accept	input			6 (tcp)		8291					KNOCK-ACCEPT		60 B	1
5	✗ drop	input						ether1						68.0 MB	697 353
6	✓ accept	forward						ether1	ether2					491.4 MB	354 073
7	✗ drop	forward						ether1	ether2					0 B	0

Теперь пару слов о том, как это работает. Первые два правила заполняют листы с адресом источника, чтобы попасть в последний - KNOCK-ACCEPT - нам нужно в течении 30 секунд обратиться к последовательно к портам 7890 и 9870. Затем в течении минуты мы должны подключиться к целевому порту, в нашем случае это Winbox. После чего все пакеты этого соединения пойдут по правилу, разрешающему уже установленные соединения. Таким образом по истечении минуты новых соединений установить не удастся, а уже установленные будут продолжать работать до отключения.

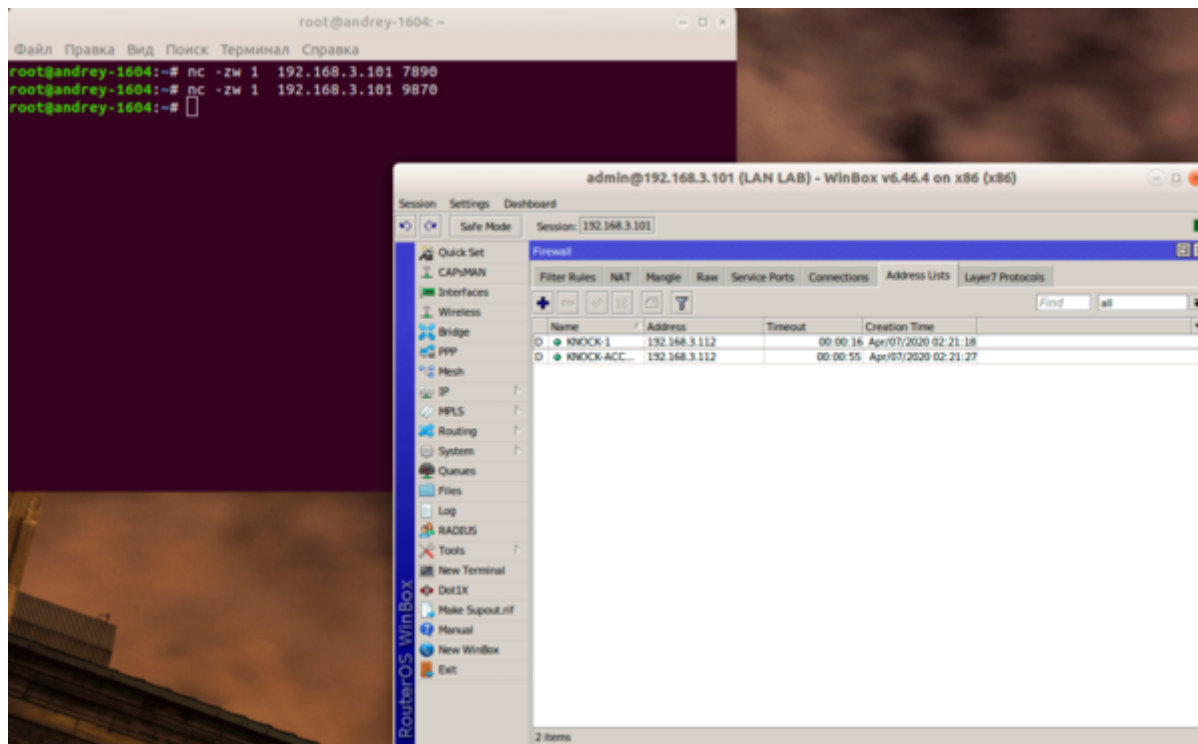
Что же, правила есть, теперь попробуем постучаться. В **Linux** для этого можно использовать команду **netcat - nc**, для того чтобы постучаться на TCP-порт используйте:

```
nc -zw 1 XXX.XXX.XXX.XXX 7890
nc -zw 1 XXX.XXX.XXX.XXX 9870
```

Где **XXX.XXX.XXX.XXX** - IP-адрес вашего роутера. Ключ **-z** указывает сканировать порт без отправки данных, ключ **-w** задает таймаут в 1 секунду.

Для UDP-портов используйте:

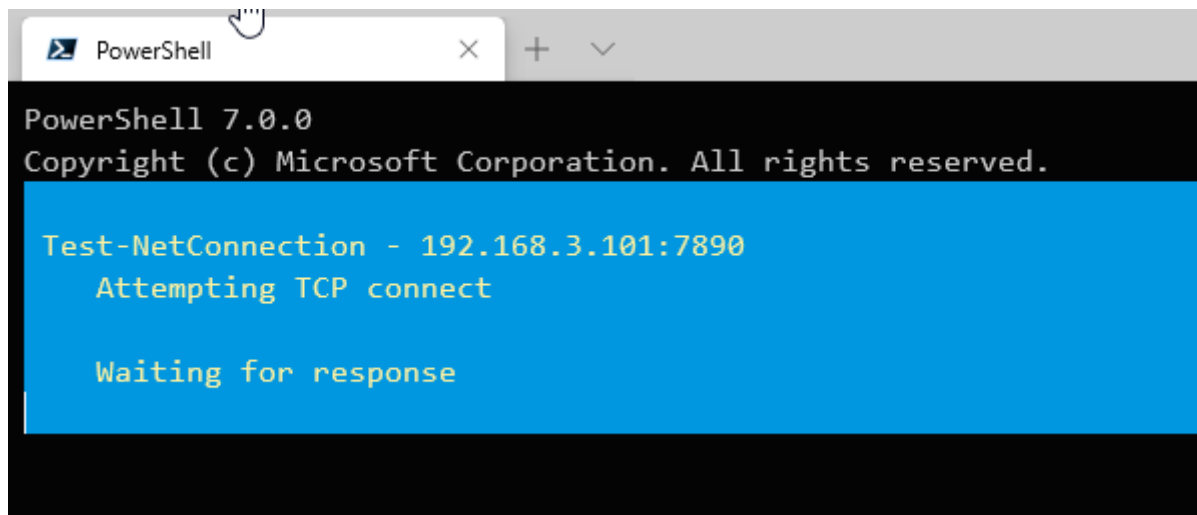
```
nc -zu XXX.XXX.XXX.XXX 7890
nc -zu XXX.XXX.XXX.XXX 9870
```



В **Windows** постучаться немного сложнее, из стандартных средств можно использовать PowerShell:

```
Test-NetConnection XXX.XXX.XXX.XXX -Port 7890
Test-NetConnection XXX.XXX.XXX.XXX -Port 9870
```

Но данная команда будет ожидать ответа от узла в течении таймаута около 20 секунд и указанного нами времени для обращения ко второму порту может не хватить, в этом случае имеет смысл увеличить время ожидания. Также таким образом можно постучать **только на TCP-порт**.

A screenshot of a PowerShell terminal window. The title bar shows 'PowerShell' with a close button. The terminal text is as follows:

```
PowerShell 7.0.0
Copyright (c) Microsoft Corporation. All rights reserved.

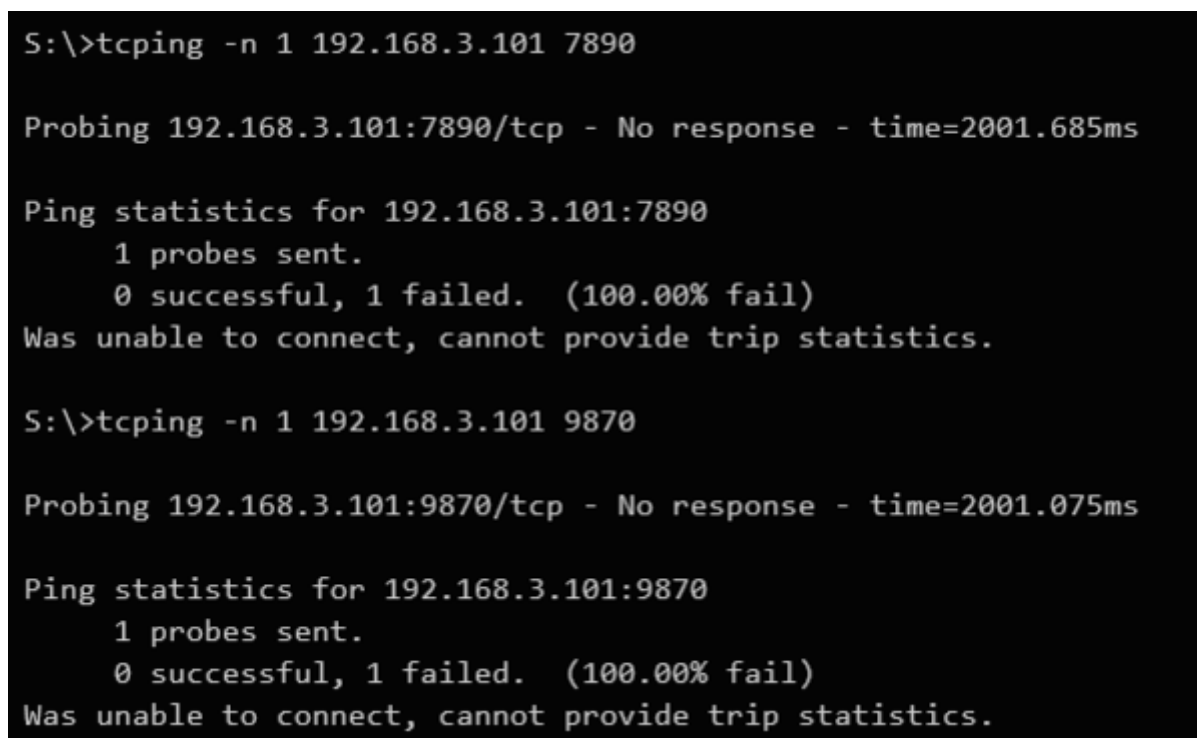
Test-NetConnection - 192.168.3.101:7890
    Attempting TCP connect

    Waiting for response
```

Другой вариант - использование утилиты tcpping, которая проще и удобнее в использовании, но позволяет опять таки стучать только в TCP-порты:

```
tcpping -n 1 XXX.XXX.XXX.XXX 7890
tcpping -n 1 XXX.XXX.XXX.XXX 9870
```

Ключ **-n** указывает количество посылаемых пакетов, в нашем случае один.

A screenshot of a Windows command prompt window. The text shown is:

```
S:\>tcpping -n 1 192.168.3.101 7890

Probing 192.168.3.101:7890/tcp - No response - time=2001.685ms

Ping statistics for 192.168.3.101:7890
    1 probes sent.
    0 successful, 1 failed. (100.00% fail)
Was unable to connect, cannot provide trip statistics.

S:\>tcpping -n 1 192.168.3.101 9870

Probing 192.168.3.101:9870/tcp - No response - time=2001.075ms

Ping statistics for 192.168.3.101:9870
    1 probes sent.
    0 successful, 1 failed. (100.00% fail)
Was unable to connect, cannot provide trip statistics.
```

Также существуют специальные утилиты, позволяющие автоматизировать этот процесс, например, PortKnock, которая умеет работать как с TCP, так и с UDP портами и может обращаться последовательно к 4 портам.

Подведем небольшие итоги, Port Knocking с использованием портов достаточно прост в реализации, позволяет строить цепочки с нужным количеством портов и времени обращения к ним, но также имеет ряд недостатков. Прежде всего это ограниченность возможностей Windows, а ситуации бывают разные и не всегда есть время и возможности искать дополнительное ПО и качать его. Также близко расположенные порты без особых проблем можно простучать сканером.

Мы провели небольшой эксперимент, простая утилита Advanced Port Scanner без проблем простучала последовательности 7890-9870 и более сложную 7890-9870-8790, но не смог справиться с 17890-9870. Поэтому не используйте близко расположенные порты и устанавливайте минимальное время действия адресов в списках.

Кроме того, подобные последовательности достаточно легко выявляются при возможности перехвата и анализа трафика, что вполне возможно при использовании публичных сетей и сторонних VPN-сервисов. Имейте это ввиду и уделите должное внимание безопасности основного сервиса, доступ к которому вы ограничиваете данной технологией.

## Port Knocking с использованием ICMP

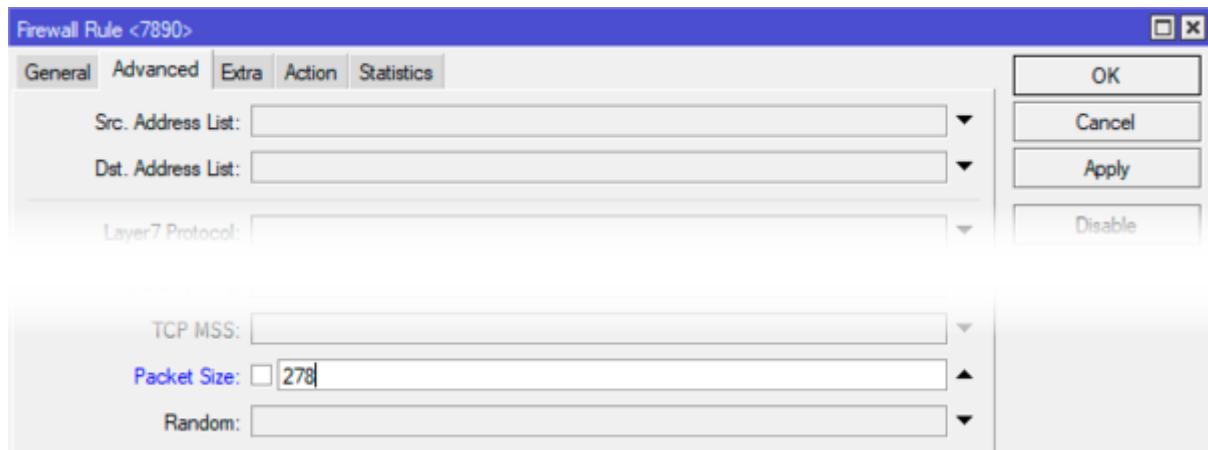
Данный вариант отличается от классического тем, что вместо определенных портов мы будем использовать ICMP-пакеты различного размера. Это проще сделать стандартными средствами ОС и труднее поддается анализу при перехвате трафика,

так как не столь бросается в глаза.

Несмотря на стандартный размер MTU в 1500 байт, это значение может быть уменьшено при использовании VLAN, VPN и т.д., поэтому мы не советуем использовать пакеты размером более 1000 байт, также учтите размер заголовков ICMP-пакета в 28 байт.

Просто выберите два (или более) произвольных числа до 1000, которые и будут вашими ключами для Port Knocking, в нашем случае это будут 250 и 209. Точно также, как и в предыдущем способе правила будут строиться относительно разрешения для уже установленных соединений.

Нам снова понадобится создать два правила для добавления адреса источника в соответствующие листы. В первом правиле укажем на **General: Chain - input, Protocol - icmp**, а на **Advanced: Packet Size - 278** (250 + 28) - размер нашего первого пакета, на **Action: Action - add src to address list, Address List - KNOCK-1, Timeout - 00:00:30**.



Во втором зададим следующие значения: **General: Chain - input, Protocol - icmp, Advanced: Src. Address List - KNOCK-1, Packet Size - 237** (209 + 28) - размер второго пакета, на **Action: Action - add src to address list, Address List - KNOCK-ACCEPT, Timeout - 00:01:00**.

Либо выполним в терминале:

```
/ip firewall filter
add action=add-src-to-address-list address-list=KNOCK-1 address-list-timeout=30s
chain=input packet-size=278 protocol=icmp
add action=add-src-to-address-list address-list=KNOCK-ACCEPT address-list-
timeout=1m chain=input packet-size=237 protocol=icmp src-address-list=KNOCK-1
```

Данные правила следует расположить **выше** правила разрешающего установленные и сопутствующие соединения. Остальные настройки полностью повторяют предыдущий способ.

Чтобы поступаться таким образом из **Linux** выполним:



```
ping XXX.XXX.XXX.XXX -s 250 -c 1  
ping XXX.XXX.XXX.XXX -s 209 -c 1
```

Где ключ **-s** задает размер пакета (без учета заголовка), а ключ **-c** количество посылаемых пакетов, в нашем случае один.

В **Windows** синтаксис будет немного иной:

```
ping XXX.XXX.XXX.XXX -l 250 -n 1  
ping XXX.XXX.XXX.XXX -l 209 -n 1
```

Здесь за размер пакета отвечает ключ **-l**, а за количество пакетов **-n**.

Принцип работы не отличается от предыдущего метода, получив ICMP-пакет размером в 250 байт адрес источника будет занесен в первый список. Затем в течении 30 секунд мы должны прислать второй пакет размером в 209 байт, в этом случае адрес будет занесен в список KNOCK-АССЕПТ и в течении минуты с устройством можно будет установить соединение.

На наш взгляд данный способ более удобен, так как позволяет использовать только штатные инструменты операционных систем и более сложен для выявления при анализе трафика. Также можно сочетать сразу оба способа, скажем, сначала постучать на выбранный порт, а затем прислать ICMP-пакет нужного размера. В любом случае выбор остается за вами и ограничивает вас только ваша фантазия и здравый смысл.

### 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.