

SMS-knocking jako zabezpieczenie zdalnego dostępu VPN.

Grzegorz Pomykała

O mnie




W branży IT od ponad 20 lat.
Na co dzień administruję infrastrukturą serwerową
(Windows, FreeBSD, wirtualizacja)
i sieciową (sieci lokalne, WiFi, tunele VPN).
Z urządzeń MikroTik korzystam od 12 lat.
Posiadam certyfikaty MTCNA,
MTCRE, MTCSE, MTCSWE.

kontakt: grzegorz@dito-it.pl



Rozwiązanie 'SMS-knocking' powstało jako odpowiedź na potrzeby klienta, który ze względów bezpieczeństwa nie chciał, aby jakiegokolwiek usługi sieciowe, w tym zdalny dostęp VPN, były 'widoczne' z zewnątrz. W obecnej chwili atakujący nie muszą nawet samodzielnie skanować w poszukiwaniu otwartych portów na urządzeniach sieciowych, zdobyć takie informacje można bez problemu, chociażby korzystając z serwisu 'shodan.io' (<https://www.shodan.io/>). Poniżej zrzut ekranu pokazujący ogólnodostępne informacje dla wskazanego adresu IP:

 **General Information**

Hostnames

Domains

CITYSTRADA.PL

Country

Poland

City

Warsaw

Organization


citystrada.pl

ISP

Citystrada.pl Sp z o.o.


ASN

AS201434

 **Open Ports**

443

1723

// 443 / TCP 

1489525118 | 2021-09-25T07:37:18.589449

Microsoft HTTPAPI httpd 2.0

HTTP/1.1 404 Not Found
Content-Type: text/html; charset=us-ascii
Server: Microsoft-HTTPAPI/2.0
Date: Sat, 25 Sep 2021 05:37:11 GMT
Connection: close
Content-Length: 315

SSL Certificate

Certificate:
Data:
Version: 3 (0x2)
Serial Number:
2c:1c:97:f4:ff:ca:ec:bc:49:ad:5a:78:64:37:26:71
Signature Algorithm: sha256WithRSAEncryption
Issuer: CN=
Validity
Not Before: Sep 13 04:37:46 2021 GMT
Not After : Sep 13 04:57:46 2022 GMT
Subject: CN=
Subject Public Key Info:
Public Key Algorithm: rsaEncryption
RSA Public-Key: (2048 bit)
Modulus:
00:cl:fb:99:79:18:59:b3:8f:1b:e1:cc:1a:57:f4:
56:53:5b:72:ae:bf:8a:07:5c:87:95:be:bb:74:cf:
6e:9f:54:d6:f2:3a:62:0c:97:a2:ee:20:8f:5c:7b:
9b:4c:4a:10:7a:91:e6:cd:52:04:17:6b:f2:14:4b:
db:8d:bf:0f:4a:d3:dd:a1:98:3a:9a:0f:a1:b7:69:

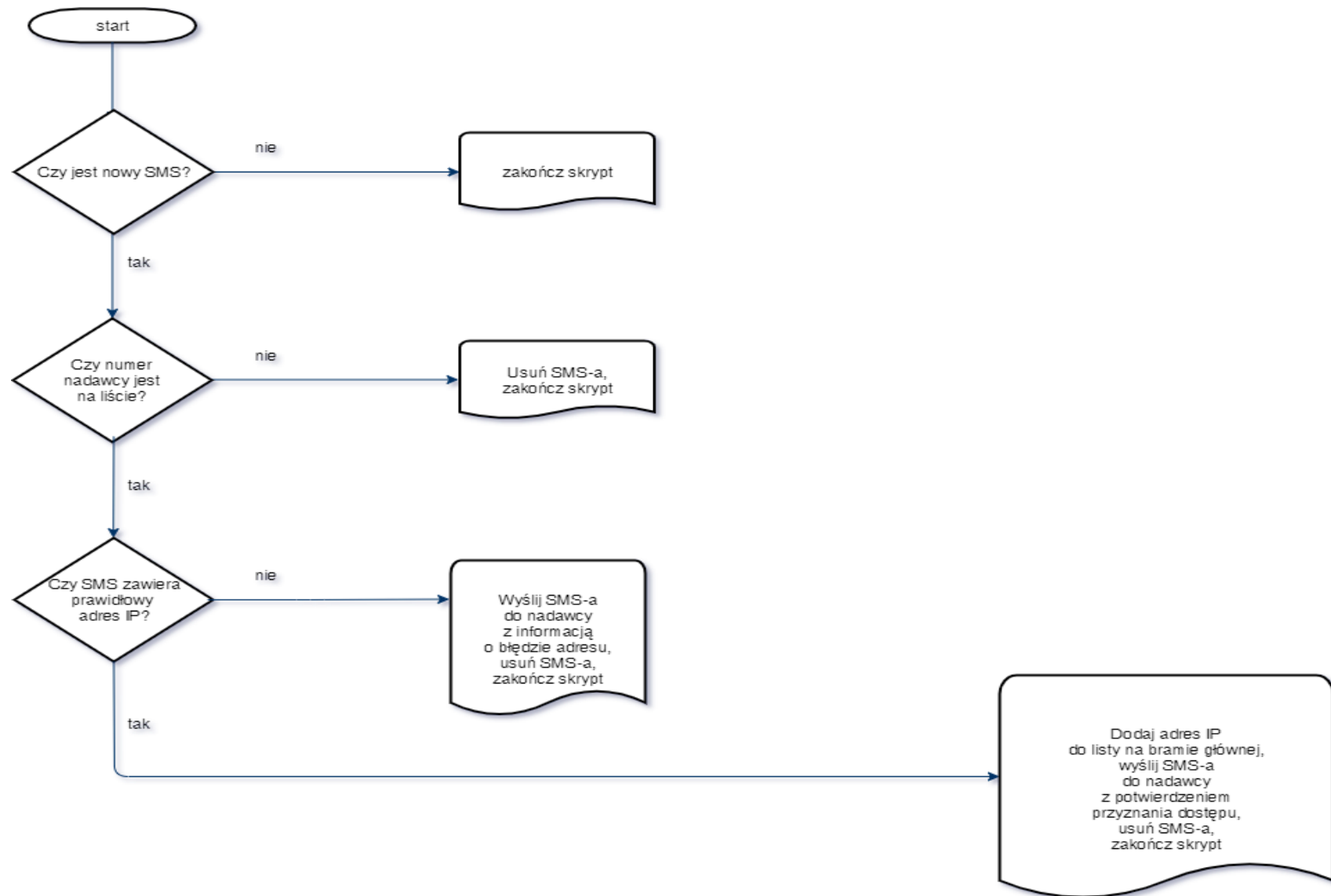
Założenie rozwiązania było takie, że wdzwaniane połączenia VPN do bramy są możliwe tylko z adresów IP znajdujących się na liście adresowej w firewall-u 'Allowed_VPN_IPs', wyzwaniem było tworzenie tej listy, szczególnie w kontekście dynamicznych adresów IP u personelu pracującego zdalnie z domu.

Z pomocą przyszedł RouterOS (skrypty, zwłaszcza polecenie 'ssh-exec') oraz wAP LTE który pracował u klienta jak zapasowe łącze do Internetu, ale pozwala on również na odbieranie i wysyłanie wiadomości SMS. Oczywiście, brama u klienta to również urządzenie MikroTik, RB1100AHx4.

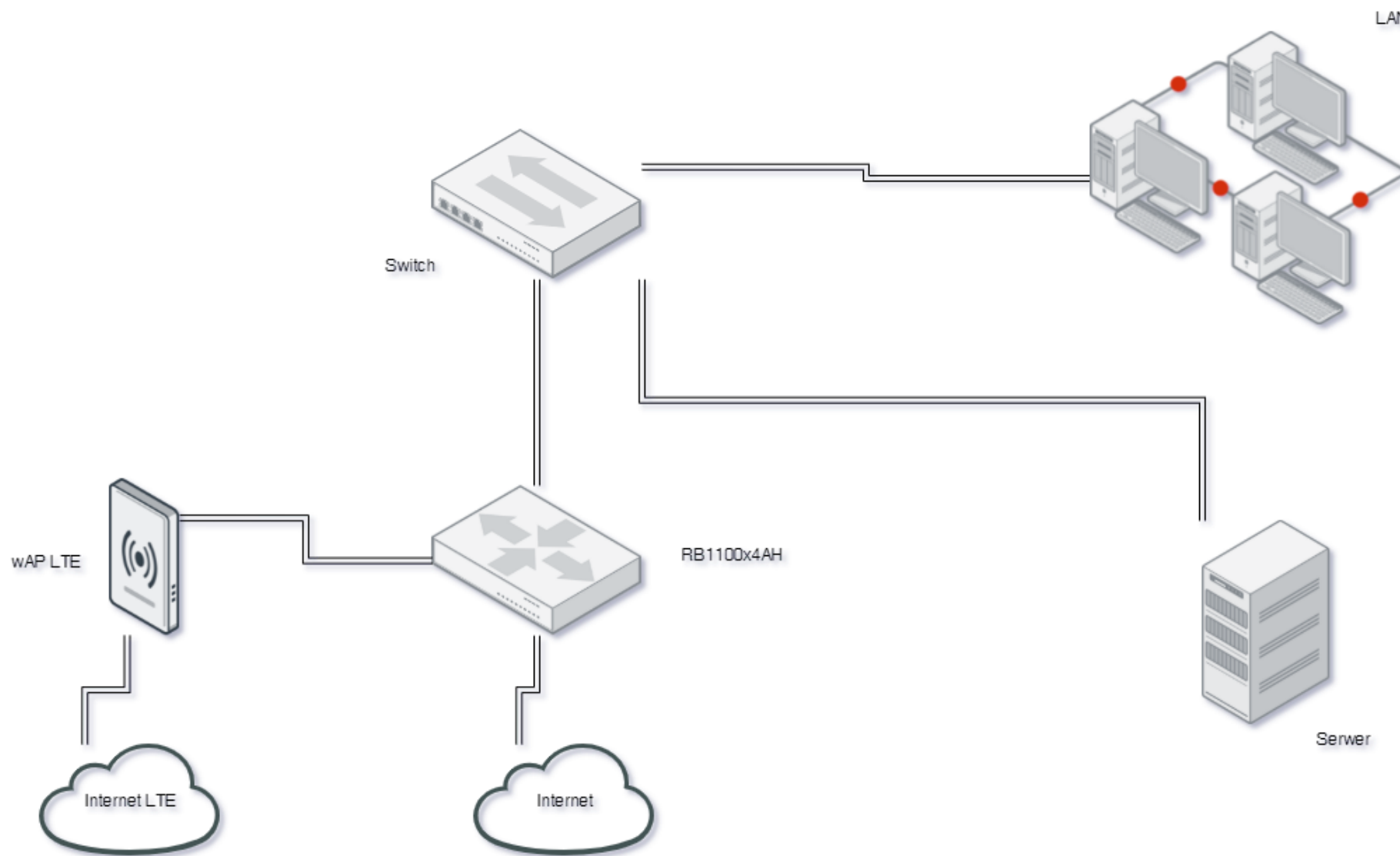
Rozwiązanie zostało stworzone w oparciu o następujące założenia:

- jest jedna centralnie zarządzana lista numerów telefonów personelu z których SMS-y są analizowane, pozostałe są niezwłocznie usuwane
- personel pracujący zdalnie jest zobowiązany do ustalenia swojego publicznego IP, korzystając np. z serwisu IPIFY (<https://www.ipify.org/>) oraz do wysłania go w wiadomości SMS do urządzenia wAP LTE
- odebrana wiadomość SMS jest sprawdzana czy zawiera prawidłowy adres IP, jeśli tak, to jest on dopisywany do listy 'Allowed_VPN_IPs' (z 'timeout=8h') na bramie która realizuje dostęp VPN, wysyłany jest też zwrotny SMS z informacją, że dostęp został przydzielony
- jeśli SMS nie zawiera prawidłowego adresu IP, odsyłana jest do nadawcy wiadomość SMS z informacją o tym fakcie
- wszystkie operacje są zapisywane w logu

Algorytm działania skryptu:



Od strony sprzętowej, rozwiązanie jest bardzo proste, wymaga dwóch urządzeń MikroTik, w tym jednego z modemem LTE, często już nimi dysponujemy, gdy funkcjonuje u klienta zapasowe łącze do Internetu oparte o LTE.



Od strony programowej, rozwiązanie oparte jest na skryptach RouterOS, sterowanie bramą główną odbywa się z wykorzystaniem polecenia 'ssh-exec'. Polecenie to nie jest interaktywne, dlatego też umożliwia skorzystanie z niego w skryptach i schedulerze, logowanie użytkownika odbywa się z wykorzystaniem klucza prywatnego/publicznego.

Opis jego zastosowania jest dostępny w dokumentacji MikroTik:

https://wiki.mikrotik.com/wiki/Manual:System/SSH_client#SSH-exec

[https://wiki.mikrotik.com/wiki/Use_SSH_to_execute_commands_\(public/private_key_login\)](https://wiki.mikrotik.com/wiki/Use_SSH_to_execute_commands_(public/private_key_login))

Wszystkie skrypty są wykonywane na urządzeniu z modemem LTE, dzięki użyciu polecenia 'ssh-exec' na bramie głównej dodawane są adresy IP do listy 'Allowed_VPN_IPs'.

Przygotowania rozpoczynamy od wygenerowania pary kluczy na potrzeby SSH i zaimportowania ich do urządzeń, w środowisku Linux lub Windows wykonujemy poniższe polecenie:

```
ssh-keygen -t rsa
```

'passphrase' musi być pusta

Wygenerowane zostaną dwa pliki 'id_rsa' oraz 'id_rsa.pub', są to odpowiednio klucz prywatny i publiczny. Pliki te należy zapisać w urządzeniach MikroTik.

Importujemy klucze do routerów. Publiczny i prywatny do lokalnego (który steruje innymi, wAP LTE), publiczny do zdalnych (które są sterowane, RB1100AHx4).

Na zdalnym routerze potrzebny jest zadedykowany użytkownik z mocnym hasłem (uprawnienia min.: ssh, read, write), np. o nazwie 'remotessh'

BARDZO WAŻNE!!! - na lokalnym routerze importujemy klucze dla użytkownika, który będzie się logował do zdalnego / wywoływał skrypty ze schedulera

Lokalny:

```
/user ssh-keys private import user=admin private-key-file=id_rsa public-key-file=id_rsa.pub passphrase=""
```

Zdalny:

```
/user ssh-keys import user=remotessh public-key-file=id_rsa.pub
```

Ustawienia 'IP/SSH' na obu urządzeniach:

```
/ip ssh set allow-none-crypto=no always-allow-password-login=no forwarding-enabled=both host-key-size=2048 strong-crypto=yes
```


Zdalne wykonywanie komend na routerze z użyciem 'ssh-exec' (przykłady):

```
/system ssh-exec address=192.168.111.240 user=remotessh command="/ip firewall  
address-list add address=1.2.3.4 timeout=8h list=Allowed_VPN_IPs comment=zdalnie"
```

```
/system ssh-exec address=192.168.111.240 user=remotessh command="/ip firewall  
address-list remove [find address=1.2.3.4 list=Allowed_VPN_IPs]"
```

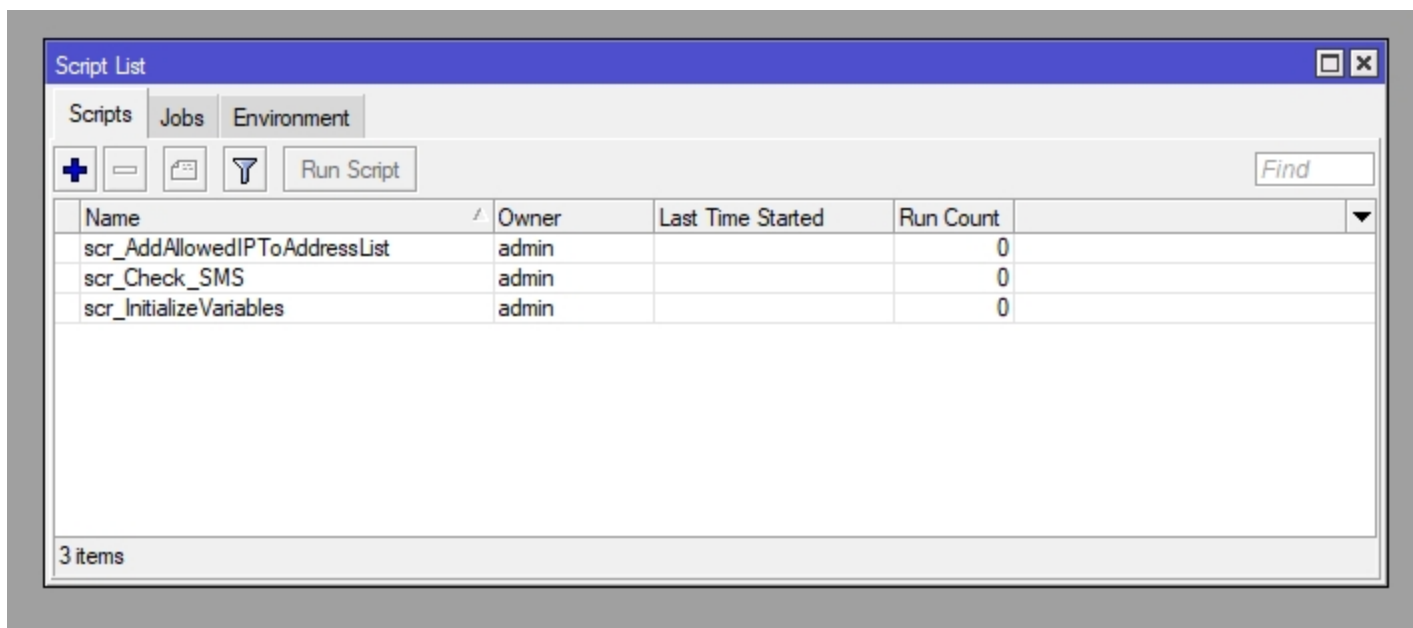
Kolejnym krokiem jest przygotowanie na bramie głównej reguł firewall-a, które dają dostęp do usług VPN tylko dla adresów z listy 'Allowed_VPN_IPs', poniżej przykład dla SSTP:

```
/ip firewall filter add action=accept chain=input comment="allow SSTP" dst-  
port=443 protocol=tcp src-address-list=Allowed_VPN_IPs
```

Oczywiście należy stworzyć stosowne reguły dla każdego rodzaju usługi VPN z której korzystamy.

Można również wykorzystać ten mechanizm dla usług w sieci LAN, do których dostęp jest realizowany poprzez Destination-NAT.

Główna funkcjonalność rozwiązania opiera się na trzech skryptach.



Pierwszy 'scr_InitializeVariables' przechowuje listę dozwolonych numerów telefonów i ich użytkowników, jest wywoływany w schedulerze po każdym uruchomieniu urządzenia. Jeżeli wprowadzimy w nim zmiany dotyczące dozwolonych numerów, wystarczy go uruchomić ręcznie, nie trzeba restartować urządzenia.

```
# scr_InitializeVariables
# initialize array of allowed phone numbers/users
:delay 30s

:global gAllowedIP;
:global gAllowedPhoneNumber;
:global gAllowedUser;
:global gaAllowedPhoneNumbers [:toarray ""];
:set ($gaAllowedPhoneNumbers ->" +48123456789") "Eustachy Pipsztycki";
:set ($gaAllowedPhoneNumbers ->" +48987654321") "Klemens Psikuta";

:delay 60s
/interface lte set disabled=no [find];
```

Drugi skrypt 'scr_Check_SMS', jest wywoływany w schedulerze, z interwałem 30 sekund. Sprawdza on czy numer z którego został wysłany jest na liście numerów dozwolonych, weryfikowana jest poprawność przysłanego adresu IP, wywołuje skrypt dopisujący adres IP do listy na bramie głównej, a na koniec usuwa SMS-a ze skrzynki odbiorczej modemu.

```

# scr_Check_SMS
:global gAllowedIP;
:global gAllowedPhoneNumber;
:global gAllowedUser;
:global gaAllowedPhoneNumbers;
:local Settings [ / tool sms get ];
:local UserIP;
:local VarType;

# check SMS in a loop
:while ([ / tool sms inbox print count-only ] > 0) do={
:local Phone [ / tool sms inbox get ([ find ]->0) phone ];
:local Message "";
:set $gAllowedUser {$gaAllowedPhoneNumbers->$Phone};

:foreach Sms in=[ / tool sms inbox find where phone=$Phone ] do={
    :local SmsVal [ / tool sms inbox get $Sms ];
    :if ($Phone = $Settings->"allowed-number" && \
        ($SmsVal->"message")~("^:cmd " . $Settings->"secret" . " script ")) do={
        :log debug "Removing SMS, which started a script.";
        / tool sms inbox remove $Sms;
    } else={
        :if ($gAllowedUser!="") do={
        :set $Message ($SmsVal->"message");
        :log info ("SMS from " . $gAllowedUser . " / phone number: " . $Phone . "
            -> " . $Message);
        }
    }
}

```

```

    :set $UserIP [:toip $Message];
    :set $VarType [:typeof $UserIP];
    :log info ($gAllowedUser . "'s IP -> " . $UserIP);
    :log info $VarType;
    :if ($VarType="ip") do={
    :set $gAllowedIP $UserIP
    :log info $gAllowedIP;
    :set $gAllowedPhoneNumber $Phone
    :log info $gAllowedPhoneNumber;
    /system script run scr_AddAllowedIPToAddressList;
    /tool sms inbox remove $Sms;
} else={
    :set $gAllowedIP $UserIP
    :set $gAllowedPhoneNumber $Phone
    /tool sms send phone-number=$gAllowedPhoneNumber message="Provided IP
        address is invalid." port=lte1;
    :set $gAllowedIP ""
    :set $gAllowedPhoneNumber ""
    /tool sms inbox remove $Sms;
}
} else={
    :log info ("SMS from unknown phone number: " . $Phone . " -> " . $Message);
    /tool sms inbox remove $Sms;
}
}
}
}

```

Trzeci skrypt 'scr_AddAllowedIPToAddressList' poprzez polecenie 'ssh-exec' dopisuje przysłany w SMS-ie adres IP do listy firewall-a na bramie głównej oraz wysła powiadomienie SMS-em o przyznaniu dostępu.

```
# scr_AddAllowedIPToAddressList
```

```
:global gAllowedIP;
```

```
:global gAllowedPhoneNumber;
```

```
:global gAllowedUser;
```

```
:global gaAllowedPhoneNumbers;
```

```
:local EntryComment;
```

```
:set $EntryComment "VPN access: $gAllowedPhoneNumber -> $gAllowedUser";
```

```
#:log info "Add to address list -> $gAllowedIP";
```

```
/system ssh-exec address=192.168.0.1 port=22 user=remotessh command="/ip firewall  
address-list remove [find comment=\"$EntryComment\"]"
```

```
:delay 1s
```

```
/system ssh-exec address=192.168.0.1 port=22 user=remotessh command="/ip firewall  
address-list add address=$gAllowedIP timeout=8h list=Allowed_VPN_IPs  
comment=\"$EntryComment\""
```

```
:delay 1s
```

```
/tool sms send phone-number=$gAllowedPhoneNumber message="Access granted for IP:  
$gAllowedIP" port=lte1;
```

Przedstawione rozwiązanie działa od ponad 2 lat, dla ok. 20 osób pracujących zdalnie.

Jak wspominałem wcześniej, można również wykorzystać ten mechanizm dla usług w sieci LAN, do których dostęp jest realizowany poprzez Destination-NAT.

Pytania?