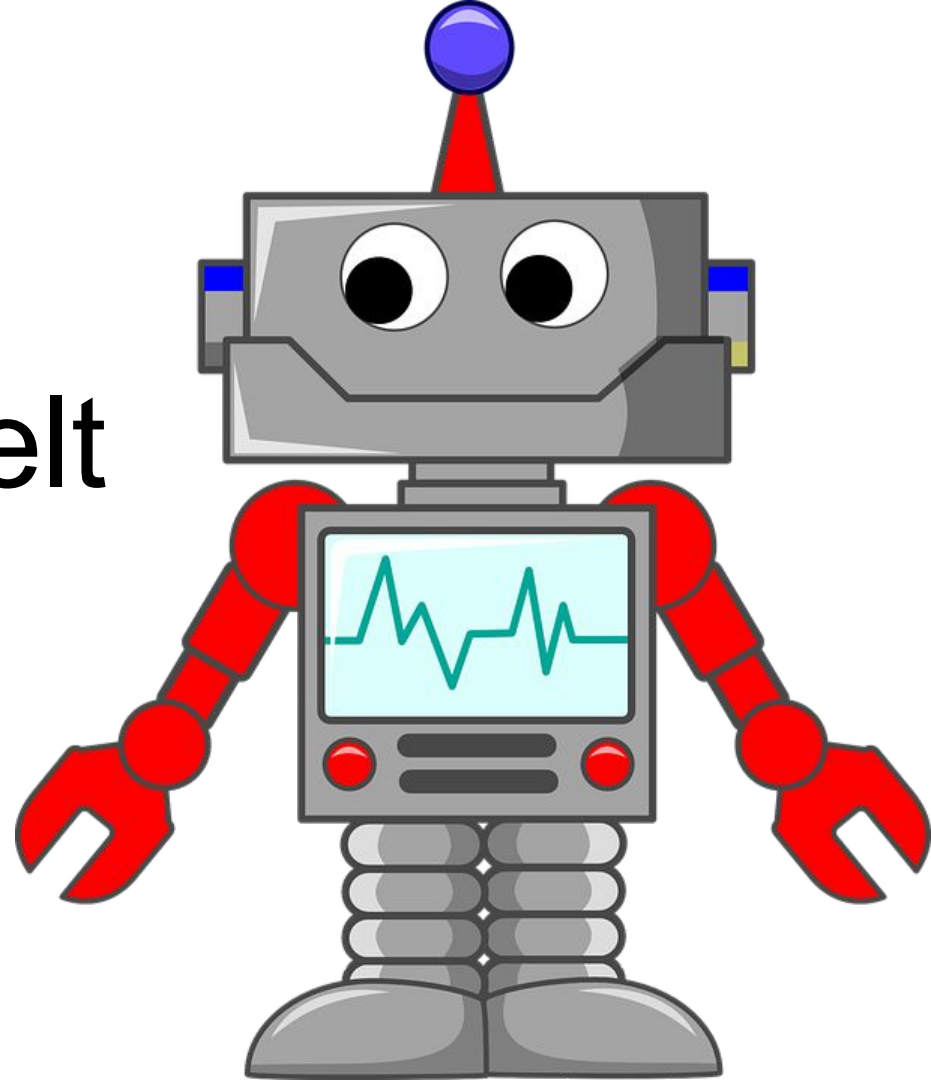
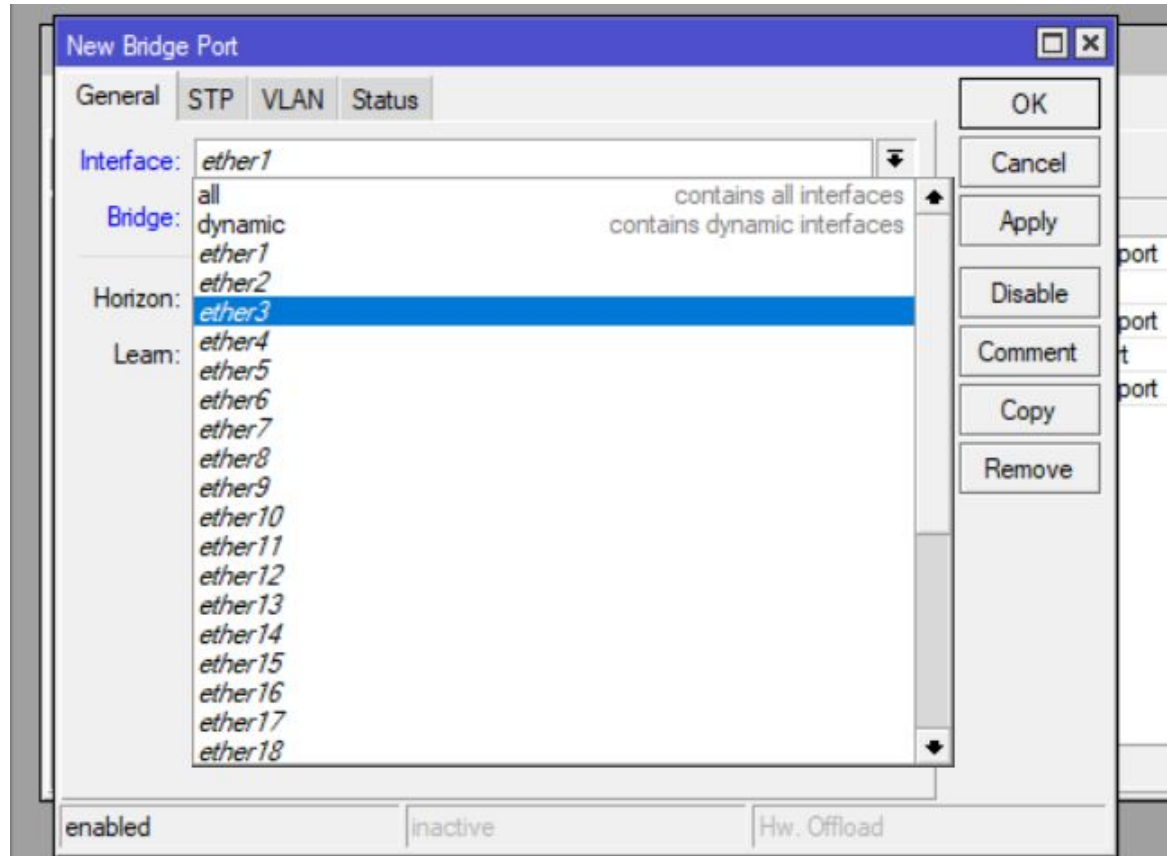


Have you ever felt  
like a robot?





Try adding 23 ports to a bridge...



Let's see...



Quick Set

CAPsMAN

Interfaces

Wireless

Bridge

PPP

Switch

Mesh

IP

Interface <wlan1>

General

Wireless

HT

WDS

Nstreme

NV2

Status

Traffic

Mode: ap bridge

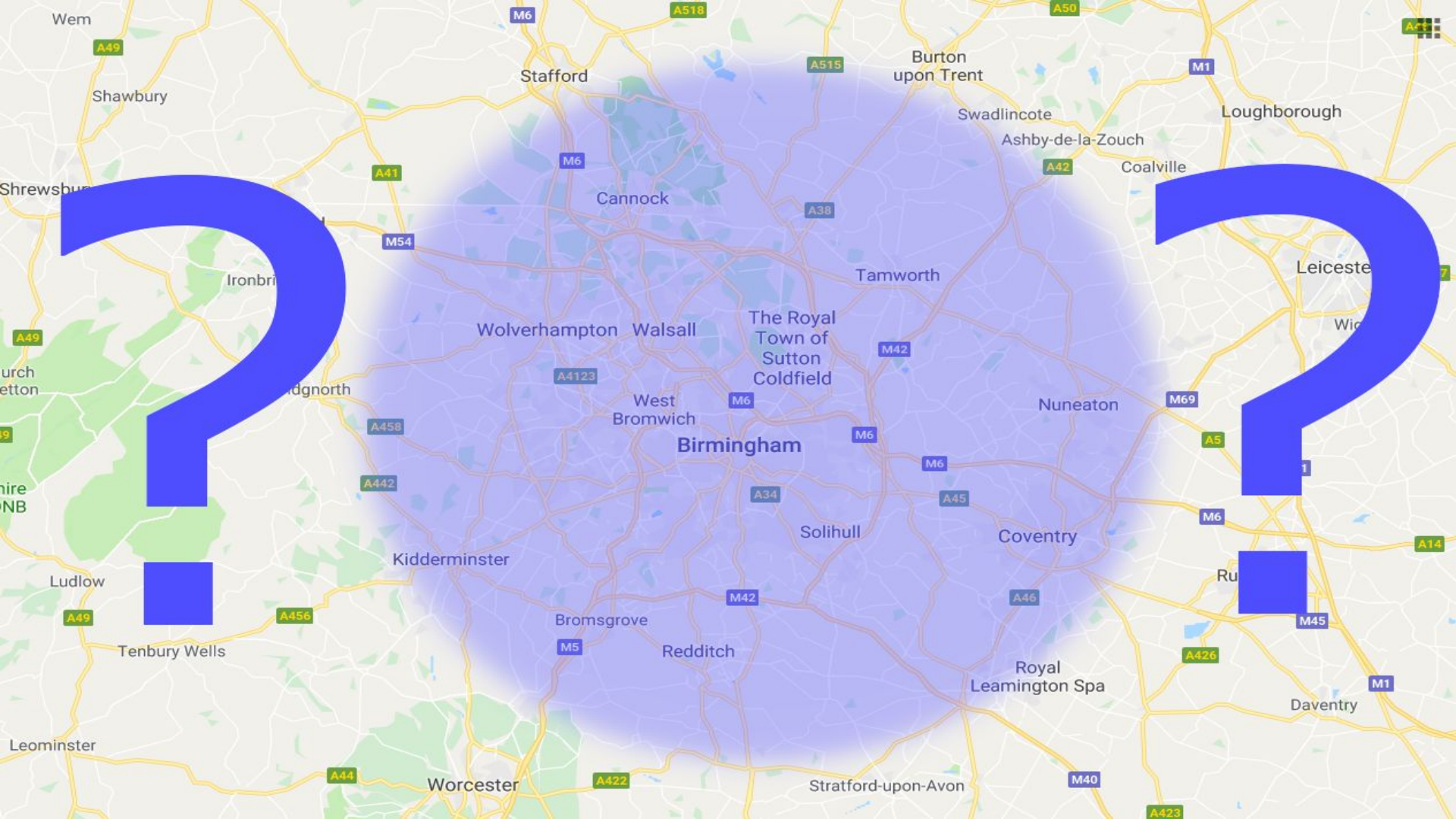
Band: 2GHz-B/G

Channel Width: 20MHz

Frequency: 2412

SSID: Cafe free wifi

Scan List: default



# Google Geolocation API

```
{  
  "homeMobileCountryCode": 310,  
  "homeMobileNetworkCode": 410,  
  "radioType": "gsm",  
  "carrier": "Vodafone",  
  "considerIp": "true",  
  "cellTowers": [  
    // See the Cell Tower Objects section below.  
  ],  
  "wifiAccessPoints": [  
    // See the WiFi Access Point Objects section below.  
  ]  
}
```



# Google Geolocation API - only WiFi

```
{
  "wifiAccessPoints": [
    {
      "macAddress": "00:25:9c:cf:1c:ac",
      "signalStrength": -43,
    },
    {
      "macAddress": "00:12:23:00:56:78",
      "signalStrength": -62,
    },
    . . . . .
  ]
}
```

## Scanner (Running)

Interface: *wlan1*

Start

Stop

Close

Connect

New Window

 Background Scan

	Address	SSID	Channel	Signal ... ▾	Noise...	Signal To Noise /	Radio Name	Route ▼
AP	40:0D:10:1E:67:21	VM7685819	2412/2...	-64	-114	50		▲
AP	52:0D:10:1E:67:21	Virgin Media	2412/2...	-64	-114	50		
AP	80:37:73:1F:B1:B0	VM890683-2G	2437/2...	-79	-112	33		
AP	48:D3:43:42:90:19	VM9403391	2462/2...	-81	-111	30		
AP	5A:D3:43:42:90:19	Virgin Media	2462/2...	-81	-111	30		
AP	D2:05:C2:15:11:B1	Virgin Media	2437/2...	-84	-112	28		
AP	00:8E:F2:CE:D1:5A	virginmedia1149311	2462/2...	-85	-111	26		
AP	C0:05:C2:15:11:B1	VM8313735	2437/2...	-86	-112	26		
AP	70:5A:0F:6F:A4:95	DIRECT-94-HP E...	2462/2...	-86	-111	25		
AP	EC:F4:51:98:9B:C4	BTHub6-ZF36	2412/2...	-86	-114	28		
AP	90:21:06:1C:B0:79	SKYC6D1B	2462/2...	-87	-111	24		

# Google Geolocation API - the result

```
{  
  "location": {  
    "lat": 51.0,  
    "lng": -0.1  
  },  
  "accuracy": 1200.4  
}
```

And finally we can do:

```
https://www.google.com/maps/?q=51.0,-0.1
```

# Geolocation - summary of the steps

1. Run the scan on the wireless interfaces
2. Prepare the JSON query with few empty MACs
3. Copy-paste the MAC addresses and signals
4. Run the API query, open the result
5. Copy-paste the coordinates to HTTP link

```

# Main geolocation script for geolocate.pyk project.
# https://github.com/0x00sec/geolocate.pyk
# Copyright: Daniel Starmach 2018
# Shared under the MIT License

# Put your Google Geolocation API key here:
local apikey "XXXXXXXXXX";

splitfilenames - Function creating array of non-empty lines from the file
# Example: local filenamesarray [splitfilenames "abcd\example.txt"];
local splitfilenames do{
  local file [File get "$1" contents];
  local filenames [toarray ""];
  local filename 0;
  local fileposition 0;
  local eofline 0;
  while {eofline} do{
    local newline [find $file "\n" $fileposition];
    if {typeof $newline}="nil" do{
      set filename $1;
    } else{
      if {($newline==$fileposition)} do{
        set filenames [$filenames,$apikey $fileposition $newline];
      }
      set fileposition [$newline+1];
    }
  }
}
return filenames;

# prepareJSON - function creating the JSON request data for Google Geolocation API from the array of wireless scan lines
# Example: local http-data [prepareJSON $filenamesarray]
local prepareJSON do{
  local request ["{"+JSONAccessPoints["{"];
  local jsonLine 1;
  foreach line in $filenames do{
    local nl [find $line "\n"];
    local nl [find $line "\n" $nl];
    local nl [find $line "\n" $nl];
    local nl [find $line "\n" $nl];
    local newline [pick $line 0 $nl];
    local signal [pick $line [$nl+1] $nl];
    if {$signal} do{
      set request [$request - ","];
    } else{
      set jsonLine 0;
    }
    set request [$request - '{"macAddress":"' + $macAddr + "'," + signalStrength":"' + $signal + '"}'];
  }
  set request [$request - ""];
  return $request;
}

# getGeolocation - function getting latitude, longitude and Accuracy data from the Google Geolocation API response (array of lines)
# response is in form of array with keys: lat, lon, acc and valid (valid=0 - no position, valid=1 - position found)
# Example: if {($getGeolocation $filenamesarray)->"valid"}{local accuracy [$getGeolocation $filenamesarray]->"acc"}
local getGeolocation do{
  local valid 0;
  local lat 0;
  local lon 0;
  local acc 0;
  foreach line in $filenames do{
    if {typeof [find $line "\{lat\}"}="nil" do{
      set $valid $valid;
      set $lat [pick $line {[find $line "\{lat\}"}] [find $line "\,"] [find $line "\{lat\}"}];
    }
    if {typeof [find $line "\{lon\}"}="nil" do{
      set $lat [pick $line {[find $line "\{lon\}"}] [find $line "\,"] [find $line "\{lon\}"}];
    }
    if {typeof [find $line "\{accuracy\}"}="nil" do{
      set $lat [pick $line {[find $line "\{accuracy\}"}] [find $line "\,"] [find $line "\{accuracy\}"}];
    }
  }
  if {($valid)} do{
    return {valid:$lat;lon:$lon;acc:$acc};
  }
}

# Initialize table for AP list
local apList [toarray ""];

# Run wireless scan on all wireless interfaces and store results in separate array lines
foreach wifiInterface in [interface wireless find] do{
  local filename [geolocate.pyk - //interface wireless get $wifiInterface name] - ".scan";
  /interface wireless scan $wifiInterface duration:10s save-file:$filename;
}

# Wait for the files to be written
delay 2s;

# Fill the AP list from the geolocate.pyk scan files
foreach scanFile in [File find name-geolocate.pyk*] do{
  local filename [File get $scanFile name];
  set $apList [$apList,$splitfilenames $filename];
  /File remove $filename;
}

# Prepare and send the Google API JSON request
local httpData [prepareJSON $apList];
/Post http url="https://www.google.com/geolocation/v1/geolocate?key=$apikey" http-content-type="application/json" http-method=post http-data="$httpData" dst-path="locationfile.txt";
delay 2s;

# Parse the results
local locationFile [splitfilenames "locationfile.txt"];
local result [$getGeolocation $locationFile];
print "";
print "";
print "";
if {($result->"valid")} do{

```

# Geolocation - the script

```

# Initialize table for AP list
:local apList [:toarr ""];

# Run wireless scan on all wireless interfaces and store results in separate array lines
:foreach wifiInterface in=[/interface wireless find] do={
    :local fileName ("geoMikroTikScan-" . [/interface wireless get $wifiInterface name] . ".scan");
    /interface wireless scan $wifiInterface duration=10s save-file="$fileName";
}

# Wait for the files to be written
:delay 2s;

# Fill the AP list from the geoMikroTik.scan files
:foreach scanfile in=[/file find name~"geoMikroTikScan-"] do={
    :local fileName [/file get $scanfile name];
    :set $apList ($apList,[${splitFileLines $fileName}]);
    /file remove $fileName;
}

# Prepare and send the Google API JSON request
:local httpData [$prepareJSON $apList];
/tool fetch url="https://www.googleapis.com/geolocation/v1/geolocate?key=$apiKey" http-content-type="application/json" http-method=post http-data="$httpData" dst-path="locationFile.txt";
:delay 2s;

# Parse the results
:local locationFile [${splitFileLines "locationFile.txt"}];
:local result [${getGoogleLocation $locationFile}];
:put "";
:put "";
:put "";
:if (($result->"valid")>0) do={
    :local lat ($result->"lat");
    :local lon ($result->"lon");
    :local acc ($result->"acc");
    :put "Coordinates found:"
    :put "Latitude: $lat";
    :put "Longitude: $lon";
    :put "Accuracy: $acc m";
    :put "Direct Google Maps link:";
    :put "https://www.google.com/maps/\?q=$lat,$lon";
} else={
    :put "Unfortunately, couldn't get location.";
}

```

[admin@MikroTik] > sys script run geoMikroTik

Flags: A - active, P - privacy, R - routeros-network, N - nstreme, T - tdma, W - wds, B - bridge

	ADDRESS	SSID	CHANNEL	SIG	NF	SNR	RADIO-NAME
AP	40:0D:10:1E:67:21	VM7685819	2412/20/gn	-71	-106	35	
AP	52:0D:10:1E:67:21	Virgin Media	2412/20/gn	-73	-106	33	
AP	C8:0C:C8:9F:EB:6C	TALKTALK9FEB66	2432/20/gn	-91	-106	15	
AP	D2:05:C2:15:11:B1	Virgin Media	2437/20/gn	-90	-106	16	
AP	80:37:73:1F:B1:B0	VM890683-2G	2437/20/gn	-84	-106	22	
AP	C0:05:C2:15:11:B1	VM8313735	2437/20/gn	-87	-106	19	
AP	2C:B0:5D:D7:F6:25	virginmedia1220263	2437/20/gn	-88	-106	18	
AP	00:8E:F2:CE:D1:5A	virginmedia1149311	2462/20/gn	-90	-106	16	
AP	5A:D3:43:42:90:19	Virgin Media	2462/20/gn	-84	-106	22	
AP	48:D3:43:42:90:19	VM9403391	2462/20/gn	-87	-106	19	
AP	90:21:06:1C:B0:79	SKYC6D1B	2462/20/gn	-90	-106	16	
AP	C4:10:8A:19:76:C8	Isomer	2472/20/gn	-89	-107	18	

Flags: A - active, P - privacy, R - routeros-network, N - nstreme, T - tdma, W - wds, B - bridge

	ADDRESS	SSID	CHANNEL	SIG	NF	SNR	RADIO-NAME
AP	80:37:73:3B:32:80	VM890683-5G	5180/20-Ce/ac/P	-84	-105	21	
AP	40:0D:10:1E:67:27	VM7685819	5220/20-eeCe/ac/P	-73	-103	30	
AP	90:21:06:D9:4F:FD	SKY83AFB	5180/20-Ceee/ac/P	-86	-105	19	

status: finished

downloaded: 0KiBC-z pause]

duration: 1s

Coordinates found:

Latitude: 51.4732423

Longitude: -0.8555551

Accuracy: 29.0 m

Direct Google Maps link:

<https://www.google.com/maps/?q=51.4732423,-0.8555551>

[admin@MikroTik] >





# Few questions we're going to answer

- Where can we have the scripts on the router?
- What can we do with the scripts?
- How can we run a script?
- How can we make the script nice and clear?

The scripts - where?

# The scripts - where? - in the CLI!

```
:foreach lease in=[/ip dhcp-server lease find] do={
  :local mac [/ip dhcp-server lease get $lease mac-address];
  :local name [/ip dhcp-server lease get $lease host-name];
  :local ip [/ip dhcp-server lease get $lease address];
  :put "MAC address: $mac, IP: $ip, host name: $name"
}
```

# The scripts - where? - in the CLI!

```
[admin@MikroTik] > :foreach lease in=[/ip dhcp-server lease find] do={:local mac [/ip dhcp-server lease get $lease mac-address]; :local name [/ip dhcp-server lease get $lease host-name]; :local ip [/ip dhcp-server lease get $lease address]; :put "MAC address: $mac, IP: $ip, host name: $name"}
```

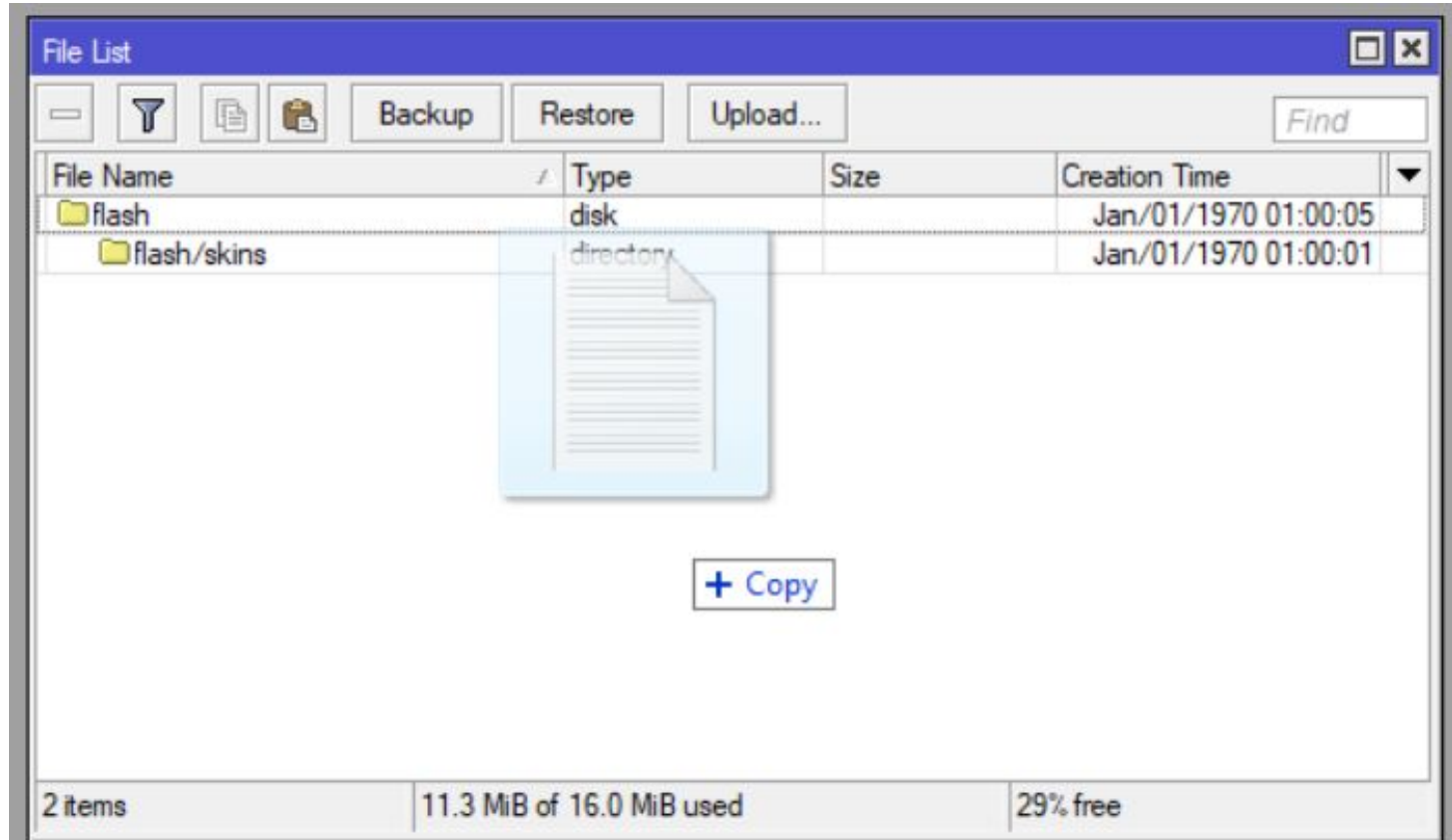
MAC address:	94:65:2D:C7:D5:55,	IP:	10.255.0.253,	host name:	OnePlus_5T
MAC address:	58:FB:84:8F:EE:2C,	IP:	10.255.0.252,	host name:	DESKTOP-C2IH9JK
MAC address:	30:07:4D:A4:48:93,	IP:	10.255.0.251,	host name:	Galaxy-S8
MAC address:	00:0C:42:F9:97:18,	IP:	10.255.0.3,	host name:	Cafe
MAC address:	E4:8D:8C:17:D7:36,	IP:	10.255.0.248,	host name:	MikroTik
MAC address:	00:08:9B:F8:6A:05,	IP:	10.255.0.5,	host name:	NASF86A05
MAC address:	18:DB:F2:15:57:C8,	IP:	10.255.0.99,	host name:	AIR007027
MAC address:	DC:9F:DB:80:9D:1A,	IP:	10.255.0.11,	host name:	AirCam
MAC address:	F0:23:B9:42:B4:AA,	IP:	10.255.0.12,	host name:	H.VIEW
MAC address:	CC:2D:E0:81:0E:2F,	IP:	10.255.0.2,	host name:	MikroTik
MAC address:	98:29:A6:46:97:03,	IP:	10.255.0.250,	host name:	LAPTOP-UNJMN524

```
[admin@MikroTik] > █
```

The scripts - where? - in the CLI!

```
[admin@MikroTik] > {  
{... :local x 1  
{... :local y 2  
{... :put "1 + 2 =  $\$(\$x+\$y)$ "  
{... }  
1 + 2 = 3  
[admin@MikroTik] > █
```

# The scripts - where? - in the file!

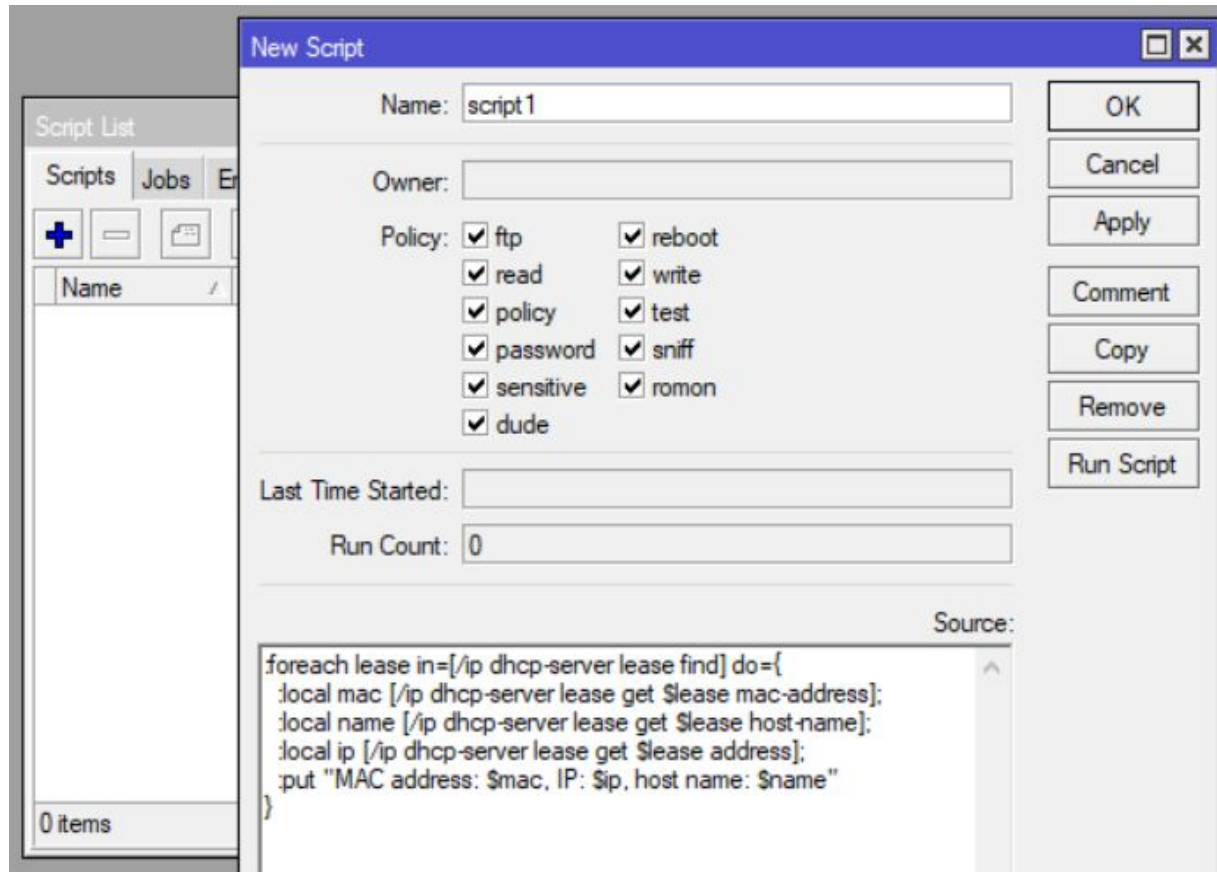


## The scripts - where? - in the file!

```
[admin@MikroTik] > /import script.txt
MAC address: 94:65:2D:C7:D5:55, IP: 10.255.0.253, host name: OnePlus_5T
MAC address: 58:FB:84:8F:EE:2C, IP: 10.255.0.252, host name: DESKTOP-C2IH9JK
MAC address: 30:07:4D:A4:48:93, IP: 10.255.0.251, host name: Galaxy-S8
MAC address: 00:0C:42:F9:97:18, IP: 10.255.0.3, host name: Cafe
MAC address: E4:8D:8C:17:D7:36, IP: 10.255.0.248, host name: MikroTik
MAC address: 00:08:9B:F8:6A:05, IP: 10.255.0.5, host name: NASF86A05
MAC address: 18:DB:F2:15:57:C8, IP: 10.255.0.99, host name: AIR007027
MAC address: DC:9F:DB:80:9D:1A, IP: 10.255.0.11, host name: AirCam
MAC address: F0:23:B9:42:B4:AA, IP: 10.255.0.12, host name: H.VIEW
MAC address: CC:2D:E0:81:0E:2F, IP: 10.255.0.2, host name: MikroTik
MAC address: 98:29:A6:46:97:03, IP: 10.255.0.250, host name: LAPTOP-UNJMN524

Script file loaded and executed successfully
[admin@MikroTik] > █
```

# The scripts - where? - in the scripts!





# The scripts - where? - in the scripts!

```
[admin@MikroTik] > /system script run script1
MAC address: 94:65:2D:C7:D5:55, IP: 10.255.0.253, host name: OnePlus_5T
MAC address: 58:FB:84:8F:EE:2C, IP: 10.255.0.252, host name: DESKTOP-C2IH9JK
MAC address: 30:07:4D:A4:48:93, IP: 10.255.0.251, host name: Galaxy-S8
MAC address: 00:0C:42:F9:97:18, IP: 10.255.0.3, host name: Cafe
MAC address: E4:8D:8C:17:D7:36, IP: 10.255.0.248, host name: MikroTik
MAC address: 00:08:9B:F8:6A:05, IP: 10.255.0.5, host name: NASF86A05
MAC address: 18:DB:F2:15:57:C8, IP: 10.255.0.99, host name: AIR007027
MAC address: DC:9F:DB:80:9D:1A, IP: 10.255.0.11, host name: AirCam
MAC address: F0:23:B9:42:B4:AA, IP: 10.255.0.12, host name: H.VIEW
MAC address: CC:2D:E0:81:0E:2F, IP: 10.255.0.2, host name: MikroTik
MAC address: 98:29:A6:46:97:03, IP: 10.255.0.250, host name: LAPTOP-UNJMN524
[admin@MikroTik] > █
```

# The scripts - where? - other places

The image shows three overlapping configuration windows in Mikrotik WinBox, all containing the same beep script:

```
:beep frequency=880 length=500ms  
:delay 1s  
:beep frequency=660 length=500ms  
:delay 1s  
:beep frequency=440 length=500ms  
:delay 1s
```

**Schedule <schedule1>**  
Name: schedule1  
Start Date: Oct/06/2018  
Start Time: startup  
Interval: 00:00:10  
Owner: admin  
Policy:  ftp,  read,  policy,  password,  sensitive,  dude,  reboot,  write,  test,  sniff,  romon  
Run Count: 5  
Next Run: Oct/06/2018 20:33:34  
On Event: (script text)  
Status: disabled

**DHCP Client <ether1>**  
DHCP Options: hostname, clientid  
Default Route Distance: 1  
Script: (script text)  
Status: enabled

**New Hotspot User Profile**  
On Login: (script text)  
On Logout: (script text)

**New Network Host**  
Host: Up, Down  
On Up: (script text)  
Status: enabled

What can we do?

```
put -- prints argument on the screen
queue -- Bandwidth management
quit -- Quit console
radius -- Radius client settings
redo -- Redo previously undone action
resolve -- perform a dns lookup of domain name
return -- return value from function
routing --
set -- Change item properties
snmp -- SNMP settings
special-login -- Special login users
system --
terminal -- commands related to terminal handling
time -- returns time taken by command to execute
toarray -- convert argument to array value
tobool -- convert argument to truth value
toid -- convert argument to internal number value
toip -- convert argument to IP address value
toip6 -- convert argument to IPv6 address value
tonum -- convert argument to integer number value
tool -- Diagnostics tools
tostr -- convert argument to string value
totime -- convert argument to time interval value
typeof -- return type of value
undo -- Undo previous action
user -- User management
while -- executes command while condition is true
export -- Print or save an export script that can be used
```

# What can we do? - the magic :commands

```
:for i from=440 to=880 step=40 do={  
  :put "Now beeping at $i MHz";  
  :beep frequency=$i length=1s;  
  :delay 1s;  
}
```

# The :commands controlling the flow

- :if
- :for
- :foreach
- :do ... while
- :while ... do
- :delay
- :return

# The :commands working on variables

- :local
- :global
- :set
- :typeof
- :tonum, :toarray, :tobool, :tostr

# The :commands interacting with user

- :put
- :log
- :beep
- :blink

# The :commands working on strings

- :find
- :pick
- :len

```
:local text "abcde"
```

```
:put [:pick $text 1 [:find $text "d"]]
```

```
bc
```



# Other useful RouterOS commands

- `/tool e-mail send`
- `/tool sms send`
- `/tool fetch`
- `/ping`
- `/file get ... contents`
- `/file set ... contents=...`
- `/tool snmp-get`

How can we run a script?

# Our example script

```
:if ([/system leds get [find] type]="off") do={  
  /system leds set [find] type=on;  
}  
else={  
  /system leds set [find] type="off";  
}
```

How can we run a script?  
-> scheduler

New Schedule

Name:

Start Date:

Start Time:  ▾

Interval:

Owner:

Policy:

<input checked="" type="checkbox"/> ftp	<input checked="" type="checkbox"/> reboot
<input checked="" type="checkbox"/> read	<input checked="" type="checkbox"/> write
<input checked="" type="checkbox"/> policy	<input checked="" type="checkbox"/> test
<input checked="" type="checkbox"/> password	<input checked="" type="checkbox"/> sniff
<input checked="" type="checkbox"/> sensitive	<input checked="" type="checkbox"/> romon
<input checked="" type="checkbox"/> dude	

Run Count:

Next Run:

On Event:

led-blink

Buttons: OK, Cancel, Apply, Disable, Comment, Copy, Remove

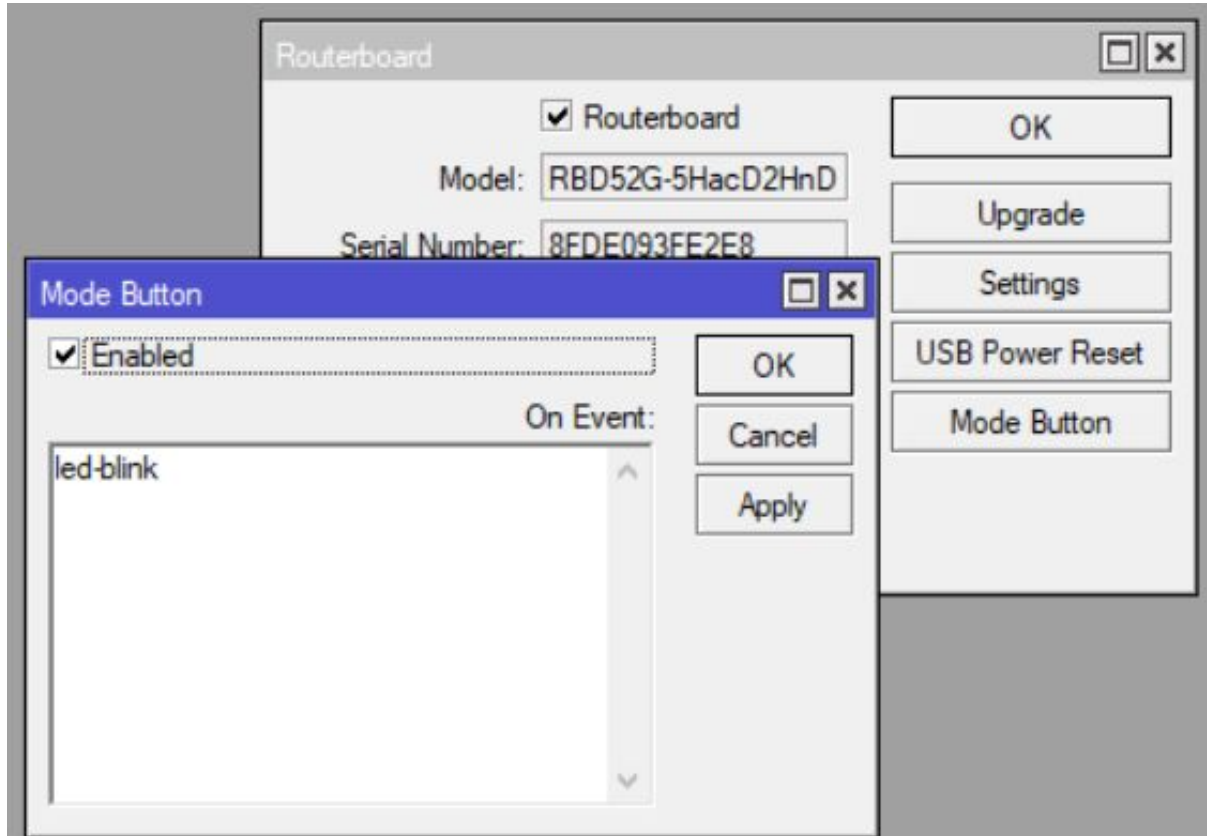
# How can we run a script? -> triggers

The image shows three overlapping configuration windows in Mikrotik WinBox, all containing a script for a beeping sound. The script is: `:beep frequency=880 length=500ms :delay 1s :beep frequency=660 length=500ms :delay 1s :beep frequency=440 length=500ms :delay 1s`

- Schedule <schedule1>**: Name: schedule1, Start Date: Oct/06/2018, Start Time: startup, Interval: 00:00:10, Owner: admin. Policy options include ftp, read, policy, password, sensitive, reboot, write, test, sniff, romon, and dude. Run Count: 5, Next Run: Oct/06/2018 20:33:34. The script is listed under "On Event:".
- DHCP Client <ether1>**: DHCP Options: hostname, clientid. Default Route Distance: 1. The script is listed under "Script:".
- New Hotspot User Profile**: The script is listed under "On Login:" and "On Logout:".

A fourth window, **New Network Host**, is also visible, containing the same script under "On Up:".

How can we run a script? -> mode button



# How can we run a script? -> FTP upload file.auto.rsc

The screenshot shows an FTP client interface with two panels. The top panel contains navigation and session controls, including 'Synchronize', 'Queue', and 'Transfer Settings Default'. Below this, the local directory path is 'C:\Users\rejes\Desktop\birmingham\' and the remote directory path is '/'. The local directory view shows a file named 'script.auto.rsc' (1 KB, RSC File) and a folder named 'screens'. The remote directory view shows a folder named 'flash' and two files: 'script.auto.log' (1 KB, changed 02/01/2018 00:14) and 'script.auto.rsc' (1 KB, changed 02/01/2018 00:14).

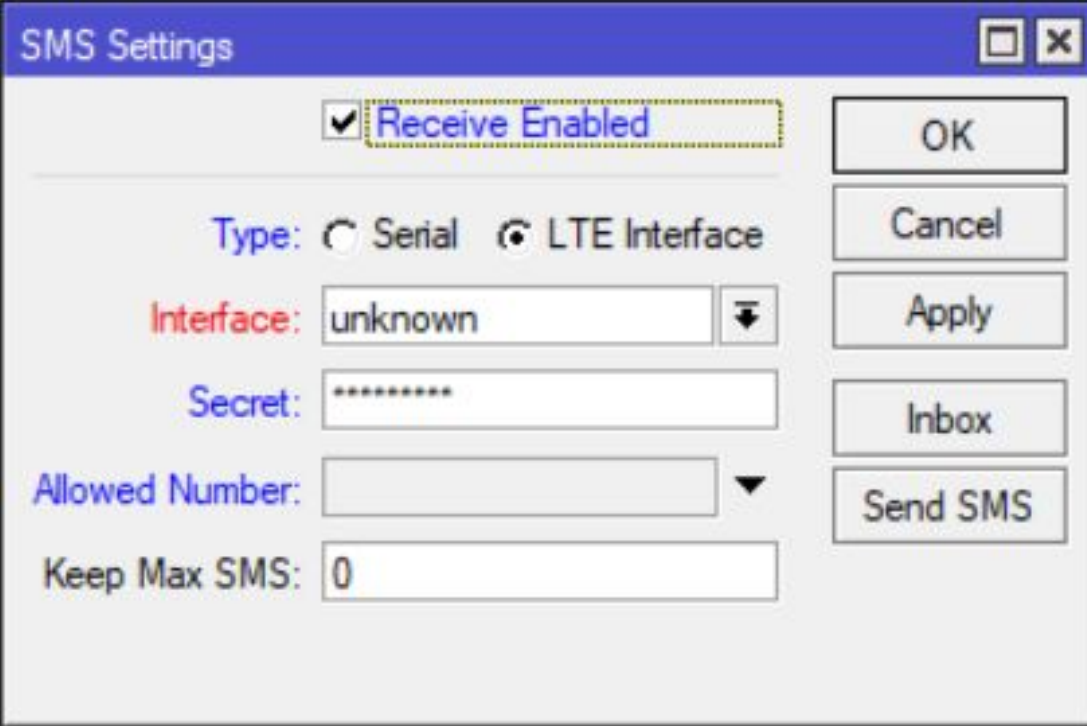
Name	Size	Type	Ch
..		Parent directory	07
screens		File folder	07
script.auto.rsc	1 KB	RSC File	07

Name	Size	Changed
..		
flash		01/01/2018 00:00
script.auto.log	1 KB	02/01/2018 00:14
script.auto.rsc	1 KB	02/01/2018 00:14

# How can we run a script? -> SMS to the router

:cmd SECRET script NAME



The image shows a dialog box titled "SMS Settings" with a blue header bar. The dialog contains several configuration options and a set of control buttons on the right side.

- Receive Enabled:** A checkbox that is checked, with the text "Receive Enabled" highlighted by a yellow dashed border.
- Type:** Radio buttons for "Serial" and "LTE Interface". The "LTE Interface" option is selected.
- Interface:** A text field containing "unknown" and a dropdown arrow.
- Secret:** A text field containing a series of asterisks "\*\*\*\*\*".
- Allowed Number:** A text field with a dropdown arrow.
- Keep Max SMS:** A text field containing the number "0".

On the right side of the dialog, there are five buttons stacked vertically: "OK", "Cancel", "Apply", "Inbox", and "Send SMS".



# How can we run a script? -> SNMP GET or SET

- We need SNMP community with write access (even for GET)
- We need to find the Script OIDs with snmpwalk
- Script can report a value with :return (string only)

```
$ snmpwalk -v2c -cpublic 192.168.88.1 1.3.6.1.4.1.14988.1.1.8
iso.3.6.1.4.1.14988.1.1.8.1.1.2.1 = STRING: "script1"
iso.3.6.1.4.1.14988.1.1.8.1.1.2.2 = STRING: "script2"
iso.3.6.1.4.1.14988.1.1.8.1.1.3.1 = INTEGER: 0
iso.3.6.1.4.1.14988.1.1.8.1.1.3.2 = INTEGER: 0
```

# How can we run a script? -> SNMP GET

```
[admin@MikroTik] > tool snmp-walk address=127.0.0.1 community=private version=2c  
oid=1.3.6.1.4.1.14988.1.1.8
```

OID	TYPE	VALUE
1.3.6.1.4.1.14988.1.1.8.1.1.2.1	octet-string	led-blink
1.3.6.1.4.1.14988.1.1.8.1.1.2.2	octet-string	exp
1.3.6.1.4.1.14988.1.1.8.1.1.3.1	integer	0
1.3.6.1.4.1.14988.1.1.8.1.1.3.2	integer	0

```
[admin@MikroTik] > tool snmp-get address=127.0.0.1 community=private version=2c  
oid=1.3.6.1.4.1.14988.1.1.18.1.1.2.1
```

OID	TYPE	VALUE
1.3.6.1.4.1.14988.1.1.18.1.1...	octet-string	The LED is now ON\n

```
[admin@MikroTik] > █
```

# Variables

# Using variables

- **:local x** - variable **\$x** visible only inside this “scope”
- **:global x** - variable **\$x** visible everywhere (in the System Environment”)
  
- **:local x 1** - setting the variable value when initializing
- **:set \$x 1** - setting the variable name anywhere else

# Variables - arrays

```
:foreach lease in=[/ip dhcp-server lease find] do={  
    :local mac [/ip dhcp-server lease get $lease mac-address];  
    :local name [/ip dhcp-server lease get $lease host-name];  
    :local ip [/ip dhcp-server lease get $lease address];  
    :put "MAC address: $mac, IP: $ip, host name: $name"  
}
```

## Variables - custom arrays

```
:local colors [:toarray ""]  
:set ($colors->"sun") "yellow"  
:set ($colors->"sky") "blue"  
:set ($colors->"grass") "green"  
  
:put "The color of the grass is:"  
:put ($colors->"grass")
```

## Variables - custom arrays

```
:foreach element,color in=$colors do={  
  :put "$element is $color"  
}
```

grass is green

sky is blue

sun is yellow

# Functions



# Functions - how to define them

```
:global function do={  
    :return "This is the result!"  
}
```

## Functions - how to run them

```
[admin@MikroTik] > $function  
[admin@MikroTik] >  
[admin@MikroTik] > :put [$function]  
This is the result!  
[admin@MikroTik] > █
```

## Functions - how NOT TO run them

```
[admin@MikroTik] > :put $function  
;(eval / (eval /returnvalue=This is the result!))  
[admin@MikroTik] > █
```

We need to RUN the function.

:put \$function - wrong!

:put [\$function] - right!

# Functions - how to pass arguments

```
:global exp do={
  :local result 1;
  :for i from=1 to=$2 do={
    :set $result ($result*$1);
  }
  :return $result
}

:put [$exp 2 8]
```

## Functions - running them with arguments

```
[admin@MikroTik] > :put [$exp 2 8]
```

```
256
```

```
[admin@MikroTik] > :put [$exp 10 9]
```

```
1000000000
```

```
[admin@MikroTik] > █
```

# Functions - and the local/global scopes

- Functions can be defined as local
- Better to define functions as global
- Functions used by other functions **NEED TO** be defined as global

```
:global function1 do={...}
```

```
:local function1 do={...}
```

```
:global function2 do={  
  :global function1;  
  ... (using [$function1])  
}
```

```
:local function2 do={  
  :local function1;  
  ... (using [$function1])  
}
```

# Functions - how I use them

```
:global pushover do={
:global urlEncode;
  :if ([:typeof $message]!="nothing") do={
    :local api "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx";
    :local user "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx";
    :local urlmessage [$urlEncode $message];
    :local string "token=$api&user=$user&message=$urlmessage";
    /tool fetch mode=https url="https://api.pushover.net/1/messages.json"
http-method=post http-data="$string";
  }
}
```

```
$pushover message="There is a problem with the router!"
```

# Functions - something special

```
:global input do={  
    :return  
}
```

```
:put "Please provide the value for x:"  
:local x [$input]  
:put "Please provide the value for y:"  
:local y [$input]  
:put "$x*$y=$(( $x*$y))"
```

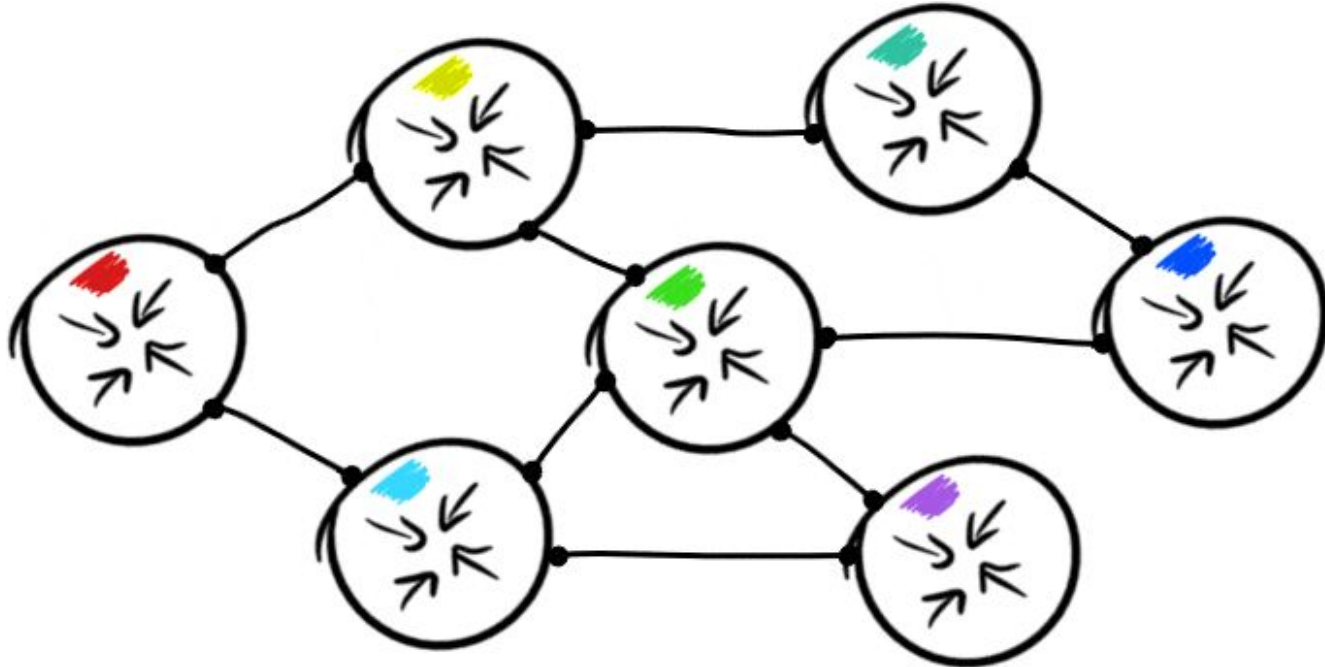


## Functions - something special

```
[admin@MikroTik] > system script run multiplication
Please provide the value for x:
value: 4
Please provide the value for y:
value: 9
4*9=36
[admin@MikroTik] > █
```

# Playing battleships over BGP

- Introduced by **Ben Cox**: <https://blog.benjojo.co.uk/post/bgp-battleships>



	YOU									
	1	2	3	4	5	6	7	8	9	10
A										
B										
C										
D										
E										
F										
G										
H										
I										
J										

	ENEMY									
	1	2	3	4	5	6	7	8	9	10
A										
B										
C										
D										
E										
F										
G										
H										
I										
J										

Please, enter coordinates and direction (L,R,D,U) to deploy the ship, e.g. D5R  
 This ship will have the length of 4 squares

value:

	YOU									
	1	2	3	4	5	6	7	8	9	10
A		[ ]		[ ]						
B						[ ]				
C			[ ]							[ ]
D			[ ]				[ ]			[ ]
E			[ ]							
F			[ ]							[ ]
G					[ ]	[ ]	[ ]			[ ]
H										
I		[ ]	[ ]	[ ]		[ ]				
J										

	ENEMY									
	1	2	3	4	5	6	7	8	9	10
A										
B										
C										
D										
E										
F										
G										
H										
I										
J										

Please, enter coordinates and direction (L,R,D,U) to deploy the ship, e.g. D5R  
 This ship will have the length of 1 squares

value: E4

	YOU										
	1	2	3	4	5	6	7	8	9	10	
A		[ ]		[ ]							
B						[ ]					
C		[ ]									[ ]
D		[ ]					[ ]				[ ]
E		[ ]			[ ]						
F		[ ]									[ ]
G					[ ]	[ ]	[ ]				[ ]
H											
I		[ ]		[ ]		[ ]					
J											

	ENEMY										
	1	2	3	4	5	6	7	8	9	10	
A											
B											
C											
D											
E											
F											
G											
H											
I											
J											

Your turn. Please fire choosing the coordinates  
e.g. A1, g5, J10

value: B2

	YOU										
	1	2	3	4	5	6	7	8	9	10	
A		[ ]		[ ]							
B						[ ]					
C		[ ]								[ ]	
D		[ ]				[ ]				[ ]	
E		[ ]			[ ]						
F		[ ]								[ ]	
G					[ ]	[ ]	[ ]			[ ]	
H											
I		[ ]	[ ]	[ ]		[ ]					
J											

	ENEMY									
	1	2	3	4	5	6	7	8	9	10
A										
B										
C										
D										
E										
F										
G										
H										
I										
J										

Shooting B2 - waiting for response...

	YOU										
	1	2	3	4	5	6	7	8	9	10	
A		[ ]		[ ]							
B						[ ]					
C		[ ]									[ ]
D		[ ]					[ ]				[ ]
E		[ ]			[ ]						
F		[ ]									[ ]
G					[ ]	[ ]	[ ]				[ ]
H											
I		[ ]		[ ]		[ ]					
J											

	ENEMY										
	1	2	3	4	5	6	7	8	9	10	
A											
B			:	:							
C											
D											
E											
F											
G											
H											
I											
J											

Your shot on B2 was a MISS.  
 Waiting for your opponent's action...

	YOU									
	1	2	3	4	5	6	7	8	9	10
A	##									
B										
C										
D										
E										
F										
G										
H										
I										
J										

	ENEMY									
	1	2	3	4	5	6	7	8	9	10
A										
B										
C										
D										
E										
F										
G										
H										
I										
J										

Your shot on B2 was a MISS.  
 Your opponent tried A1 and HIT.  
 Your turn. Please fire choosing the coordinates

value:



	YOU									
	1	2	3	4	5	6	7	8	9	10
A	##	##	::							
B	::	::	::		[]					
C	[]									[]
D	[]				[]				[]	
E	[]			[]						
F	[]					::			[]	
G				[]	[]	##		::		[]
H								::		
I	[]	[]	[]		[]					
J										

	ENEMY									
	1	2	3	4	5	6	7	8	9	10
A	::		::							
B		::								
C										
D				::						
E					::					
F					##	##				
G					::					
H					::					
I										
J										

Your shot on G6 was a HIT.  
 Your opponent tried F6 and HIT.  
 Your turn. Please fire choosing the coordinates

value:

	YOU									
	1	2	3	4	5	6	7	8	9	10
A	## ## ::				::					
B	:: :: ::		[ ]				:: ::			
C	[ ]							:: ##		
D	[ ]				[ ]			:: ##		
E	[ ]		[ ]				:: ::			
F	[ ]		:: :: ::	:: ::		[ ]				
G			:: ## ## ##	:: ::		[ ]				
H			:: :: ::	:: ::				::		
I	[ ]	[ ]	[ ]		[ ]				::	
J										

	ENEMY									
	1	2	3	4	5	6	7	8	9	10
A	::	::							::	
B		::								
C								##		
D				::						
E	::	::	:: ::	:: ::	:: ::					
F			:: ## ## ##	::						
G	::	::	:: ::	:: ::	:: ::					
H					::					
I	::	::	::	::	::					
J	##	::	## ##	::						

Your shot on C8 was a HIT.  
 Your opponent tried I10 and MISSED.  
 Your turn. Please fire choosing the coordinates

value:

	YOU									
	1	2	3	4	5	6	7	8	9	10
A	## ## ::	::	::	::	::	::	::	::	::	::
B	::	::	::	[]				::	::	::
C	## ::	::		::	::	##				
D	## ::	::	::	::	[]			::	##	
E	## ::	::	##	::	::	::		::	::	
F	## ::	::	::	::	::	::		::	##	
G	::	::	::	##	##	##	::	::	::	##
H	::	::	::	::	::	::	::		::	::
I	##	##	##	::	##	::		::	::	
J	::	::	::	::	::	::	::		::	

	ENEMY									
	1	2	3	4	5	6	7	8	9	10
A	::	::	::	::	##	::			::	
B	::	::	::	::	::	::	::	::	::	::
C	##	::			::	##	##	##	##	##
D	::	::	::	::	::	::	::	::	::	::
E	::	::	::	::	::	::	::	::	##	##
F	##	::	::	##	##	##	::	::	::	::
G	::	::	::	::	::	::	::			
H				::	::	##	::		::	::
I	::	::	::	::	::	##	::		::	##
J	##	::	##	##	::	::	::		::	##

Your shot on I6 was a HIT.  
 Your opponent tried C4 and MISSED.  
 !!YOU WIN!! :)

value:

# Questions?

