# Port Knocking с RouterOS

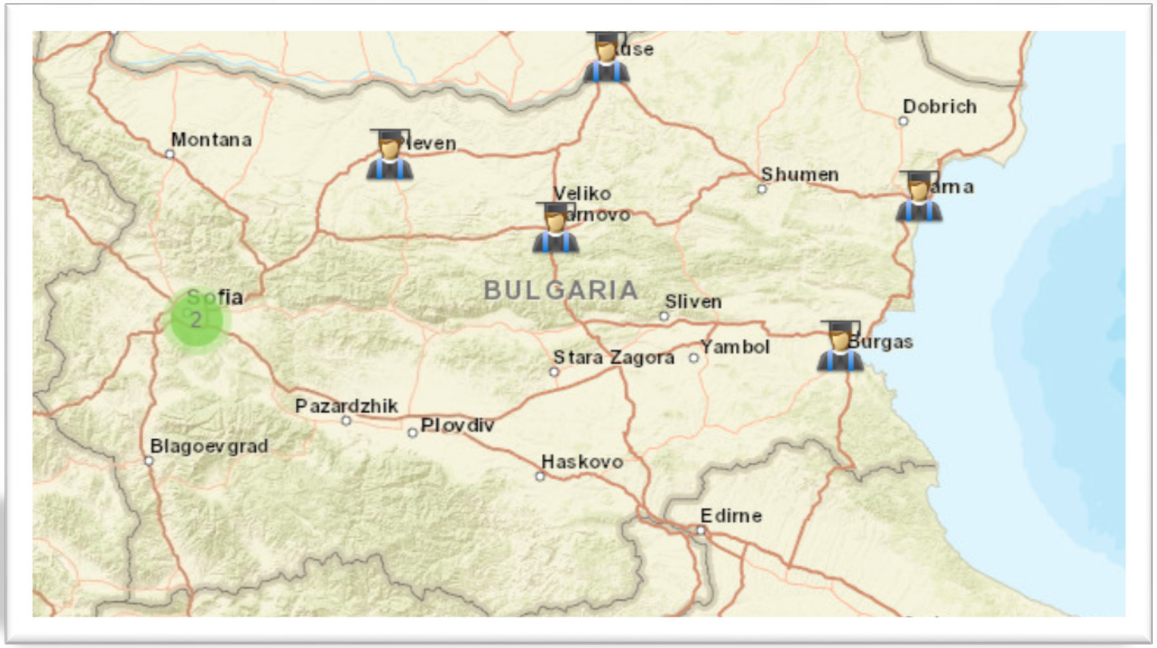
Допълнително ниво на защита
за вашата мрежа

# Добри Бояджиев

- Опит с MikroTik RouterOS **2008**
- I-ва MikroTik Академия (УниБИТ) **2014**
- Основи на MikroTik RouterOS **2016**
- Инструктор (Train the Trainer в Латвия) **2016**
- Координатор за България **2017**

# Координатор (~7000 км)

- **10** обучения (**200+** часа)
- **9** преподаватели в **5** нови академии



# Предстои да видите

- По време на презентацията се очаква да бъдат (разяснени/показани/описани):
  - Основните концепции и начин на работа
  - Различните техники и особености при изпълнение с RouterOS (сървър или клиент)
  - Добри практики и препоръки

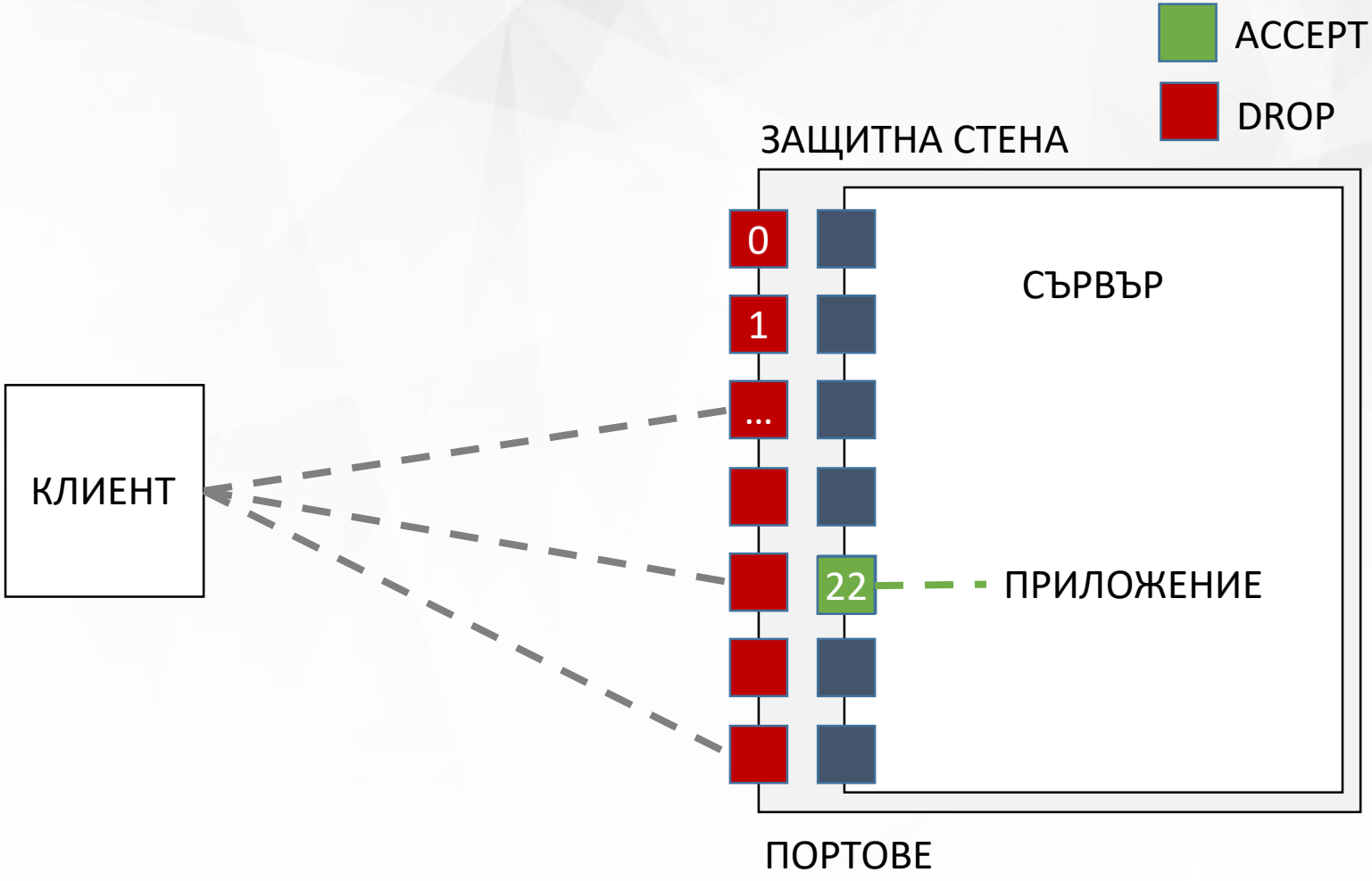
# Защо Port Knocking?

- Важни заключения от Deep-dive: MikroTik exploits - a security analysis (Tomas Kirnak) - MUM в САЩ, април 2019:
  1. „Подсигурете портовете за администриране на вашия рутер (не ги оставяйте отворени в публичната мрежа)“
  2. „Подсигурете подходяща защитна стена“

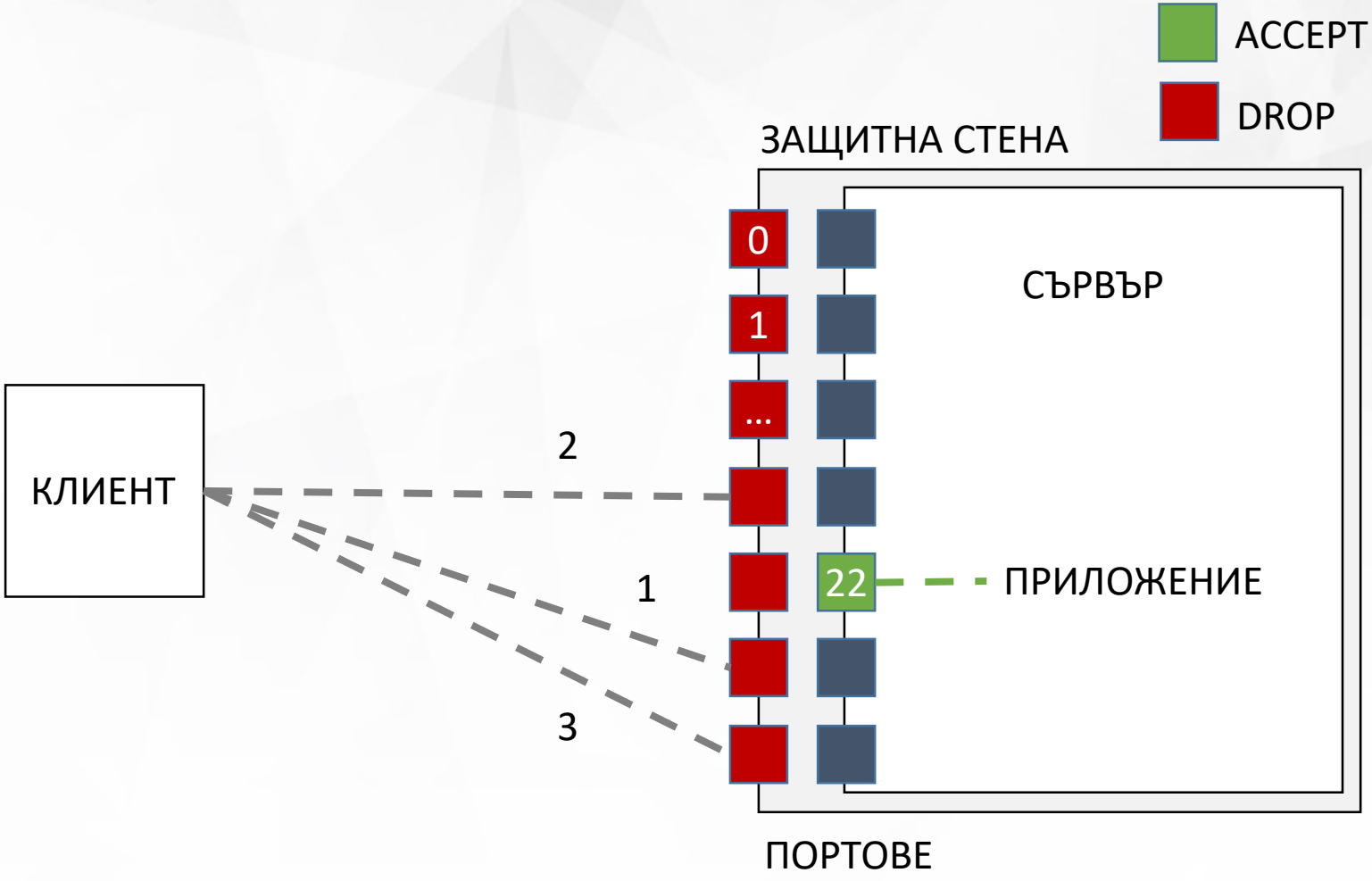
# Port Knocking

- „Метод за предаване на информация към затворени портове на мрежови устройства“, имащ за цел удостоверяване, което да осигури достъп до защитени услуги
- Различни варианти за изпълнение:
  - Строга последователност от пакети
  - Съдържание на пакета (OTP)

# В лесни стъпки (1)



# В лесни стъпки (2)



# В лесни стъпки (3)

# Предимства

- Удостоверяване чрез защитната стена
- Портовете са затворени при сканиране
- Динамична, гъвката и прозрачна защита
- 3 порта: ~141 трилиона комбинации
- Опростен дизайн, лесен за анализ

# Недостатъци

- Почуквания, които не пристигат в същата последователност (Out-of-order delivery)
- Без криптиране не предотвратява защита от spoofing и man-in-the-middle атаки
- Неприложим при публични услуги

# RouterOS и Port Knocking

MikroTik RouterOS, като сървър и клиент

# RouterOS и Port Knocking

- RouterOS може да играе ролята на **сървър**, като проследява (и действа):
  - последователността на опитите за връзка
  - данните, които могат да се пренасят (payload)
- Може да изпълнява и ролята на **клиент**, като прави предварително дефинирани опити за връзка и пренася информация с определени инструменти

# Port Knocking

## сървър

Последователност от връзки и/или пренасяне на информация в протоколите

# Port Knocking сървър

- Проследяване на строга последователност от опити за връзка към портове (TCP и/или UDP)
- Пренасяне и откриване на допълнителна информация чрез протоколи като ICMP (Ping Knocking), TCP, UDP, HTTP, DNS и др.
- Да ги използваме в комбинация

# Елементи в RouterOS

- /ip firewall **filter**:
  - **В условия (Matcher):** *chain; protocol; dst-port; connection-state; src-address-list; layer7-protocol; packet-size и др.*
  - **В действие (Action)**: add-src-to-address-list, accept, drop
- /ip firewall **connection**
- /ip firewall **address-list**
- /ip firewall **layer7-protocol**
- /ip firewall **nat;** /ip firewall **raw**

# TCP/UDP Port Knocking

- Създаваме 3 правила (**/ip firewall filter**):
  - Всяко от тях следи даден порт, като при опит за връзка към него, записва временно IP адреса на подателя в определен (според стъпката) списък.
  - Всяко следващо правило включва информация от предишното
- Добавяме правило, което да разрешава достъпа до услугата, само за IP адресите, попаднали в третия списък
- Какво още е необходимо?

# (ICMP) Ping Knocking

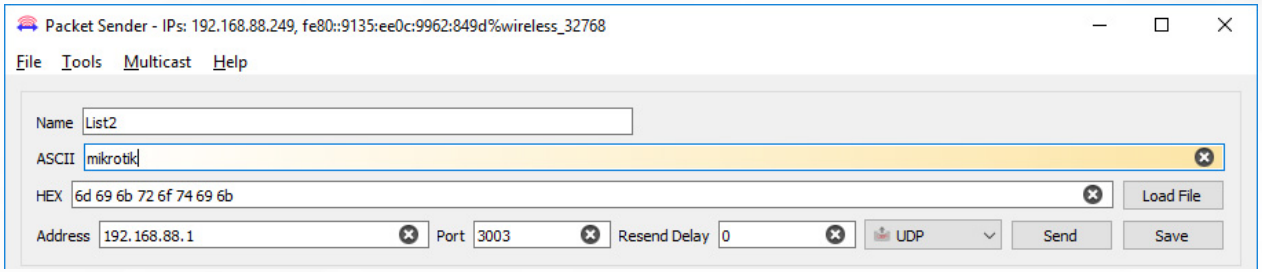
- ping 192.168.88.1 **-l 1000** -n 1
- /ip firewall **filter**:
  - **В условия (Matcher):** icmp; **packet-size**
- При ping с payload X байта, пакетът е с размер X+28 байта (X + 20 байта за IP хедър + 8 байта за ICMP хедър)
- Да видим как изглежда на практика?

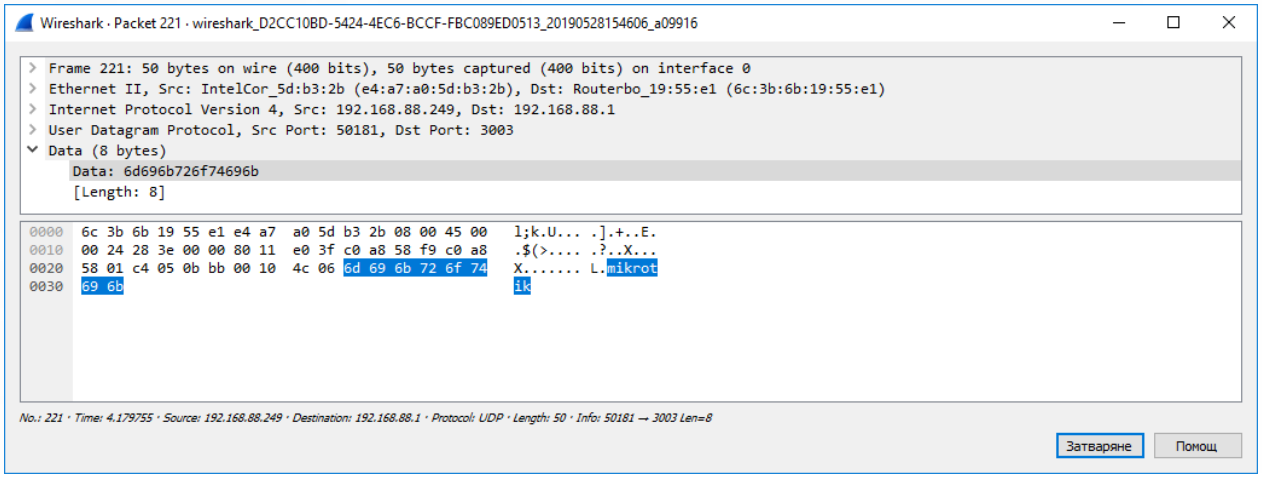
# Port Knocking с данни

- Можем да пренасяме данни в ICMP хедъра:
  - **ping 192.168.88.1 -p 6d316b723074316b**
  - 6d316b723074316b е **m1kr0t1k** (Hex)
- И да ги откриваме:
  - /ip firewall layer7-protocol \
    add name="ICMP payload" regexp=m1kr0t1k
  - /ip firewall filter add action=accept chain=input \
    layer7-protocol="ICMP payload" protocol=icmp
- Да видим как изглежда на практика?

# Port Knocking с данни

- Същото може да бъде постигнато и с транспортните протоколи TCP и UDP



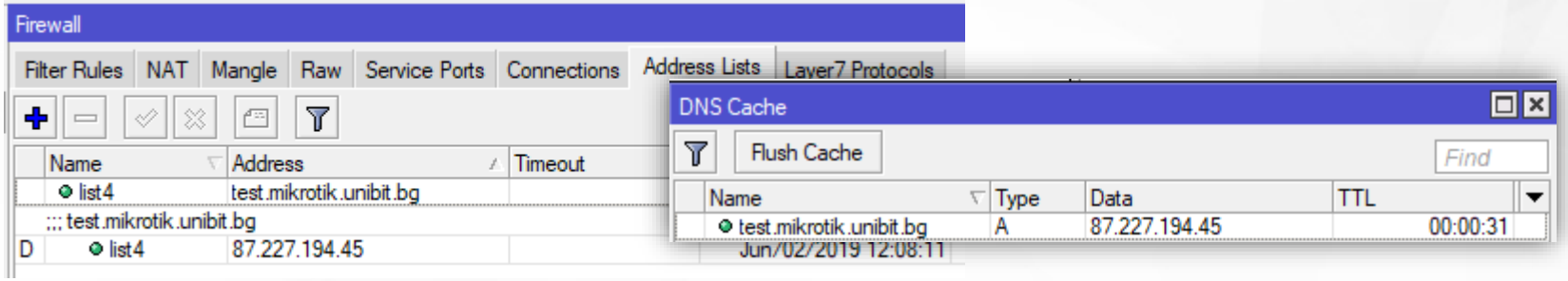



# HTTP URL Knocking

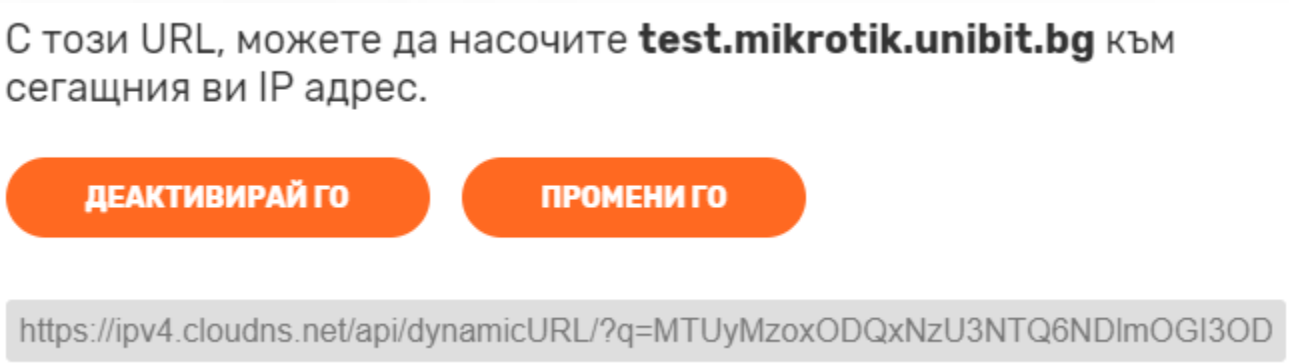
- Може да се извършва чрез HTTP GET заявка
  - http://192.168.88.1/m1kr0t1k
- Проверката става например чрез:
  - /ip firewall filter add action=add-src-to-address-list \
    address-list=list3 address-list-timeout=1m \
    chain=input **content=**m1kr0t1k \
    dst-address=192.168.88.1 dst-port=80 protocol=tcp
- Да видим как изглежда на практика?

# DDNS Knocking

- Имена на хостове в защитната стена




- Услугата [динамичен DNS]



# Port Knocking

## клиент

CLI, GUI, RouterOS

# Port Knocking клиент (CLI)

- **Тестване:** telnet, ping, curl
- Windows:
  - PortQry -n 192.168.88.1 -o 1001,3003,2002 -p both
  - knock 1.1.1.1 1001:tcp 3003:udp 2002:tcp
  - It's me (IM)
- Linux:
  - netcat -u 192.168.88.1 1001
  - hping3 -s  192.168.88.1 -c 1 -p 3003
  - knock 192.168.88.1 1001:tcp 3003:udp 2002:tcp
  - socat - TCP:192.168.88.1:1001

# Port Knocking клиент (GUI)

- **Тестване**: Уеб браузър, Winbox, Port Scanner
- Windows:
  - Windows Port Knock Application
  - KnockKnock - Port Knocking for Windows
  - PortQryUI
  - Packet Sender
- Android:
  - Port Knocker
  - Knock on Ports
  - Knock the Port

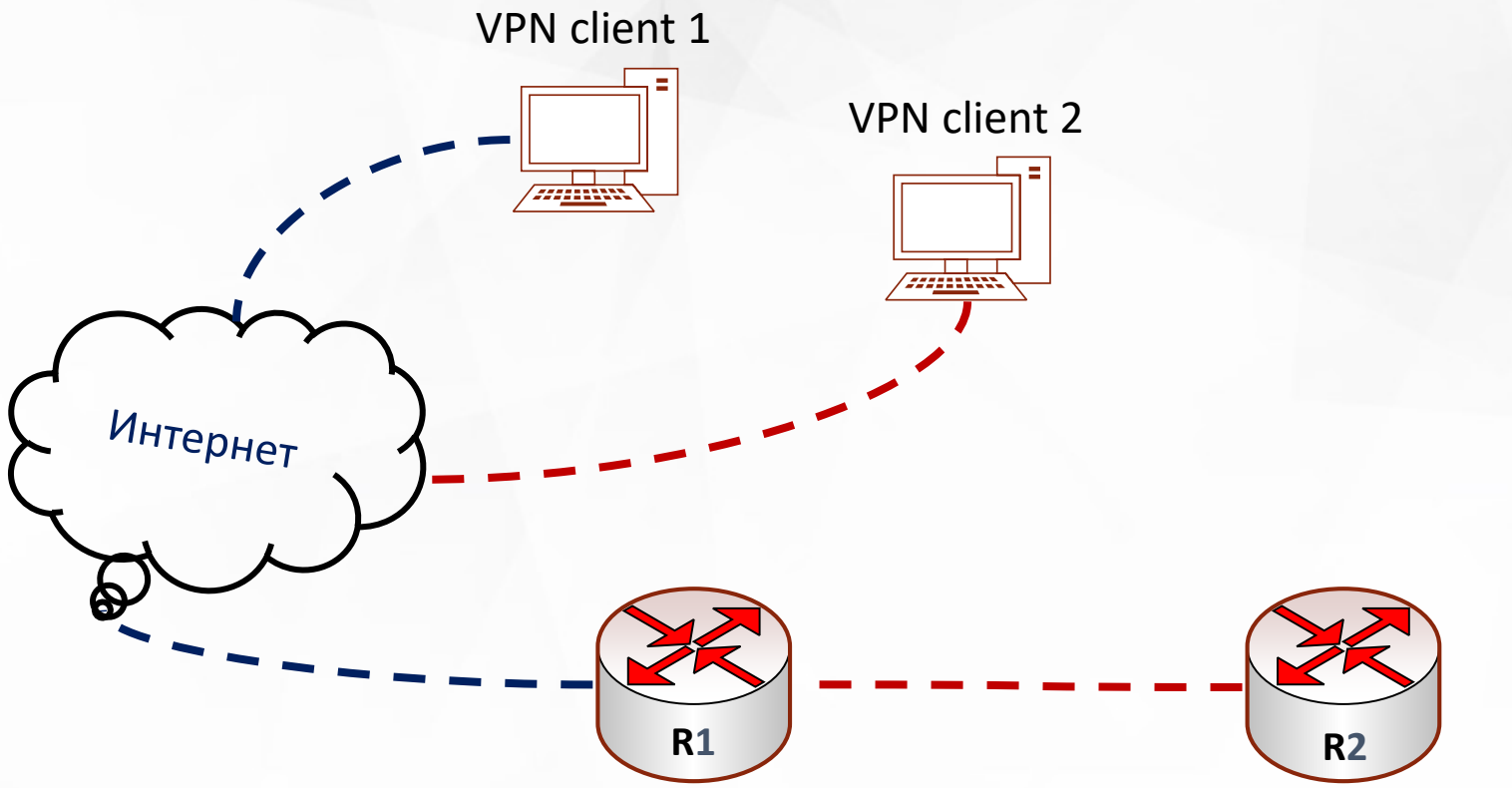
# Port Knocking клиент (RoS)

- /system **telnet** 192.168.88.1 1001
- /**ping** (ICMP)
  - /ping 192.168.88.1 size=1001 count=1 interval=2
- /tools **fetch**
  - /tool fetch address=192.168.88.1 mode=http port=1001 **src-path=*m1kr0t1k***
- /tools **traffic-generator**
  - Каквито пакети (RAW) са ви необходими ☺

# Port Knocking при препращане на портове

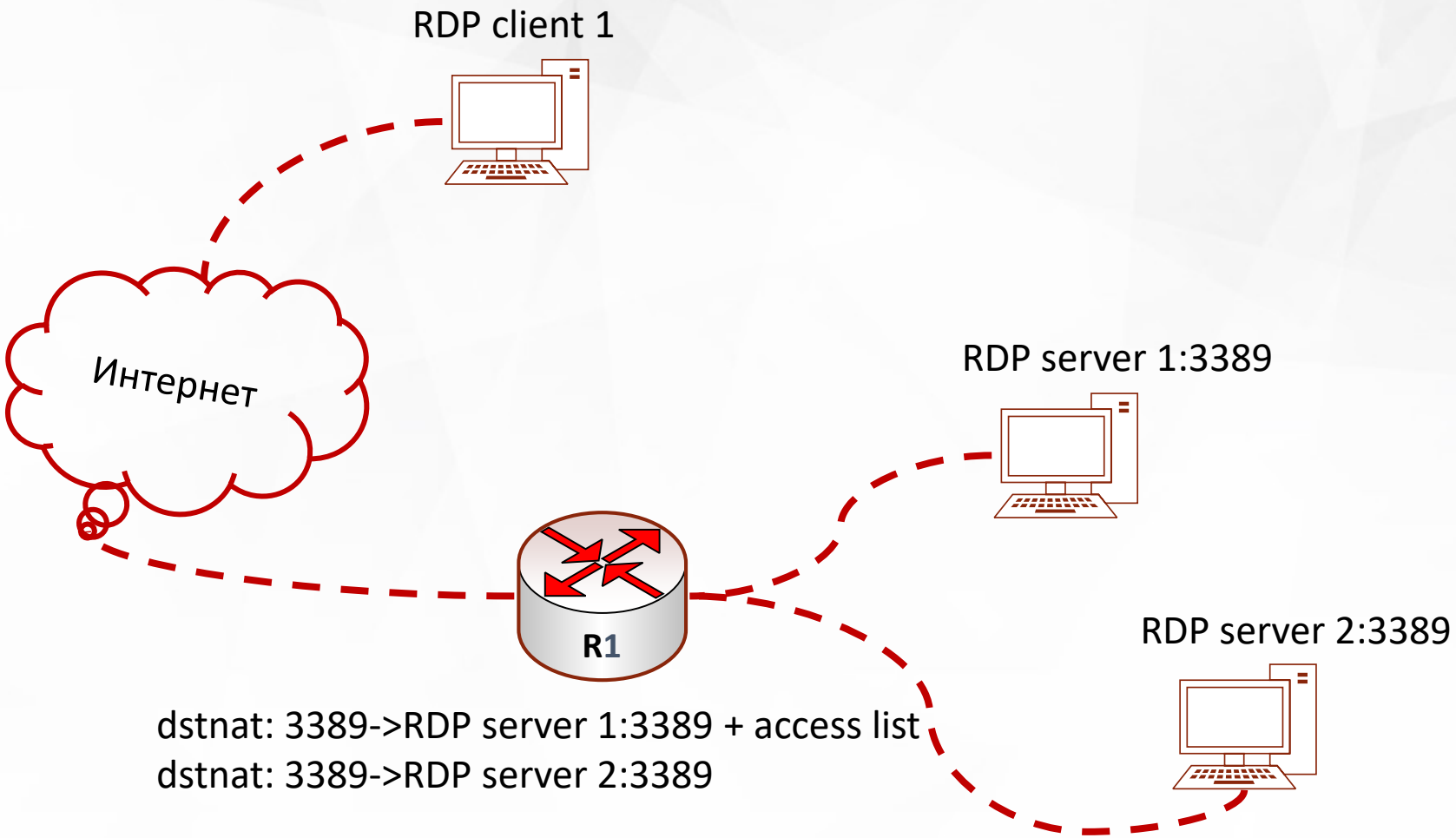
Използване на един и същи мрежови цокъл (сокет) за различни услуги

# Port Forwarding (1)

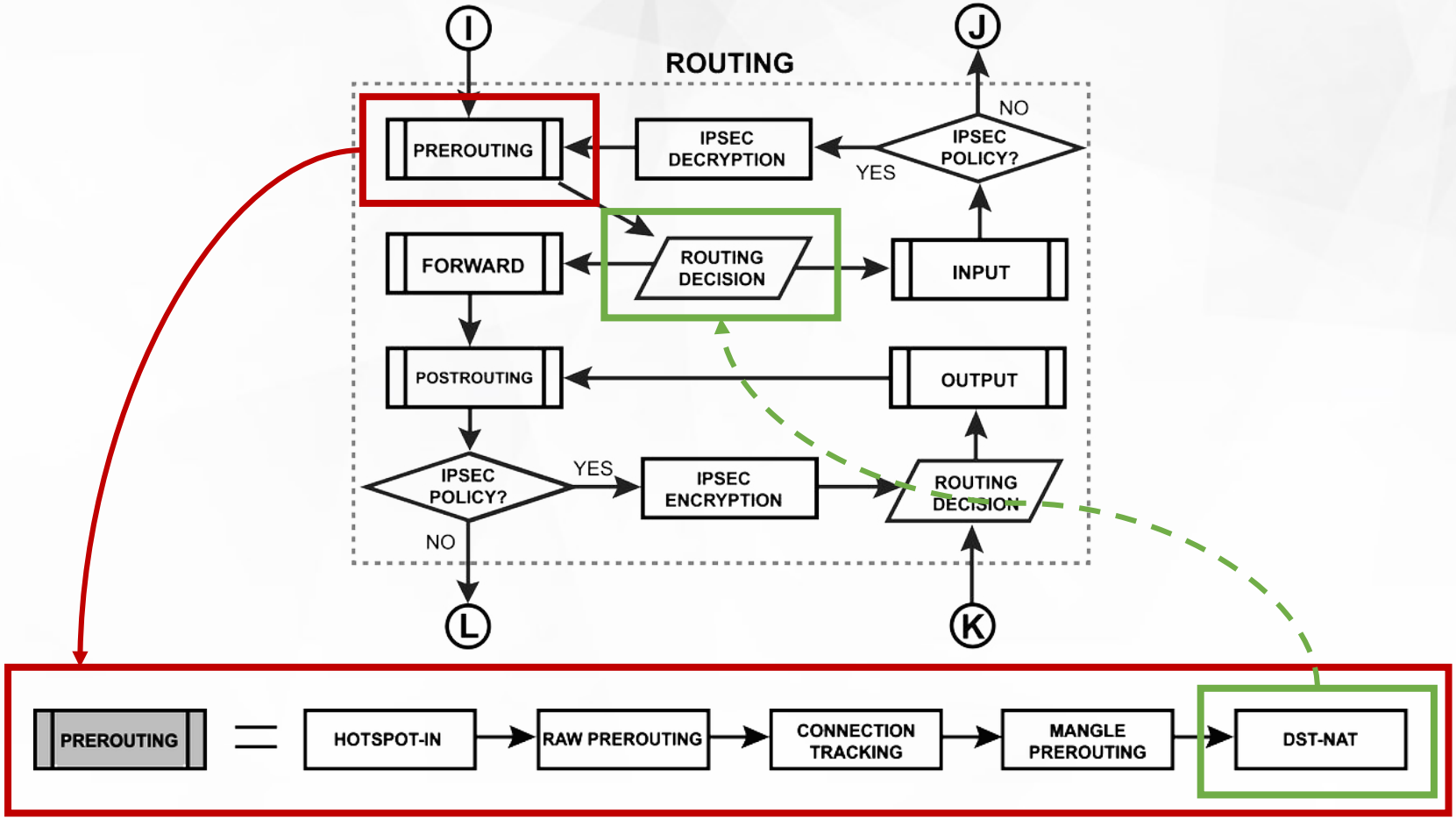



dstnat: 443->VPN server 2:443 + access list

# Port Forwarding (2)



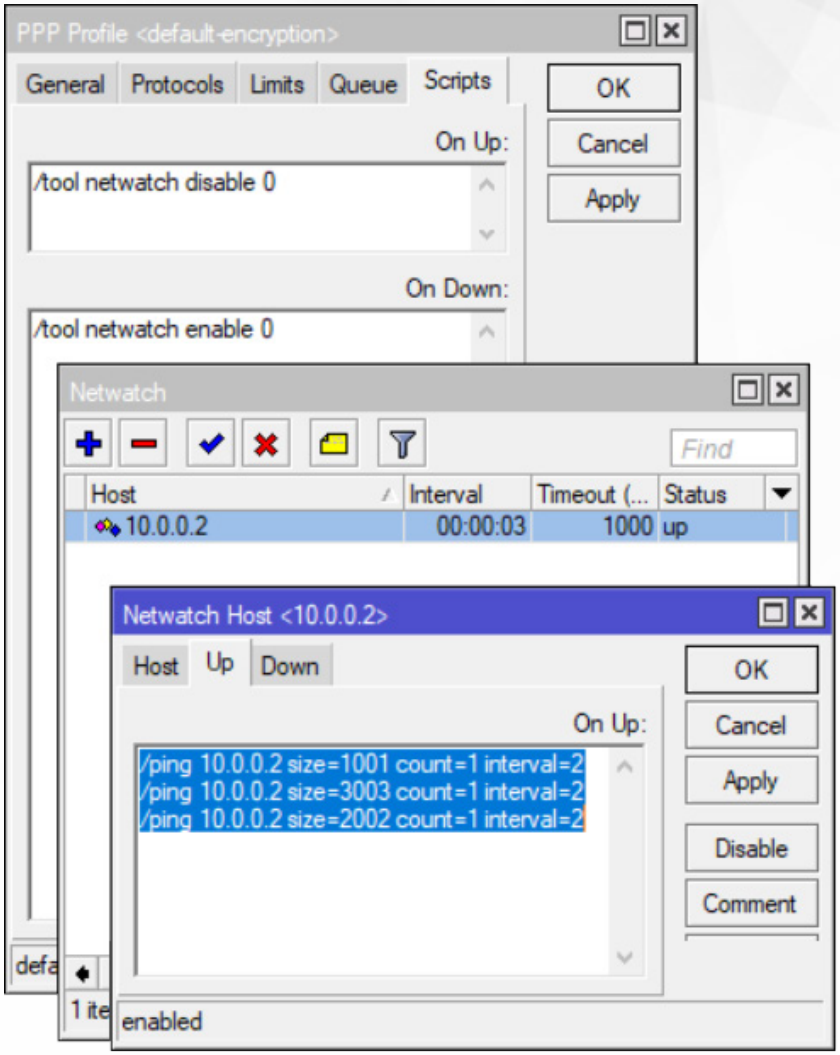
# Packet Flow | Routing

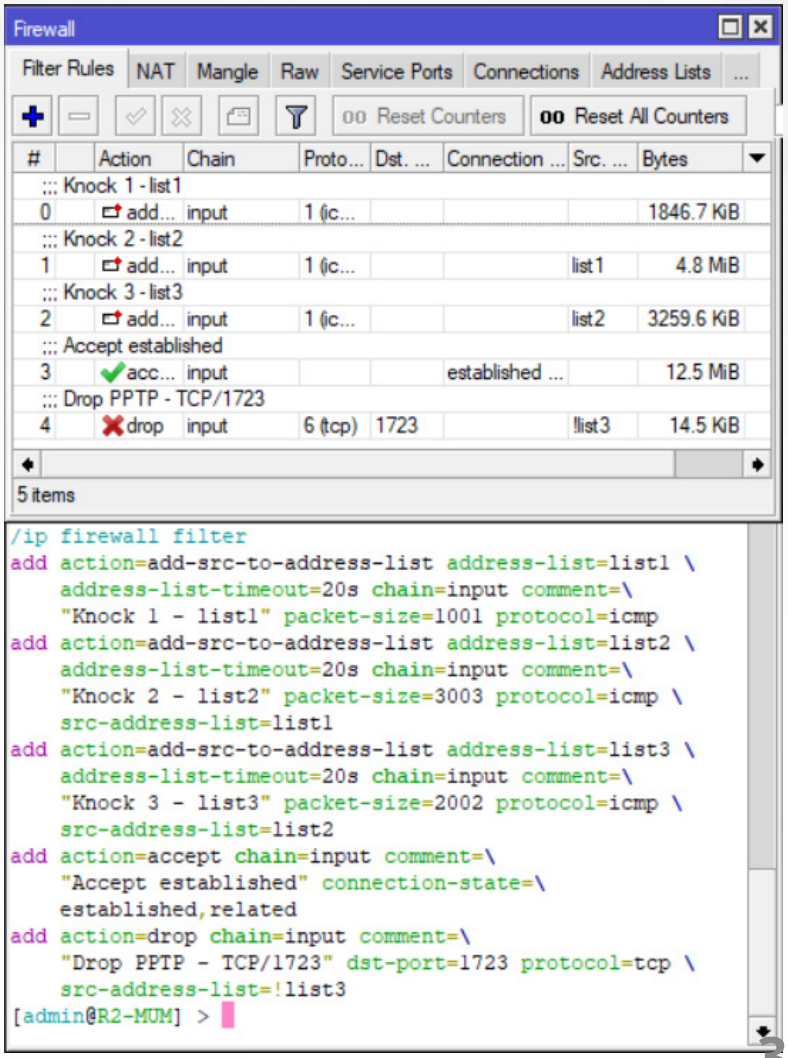


# VPN клиент / VPN сървър Port Knocking

Port Knocking RouterOS -> RouterOS

# VPN клиент / VPN сървър



# Добри практики

При изпълнение на Port Knocking

# Добри практики

1. **Приемете** връзки от тип **established** (и related)
2. Създайте ваша **уникална комбинация** (TCP/UDP/ICMP, данни)
3. Използвайте **custom chains** (ICMP, L7)
4. Още по-сигурно (**Address List Knocking**)
5. **dstnat** (NAT) или **forward** (Filter)

# Добри практики

6. /ip firewall **raw** – с някои ограничения
7. Комбинирайте със **защита от сканиране на портове**
8. Автоматизирайте
9. Не започвайте с **ICMP (ping)**
10. Не използвайте като **единствен слой на защита**

# Въпроси
# и отговори

Port Knocking с RouterOS. Допълнително ниво на защита за вашата мрежа

# Благодаря!

д-р Добри Бояджиев
MUM, София, 4 април 2019 г.
dobria@gmail.com, +359 884 923 752