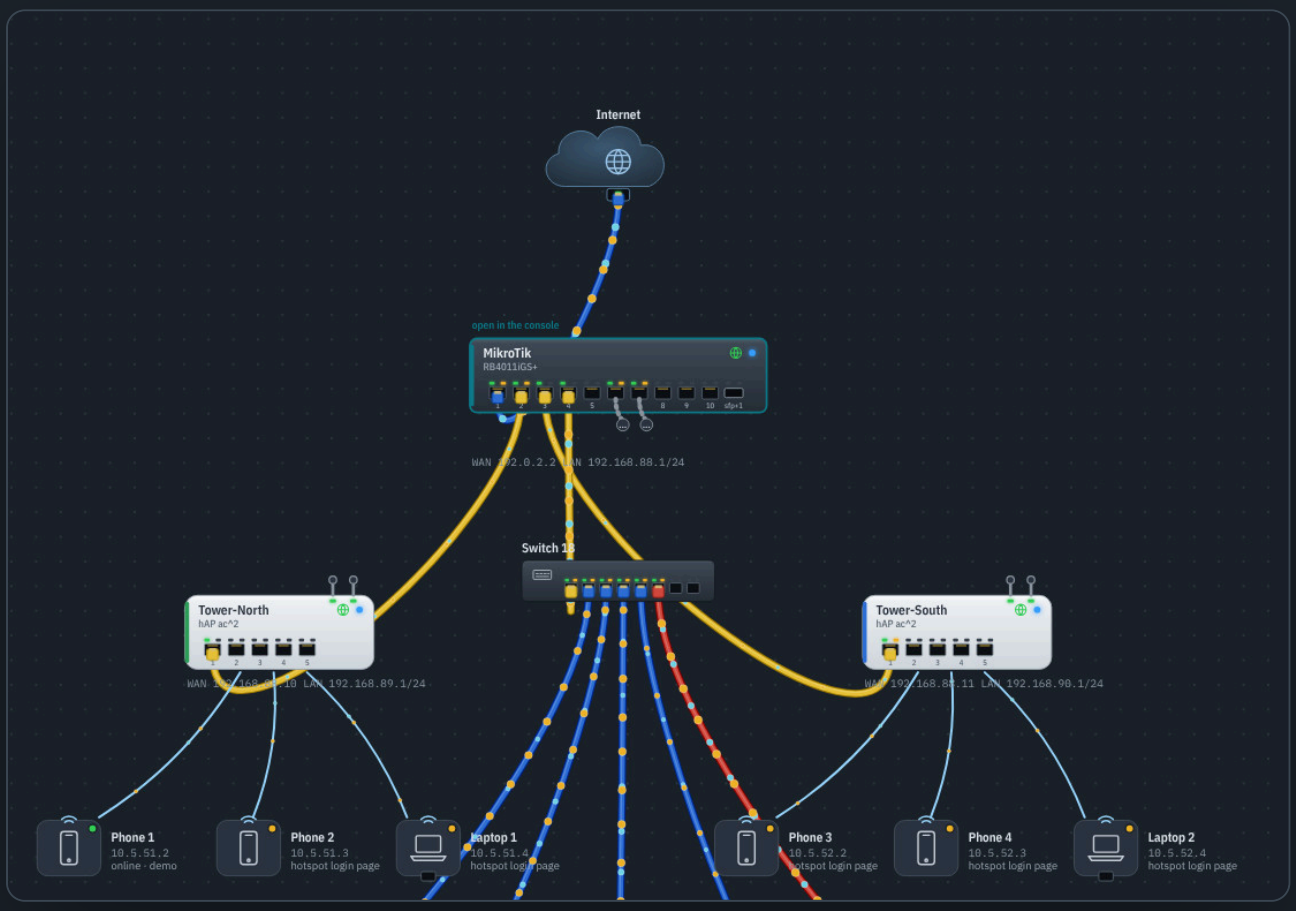
TikLab

TikLab 1.4

User manual

The RouterOS-compatible router simulator for Windows and Linux: train, test your apps, and build whole networks of MikroTik routers on one PC.





Version 1.4.0, November 2026

Contents

1 **Welcome to TikLab** 3
- What TikLab does 3
- What is in the package 3
- How this manual is organised 4

2 **Installing and starting** 5
- On Windows 5
- On Linux 5
- The start window 5
- Logging in 6

3 **The console at a glance** 7
- The main screen 7
- Working with windows 8
- Dashboard 8
- Menus, lists and editing 9
- Terminal 9
- About the real WinBox 10

4 **Simulating a router** 11
- Power and behaviour 11
- Vouchers 12
- Virtual clients and cables 12
- Router models 13

5 **Hotspot and vouchers** 14
- The login page 14
- End-to-end voucher test 14

6 **Connecting apps and TapTap** 15
- Services and ports 15
- TapTap over the API 15
- TapTap Link and Tunnel 15

7 **Wi-Fi on this PC** 17
- Broadcasting the router’s Wi-Fi 17
- Windows and Linux 17

8 **The Network Lab** 19
- The workbench 19
- Adding routers and devices 20
- Running cables 20
- What the cables do 20
- The inspector 21
- Logging a device in to a hotspot 22
- Configuring several routers 22
- Sample networks 23
- Reaching lab routers from other tools 23

9 **Moving configurations** 24
- Export to a real router 24
- Import a configuration 25
- Clone or send directly 26
- When TikLab cannot reach a router 26

10 **Files and hotspot pages** 28
- Files 28
- Hotspot Page Studio 28

11 **Scripts and WireGuard** 30
- RouterOS scripting 30
- WireGuard 30

12 **Reference** 31
- Command-line options 31
- Separate instances 31
- Data and resetting 32

13 **Troubleshooting** 33
- Problems and solutions 33

14 **Limitations and licences** 34
- Limitations 34
- Licences 34

1

Welcome to TikLab

TikLab is a portable program that behaves like a MikroTik router on your network. This chapter explains what it is for and what comes in the package.

What TikLab does

TikLab simulates a MikroTik router running RouterOS 7. Apps and people talk to it exactly as they would talk to a real router: over the RouterOS API, the REST API, SSH, Telnet, FTP and its hotspot login page. It is one program with no installation, and it runs on Windows and Linux.

Use it to:

- **train** staff on RouterOS without risking a customer router;
- **test apps** such as TapTap that create hotspot vouchers, PPPoE accounts and read live traffic through the API;
- **try hotspot login pages** on real phones before you put them on a router;
- **build whole networks** of several routers, switches and devices in the Network Lab, and configure each router from one console;
- **move configurations** between TikLab and real MikroTik routers in both directions.

What is in the package

Download the package for your system and unpack it anywhere, for example on the desktop or a USB stick.

Package	For	Contents
TikLab-1.4.0-windows-x64.zip	Windows 10 and 11 (64-bit)	`tiklab.exe`, `Start TikLab.bat`, `List router models.bat`, README, samples
TikLab-1.4.0-linux-x64.tar.gz	Linux PCs and servers (64-bit)	`tiklab`, `start-tiklab.sh`, README, samples
TikLab-1.4.0-linux-arm64.tar.gz	Raspberry Pi 4 and 5, other ARM boards	the same as Linux x64
TikLab-1.4.0-source.zip	developers	the source code, to build TikLab yourself

Everything TikLab saves (the configuration, keys, certificates, files and the Network Lab) goes into a folder called `tiklab-data` next to the program. To move TikLab to another PC, copy the program and that folder.

Note TikLab is an independent simulator. It is not made by or affiliated with MikroTik and does not contain RouterOS. It re-implements RouterOS behaviour and protocols so that apps and people can work with it as if it were a router. MikroTik, RouterOS and WinBox are trademarks of MikroTik.

How this manual is organised

Chapters 2 and 3 get TikLab running and show you around the console. Chapters 4 to 7 cover the simulated router, the hotspot, connecting apps such as TapTap, and broadcasting the router's Wi-Fi from your PC. Chapter 8 is about the Network Lab. Chapters 9 to 11 cover configuration transfer, files, login pages and scripting, and chapters 12 to 14 are reference: options, ports, troubleshooting and limitations.

All pictures are real screenshots of TikLab 1.4. In them, the PC running TikLab has the address **192.0.2.2**; yours will show your PC's own address.

2

Installing and starting

TikLab needs no installation. Unpack it, start it, and log in from your browser.

On Windows

Use the Windows package on Windows 10 or 11.

1. Unzip **TikLab-1.4.0-windows-x64.zip**.
2. Double-click **Start TikLab.bat** (or `tiklab.exe`).
3. If Windows SmartScreen warns about an unknown app, choose **More info** and then **Run anyway**.
4. When Windows Firewall asks, allow access on **Private networks**. Phones, other PCs and the TapTap app can then reach the simulator.
5. A console window opens with the addresses and ports, and the TikLab console opens in your browser.

> **Tip** Keep the black console window open while you use TikLab. Closing it, or pressing Ctrl+C in it, stops the simulator. The configuration is saved automatically.

On Linux

The Linux packages run on 64-bit PCs and servers, and on ARM boards such as the Raspberry Pi 4 and 5.

1. Unpack the archive: `tar xzf TikLab-1.4.0-linux-x64.tar.gz`
2. Run `./start-tiklab.sh` (or `./tiklab`).
3. Open the console address it prints if no browser opens by itself.

Ports below 1024 (80, 443, 22, 23 and 21) need root. Without root, TikLab moves those services to 8080, 8443, 2222, 2323 and 2121 automatically. To keep the standard ports without running as root, give the program the right once:

```
sudo setcap 'cap_net_bind_service=+ep' ./tiklab
```

The start window

When TikLab starts, it prints every address and port it uses. This is the real output from the PC used for this manual:

```
TikLab 1.4.0 - RouterOS-compatible router simulator
-------------------------------------------
- fresh configuration created (RB4011iGS+)
- model RB4011iGS+ (rb4011): 10 x 1G, 1 x SFP+, identity "MikroTik"
- data directory: /opt/TikLab-1.4.0-linux-x64/tiklab-data

WinBox-style GUI   http://192.0.2.2:8291/      (this PC: http://127.0.0.1:8291/)
RouterOS API       192.0.2.2:8728
RouterOS API-SSL   192.0.2.2:8729
REST API           http://192.0.2.2:80
Hotspot portal     http://192.0.2.2:80/login
REST API (HTTPS)   https://192.0.2.2:443
SSH                192.0.2.2:22
Telnet             192.0.2.2:23
FTP (files)        192.0.2.2:21

Login: admin (empty)   -   Discovery (MNDP) on UDP 5678: true
Press Ctrl+C to stop. Configuration is saved automatically.
```

If a port is already taken by another program, TikLab picks the next free one and shows that port here and in the console's status bar.

Logging in

The console opens in your browser at `http://127.0.0.1:8291/`. From another PC or a tablet, open `http://PC-address:8291/`.

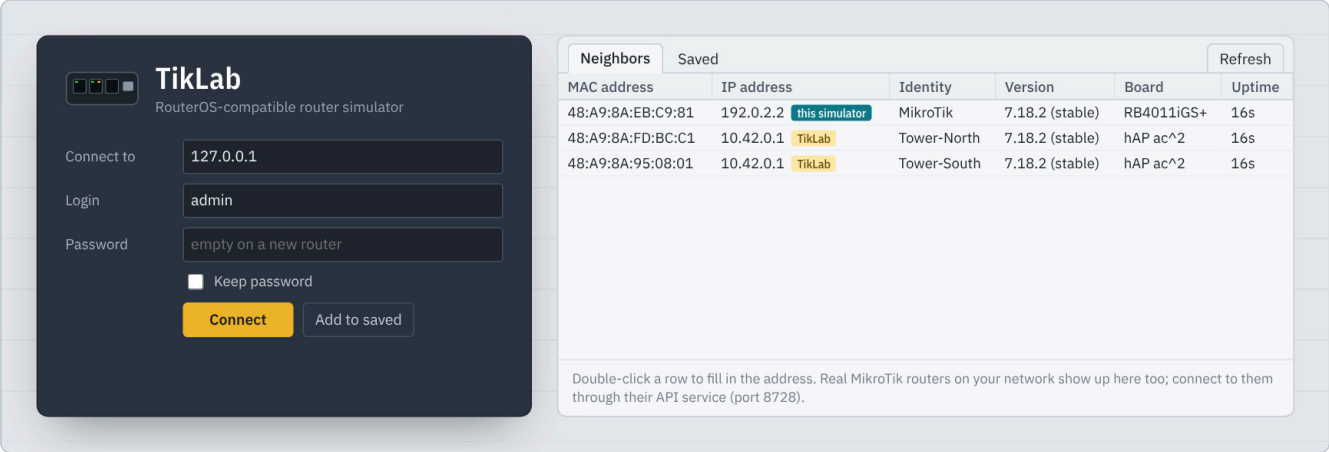


Figure 2.1 The login window. The Neighbors list shows this simulator, the other routers of your Network Lab, and real MikroTik routers found on the network.

Log in as **admin** with an **empty password**, which is how a brand-new MikroTik router ships. Then:

- **Connect to** `127.0.0.1` means this simulator. Any other address connects the console to a real MikroTik router through its RouterOS API (port 8728), so you can manage real routers with the same console.
- Double-click a row in **Neighbors** to fill in its address.
- **Add to saved** keeps routers you use often in the **Saved** tab. Tick **Keep password** only on a PC you trust.

3

The console at a glance

TikLab's console works like WinBox, in your browser. This chapter shows its parts and how to work with windows, lists and the terminal.

The main screen

After you log in, the console shows the menu on the left, the session and live figures at the top, your windows in the middle and the taskbar and service ports at the bottom. The Dashboard opens by itself.

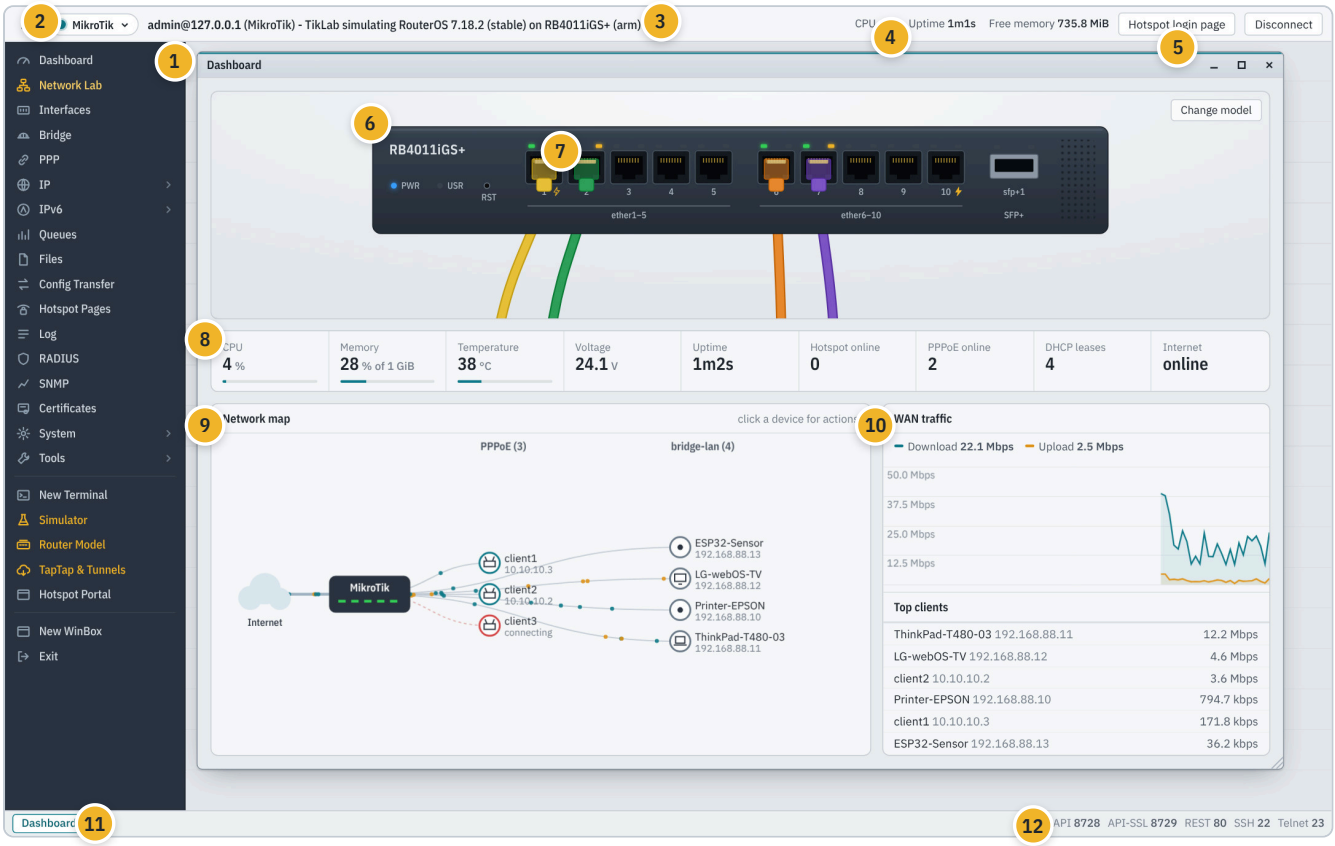


Figure 3.1 The console after logging in, with the Dashboard open.

1. **Menu.** The same layout as WinBox, plus TikLab's own tools in yellow.
2. **Router chip.** Which router you are configuring. Click it to switch routers (chapter 8).
3. **Session.** User, router address, identity, RouterOS version and board.
4. **Live figures.** CPU load, uptime and free memory.
5. **Hotspot login page.** Opens the router's captive portal in a new tab.
6. **Front panel.** The router's real port layout. LEDs show link and activity.
7. **Ports.** Click a port to plug or unplug its cable; double-click for its settings.
8. **Gauges.** CPU, memory, temperature, voltage, uptime, sessions, leases and internet.
9. **Network map.** The devices on the router with animated traffic. Click one for actions.
10. **WAN traffic and top clients.** Live graph and the busiest devices.
11. **Taskbar.** One button per open window. Click to bring it forward or minimise it.
12. **Service ports.** Where apps reach this router: API, API-SSL, REST, SSH, Telnet.

Working with windows

Windows in the console behave like windows on a desktop.

- Drag a window by its title bar; drag the bottom-right corner to resize it.
- The three title-bar buttons minimise, maximise and close. Double-clicking the title bar also maximises.
- Every open window has a button in the taskbar at the bottom.
- **New WinBox** opens a second console tab, for example to watch the log on one screen while you work on another.
- The menu button at the top left hides the menu to give the windows more room.

Dashboard

The Dashboard is a live picture of the router: its front panel, its health and its traffic.

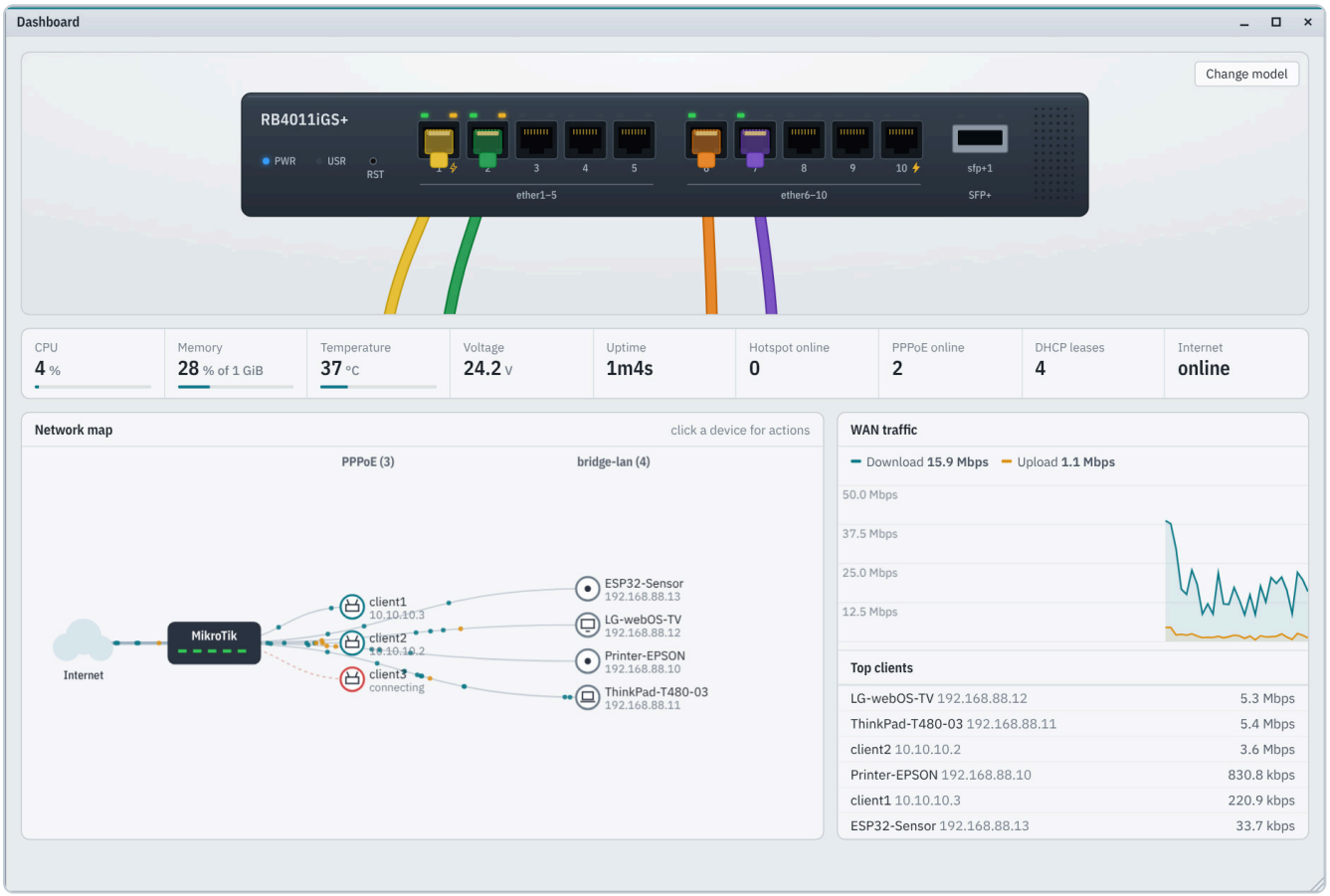


Figure 3.2 The Dashboard of the simulated RB4011iGS+ with cables in ports 1, 2, 6 and 7.

- The **front panel** matches the chosen model: Ethernet, SFP and SFP+ ports, PoE markers, and the board name. Link LEDs follow the port state and activity LEDs blink with the simulated traffic.
- **Click a port to plug or unplug its cable.** The interface goes down, its DHCP clients drop and the traffic stops, as on a real router.
- **Change model** switches the simulated hardware (see chapter 4).
- The **network map** shows PPPoE customers, LAN devices and hotspot users with their addresses. Click a device to log it in to the hotspot, disconnect it or change its settings.

- **WAN traffic** draws download and upload, and **Top clients** lists the busiest devices.

Menus, lists and editing

Menus open windows with the same tabs and columns as WinBox. Lists refresh by themselves.

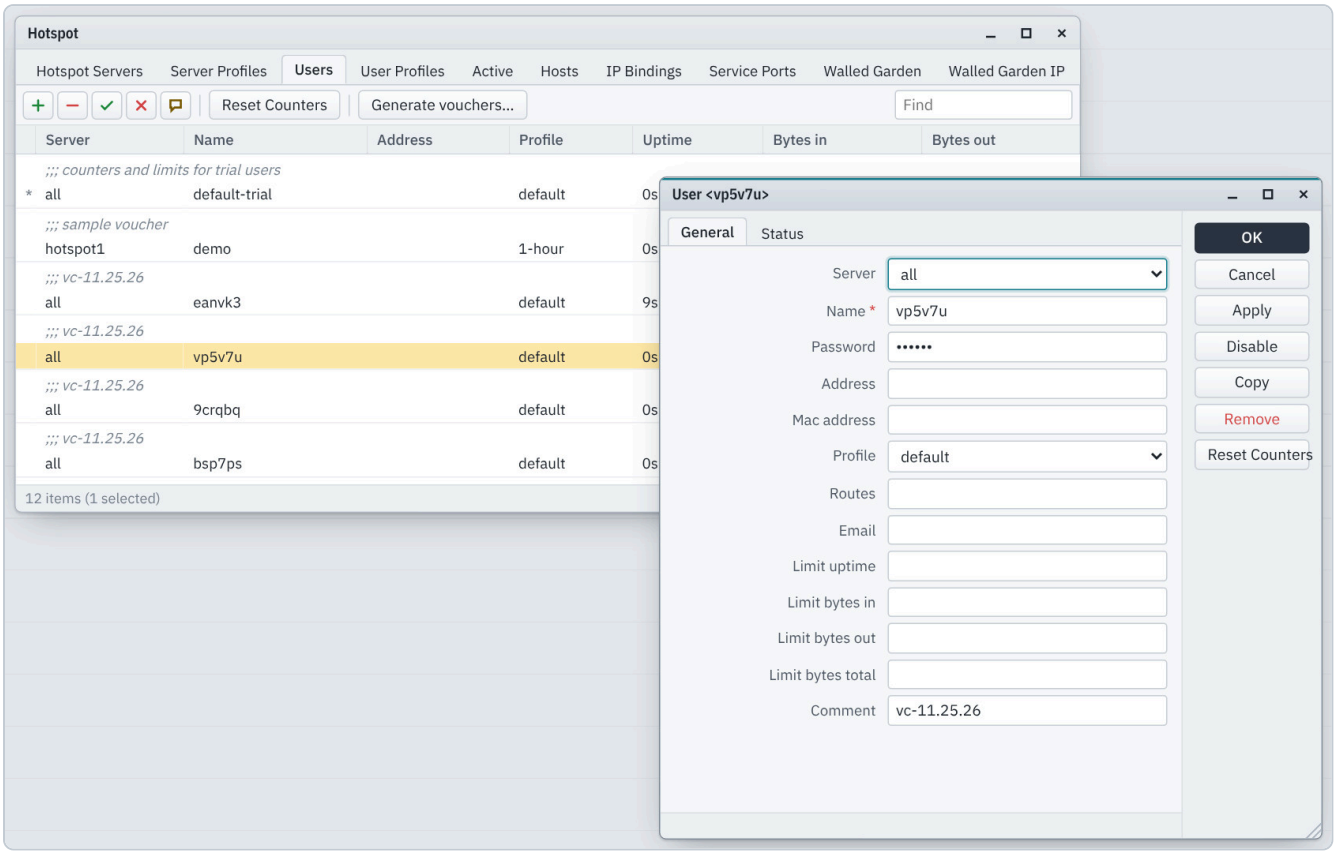


Figure 3.3 IP › Hotspot with the Users tab, and a voucher opened for editing.

- The toolbar buttons add (+), remove (–), enable (✓), disable (×) and comment the selected rows. Extra buttons appear where RouterOS has commands, such as **Reset Counters**.
- Double-click a row to open it. The dialog has the RouterOS fields with their defaults, and checks values the way RouterOS does: a wrong value is refused with RouterOS's own error message.
- **Find** filters the list as you type. Click a column header to sort.
- Dynamic entries (made by the router, such as DHCP leases or active sessions) are shown in blue.

Terminal

New Terminal opens the RouterOS command line. It is the same command line you get over SSH and Telnet.

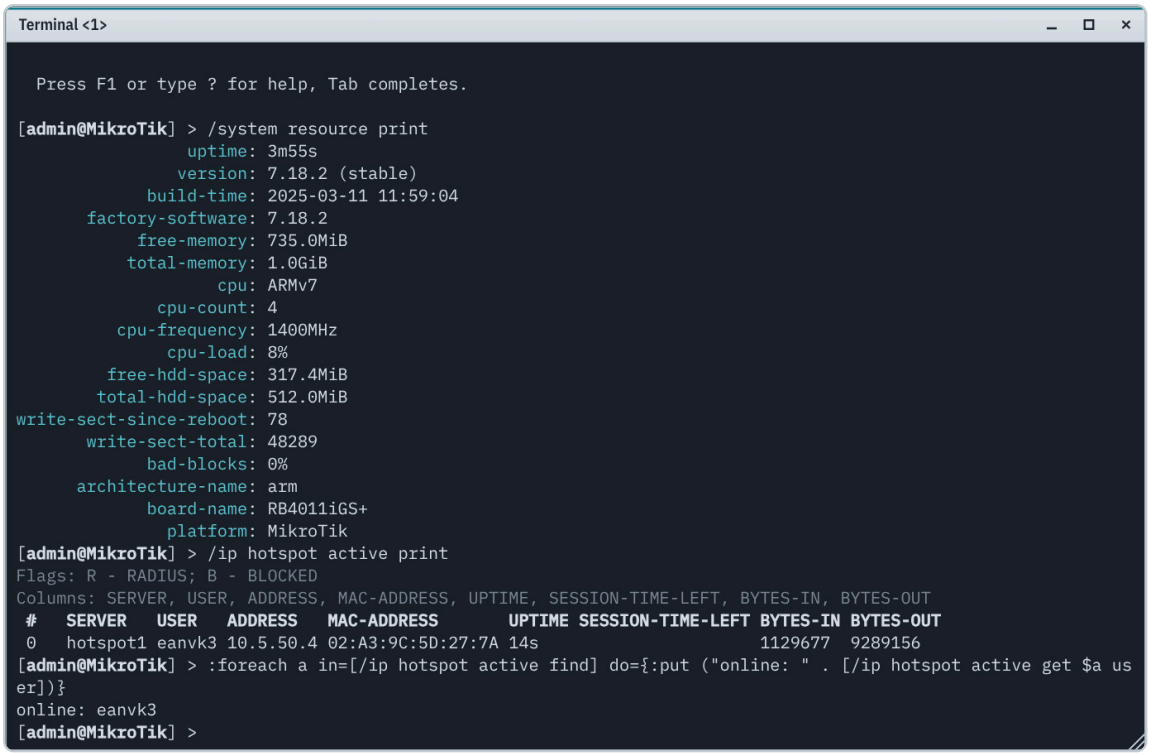


Figure 3.4 New Terminal: system resources, the active hotspot sessions, and a small script.

- `Tab` completes commands and parameters; `?` or `F1` shows help for the current menu.
- Commands work with paths, `print`, `detail`, `where`, `[find …]`, `export` and the scripting language (chapter 11).
- Pasting several lines runs them as one script, as in WinBox. This is how you paste install blocks from apps such as TapTap.
- `Ctrl`+`C` interrupts a running command, such as a ping or a monitor.
- You can open several terminals at once.

About the real WinBox

TikLab announces itself with MNDP, so it **shows up in real WinBox's Neighbors list** with its MAC address, IP address, identity, version and board. Real WinBox cannot log in to it, because WinBox needs MikroTik's own encrypted protocol and the GUI files that ship inside RouterOS. Use TikLab's console instead, at `http://PC-address:8291/`.

If you need genuine RouterOS, for example to practise with the real WinBox, MikroTik offers a free Cloud Hosted Router (CHR) image that runs in VirtualBox, VMware or QEMU.

4

Simulating a router

The Simulator window controls the things you would do with your hands on a real router: power, cables, customers and the hardware model.

Power and behaviour

Open **Simulator** in the menu. The first tab, **Power & behaviour**, powers the router and sets how busy the simulated network is.

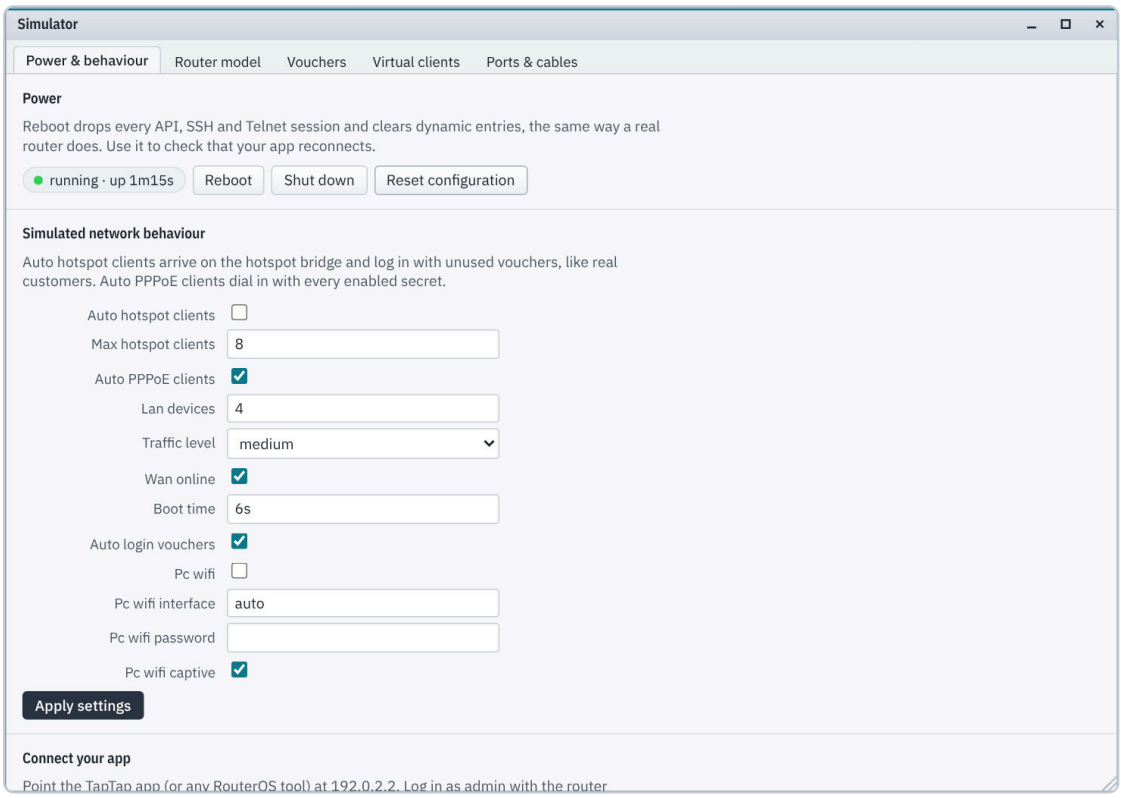


Figure 4.1 Simulator › Power & behaviour.

- **Reboot** drops every API, SSH and Telnet session and clears dynamic entries, as a real router does. Use it to check that your app reconnects.
- **Shut down** powers the router off: every service stops. The console then offers **Power on**.
- **Reset configuration** returns to the factory configuration of the current model.

The settings below change how the simulated network behaves. Press **Apply settings** after changing them.

Setting	What it does
Auto hotspot clients	Customers arrive on the hotspot by themselves and log in with unused vouchers, like real customers.
Max hotspot clients	How many of those customers there can be at once (8 by default).

Setting	What it does
Auto PPPoE clients	Simulated CPEs dial in with your PPP secrets.
LAN devices	How many devices (PCs, TVs, printers, sensors) sit on the LAN.
Traffic level	How busy the network is: low, medium, high or extreme.
WAN online	Switches the internet off and on, to test what your app shows when a site goes offline.
Boot time	How long a reboot takes (6 seconds by default).
Auto login vouchers	Lets the arriving customers use vouchers you created.
PC Wi-Fi settings	Used by Wi-Fi on this PC (chapter 7).

The window also lists the addresses and ports to point your app at.

Vouchers

The **Vouchers** tab creates batches of hotspot users, with the code as both the name and the password, exactly as a voucher app does through the API. Choose how many, the user profile, the code length, an optional prefix, an uptime limit and a data limit, and whether codes use letters and digits or digits only. Press **Generate**: the codes appear in the box, ready to copy, and in the list below.

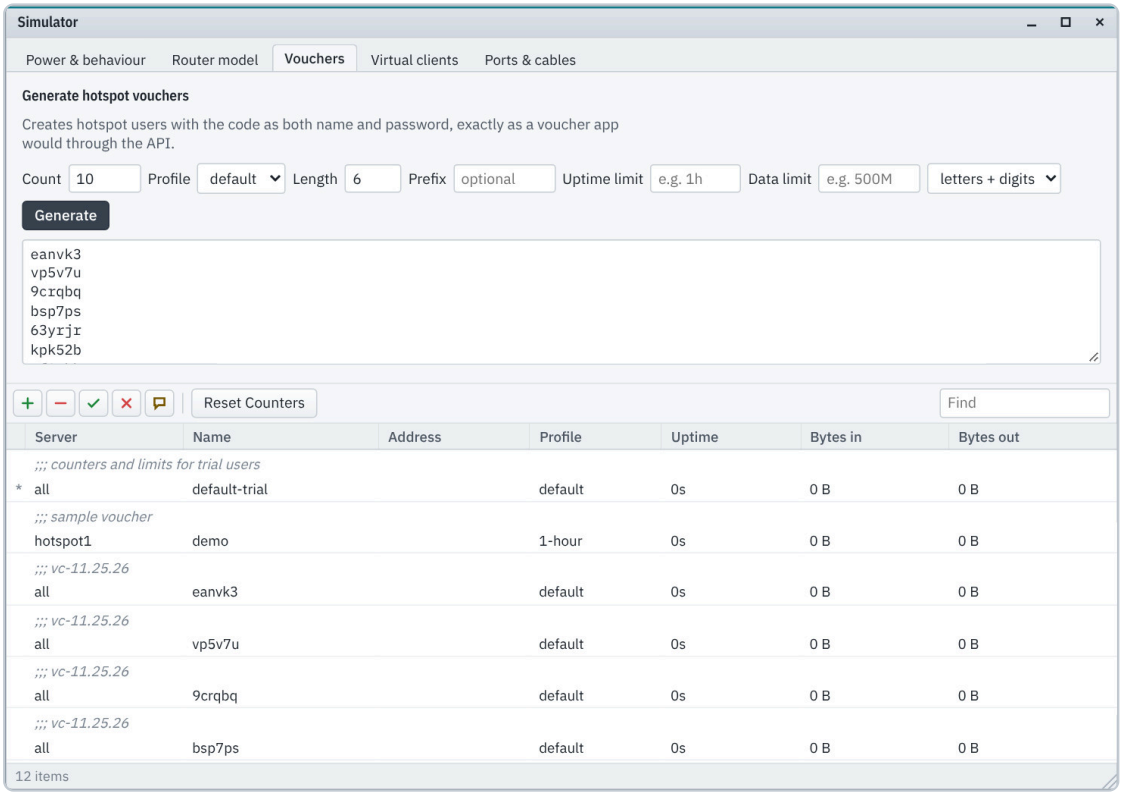


Figure 4.2 Simulator › Vouchers: ten codes generated for the default profile.

Virtual clients and cables

Two more tabs manage the simulated devices and the cables in the router's ports.

- **Virtual clients** lists the simulated devices: phones, laptops, PCs, TVs and PPPoE CPEs. Add your own, then connect, disconnect, log them in to the hotspot with a voucher, or log them out.
- **Ports & cables** lists every port with its cable, speed (10M to 10G) and cable colour. **Plug in** and **Unplug** do the same as clicking a port on the Dashboard.

Router models

TikLab simulates 24 MikroTik models. Each one reports its real board name, CPU, core count and memory to apps, and has the right ports: 10/100 or Gigabit, 2.5G, SFP, SFP+ and combo, with PoE markers. Wireless models get `wlan1` / `wlan2` (wireless package); the Wi-Fi 6 hAP ax models get `wifi1` / `wifi2` (wifi package).

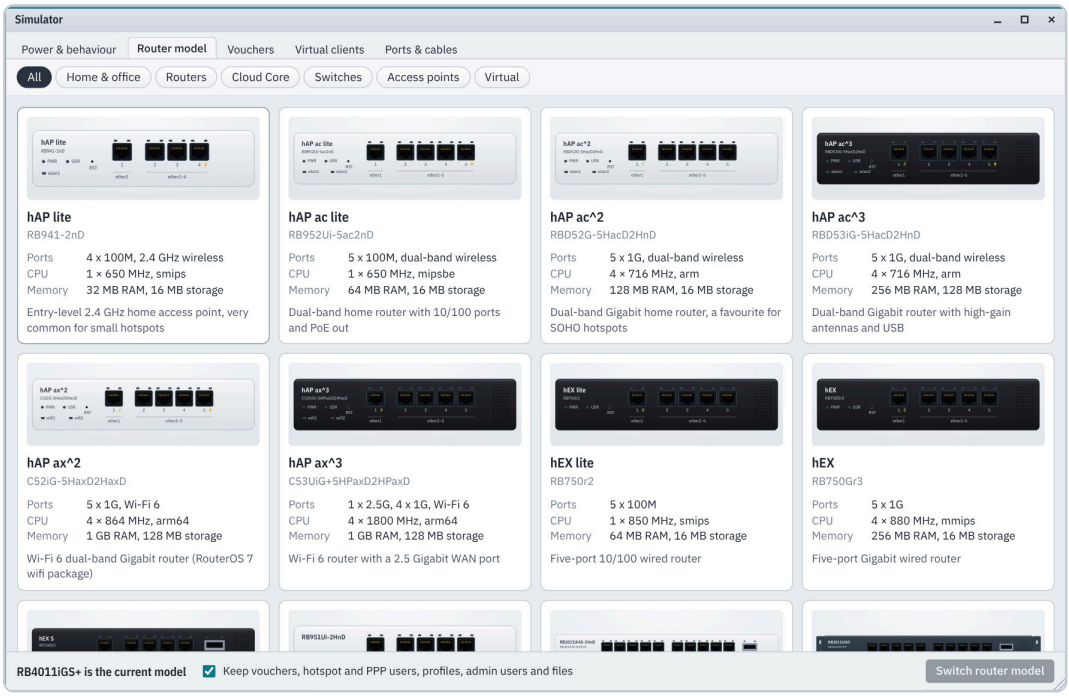


Figure 4.3 Simulator › Router model: the model gallery with front-panel previews.

Family	Models (key)
Home and office	hap-lite, hap-ac-lite, hap-ac2, hap-ac3, hap-ax2, hap-ax3 (Wi-Fi 6), hex-lite, hex, hex-s, rb951
Routers	rb2011, rb3011, rb4011 (default), rb5009, l009
Cloud Core	ccr1009, ccr1036, ccr2004, ccr2116
Switches	crs326, crs328
Access points	cap-ac, wap-ac
Virtual	chr

To switch, pick a card and press **Switch router model**. The router reboots with the new hardware and its default configuration. With **Keep vouchers, hotspot and PPP users...** ticked, your vouchers, hotspot and PPP users, profiles, admin users, scripts and files are kept, so your app's data survives the switch.

You can also switch from the command line or the API with `/sim change-model model=hap-ax2` (add `keep-users=no` for a clean reset), or start TikLab with `-model hap-ac2`.

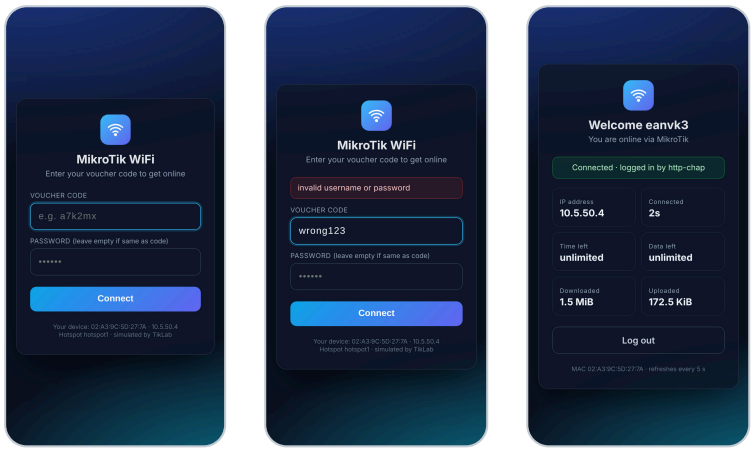
5

Hotspot and vouchers

The simulated router runs a real captive portal. Phones and laptops on your network can open it and log in with vouchers, and your app sees the sessions.

The login page

Every simulated router with a hotspot serves a login page, just like a real one. Customers see it before they are allowed online.



Login | Wrong voucher | Logged in

Figure 5.1 The hotspot login page on a phone: asking for a voucher, refusing a wrong code, and the status page after logging in.

Open `http://PC-address/login` on a phone or laptop on the same network, or click **Hotspot login page** at the top of the console. The pages come from the router's files (`hotspot/login.html` and the others), with MikroTik's template variables and both CHAP and PAP login, so your own login page design works as it would on a router (chapter 10).

End-to-end voucher test

To check a whole voucher flow, from your app to a phone and back:

1. Create vouchers in your app, or with **Simulator › Vouchers › Generate**.
2. On a phone on the same Wi-Fi as the PC, open `http://PC-address/login` and enter a voucher code.
3. The session appears in your app and under **IP › Hotspot › Active**. Uptime, bytes and the time left count down live.
4. When the voucher's uptime or data limit runs out, the user is logged out with the same messages RouterOS uses.

You can also log a device in from the API (`/ip/hotspot/active/login user=… password=… ip=… mac-address=…`) or by clicking a device on the Dashboard's network map. Rate limits from hotspot user profiles apply: a user on a 2M/2M profile never goes faster than 2 Mbps, and its dynamic queue appears under Queues.

6

Connecting apps and TapTap

Any app that manages MikroTik routers can manage TikLab. Use the PC's address, the user admin and the router password.

Services and ports

The simulator offers the same services as a MikroTik router, on the same ports:

Service	Port	Use
RouterOS API	**8728**	What most MikroTik apps use: vouchers, PPPoE, monitoring
RouterOS API-SSL	8729	The API over TLS, with a self-signed certificate
REST API	80	`http://PC-address/rest/…`, RouterOS 7 REST
REST over HTTPS	443	Self-signed certificate
Console	8291	`http://PC-address:8291/` in any browser
Hotspot login page	80	`http://PC-address/login`
SSH	22	RouterOS command line, `ssh admin@PC-address`
Telnet	23	RouterOS command line
FTP	21	Files and hotspot pages (FileZilla, WinSCP, Windows ftp)

On Linux without root, ports 80, 443, 22, 23 and 21 move to 8080, 8443, 2222, 2323 and 2121. Ports can be changed as on a real router (`/ip service set api port=8730`) or with command-line options (chapter 12).

TapTap over the API

TapTap installed on your own network can manage TikLab directly: add a router with the PC's address, port 8728, user admin and the router password. Everything TapTap reads and changes over the API has been checked against TapTap's own code: vouchers, profiles, active sessions, IP bindings, interfaces, live traffic, WAN analysis and the full configuration snapshot.

TapTap Link and Tunnel

TikLab joins TapTap the same way a real MikroTik does, through TapTap's own install scripts, so nothing special is needed on the TapTap side.

1. In TapTap, add a router and choose **TapTap Link**. Copy the install block TapTap shows.
2. In TikLab, open **TapTap & Tunnels**, paste the block and press **Run on the simulated router**. (Pasting it into a terminal works too.)
3. The simulated router installs the taptap-link script and scheduler and checks in with TapTap every few seconds over HTTPS. TapTap's page turns green; vouchers, sessions, inventory syncs, backups and the other Link commands run on the simulator as on a router.

4 **TapTap Tunnel** (RouterOS 7 models): when the tunnel is enabled in TapTap, its block creates the `taptap-wg` WireGuard interface and an API user. TikLab brings the WireGuard tunnel up for real, so TapTap's servers reach the simulator's API through it from the internet, with no port forwarding on your network.

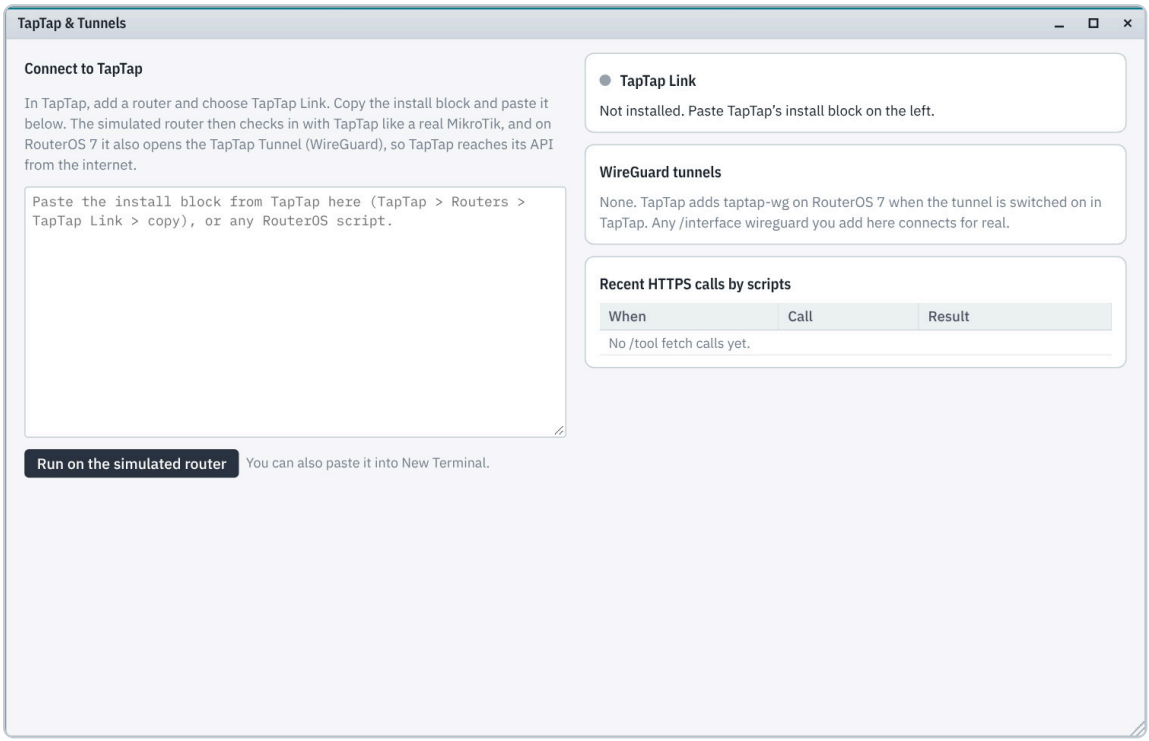


Figure 6.1 TapTap & Tunnels before Link is installed: paste TapTap's block on the left; the status of Link, WireGuard tunnels and the last HTTPS calls appear on the right.

Once Link is installed, the window shows whether it is checking in (and why not, when it fails), the tunnel's handshake and traffic, TapTap's API logins through the tunnel, and the last HTTPS calls the router made.

7

Wi-Fi on this PC

When the simulated router has Wi-Fi, your PC's own Wi-Fi adapter can broadcast the router's network, so real phones can join it and log in to the simulated hotspot.

Broadcasting the router's Wi-Fi

To let real phones join the simulated hotspot:

1. Use a model with Wi-Fi (hAP, cAP or wAP), or configure a Wi-Fi router in the Network Lab.
2. Open **Wi-Fi on this PC** in the menu (or the button on the Dashboard) and switch it on.
3. Phones scan the first QR code to join the network and the second to open the login page. Phones that check for a captive portal (Android, iPhone) open the login page by themselves.

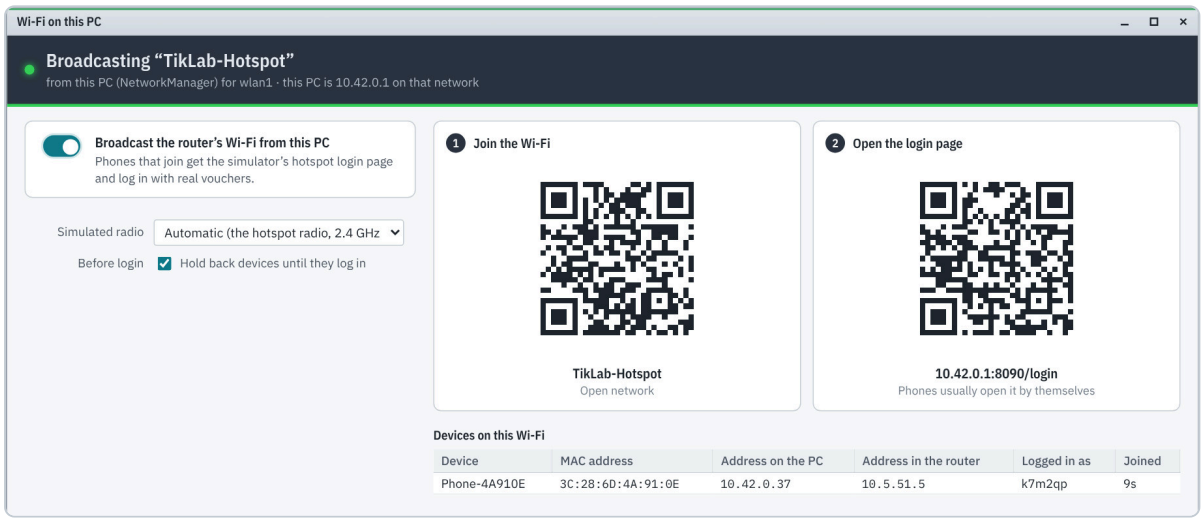


Figure 7.1 Wi-Fi on this PC, broadcasting the hAP ac² "TikLab-Hotspot" network. One phone has joined and logged in with a voucher.

- The PC broadcasts the SSID and password of the simulated radio, by default the one serving the hotspot, 2.4 GHz first. It follows the simulator: change the SSID, disable the radio or switch the router off, and the PC's Wi-Fi follows.
- Phones that join appear in the simulator with their **real MAC address**: in the registration table, DHCP leases, hotspot hosts and active sessions, so your app sees them.
- Customers on this Wi-Fi never get the router console, only the login pages.
- Only one router at a time can broadcast from a PC.

Windows and Linux

How the PC broadcasts depends on its operating system.

	Windows 10 and 11	Linux
How	Mobile Hotspot (needs a network connection to share)	NetworkManager (nmcli); the adapter must support AP mode
Open network without a password	Not possible: Windows needs a password, shown in the window	Yes
Phones held back until they log in	No: they get the login page but can browse before logging in	Yes, when TikLab runs as root
Afterwards	Mobile Hotspot gets back its previous name and password	The TikLab-Hotspot connection is removed

Tip On Windows, if phones join but the login page does not open, Windows Firewall is blocking TikLab on the hotspot network. The window shows the one command that allows it; run it as administrator.

8 The Network Lab

Build a whole network out of several MikroTik routers, switches, the internet and end devices, wire it up with cables, and configure each router from the same console.

The workbench

Open **Network Lab** in the menu. On first use the canvas holds your main router and the Internet, connected by the cable in its ether1. Every router you add is a complete simulated router, with its own configuration, files, users and services, running inside the same TikLab.



Figure 8.1 The Network Lab window with the "Hotspot towers" and "Office" sample networks around the main router.

1. **Parts.** Drag routers, switches, the Internet and devices onto the canvas. Choose the router model below the router tile.
2. **Toolbar.** Fit, zoom, the colour for new cables, and whether to show addresses and traffic.
3. **Sample networks.** Builds a home, an office, hotspot towers or a chain of routers.
4. **Main router.** The router this TikLab started with, here open in the console.
5. **Lab router.** A second full router with its own configuration. Its colour bar matches the router chip.
6. **Switch.** Everything plugged into it shares one network.
7. **Device status.** The dot is green when online, amber at a hotspot login page, blue on the network without internet.
8. **Traffic.** Dots flow amber toward the devices and blue toward the internet; dotted lines are Wi-Fi.
9. **Inspector.** Details and actions for the selection, or an overview of the network.

Adding routers and devices

Everything comes from the parts list on the left of the workbench.

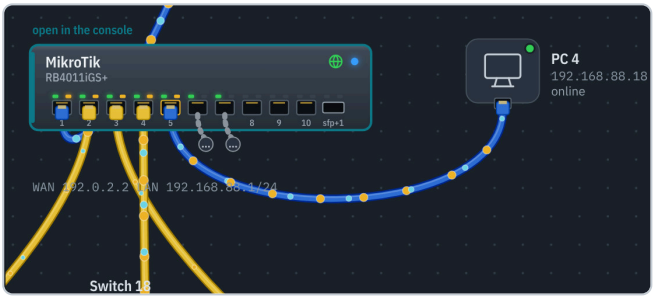
- Drag a tile from the left onto the canvas. You can also select a tile and press `Enter` to add it in the middle of the view.
- For a router, pick the model first in the list under **MikroTik router**. A new router boots in a second or two.
- **Give it its own LAN subnet** (ticked by default) moves a new router's LAN off the factory 192.168.88.0/24, to 192.168.89.0/24, 192.168.90.0/24 and so on, so routers plugged into each other do not clash.
- Devices are a PC, laptop, phone, tablet, smart TV, printer, camera, smart plug and server. Each gets a realistic MAC address.
- Drag a device to move it. Scroll to zoom, drag the empty canvas to pan, and press `F` to fit everything in view.

Running cables

Cables connect ports. Drag from one port to another and the cable follows the pointer, hanging like a real patch cable.



Dragging from the new PC: the free ports light up and the cable snaps to ether5.



Plugged in: the PC gets 192.168.88.18 from the router and traffic starts to flow.

Figure 8.2 Connecting a PC to the main router.

- Drag from a port to another port. Ports that can take the cable light up; the others dim. The cable snaps to the nearest free port.
- Drop the cable on a device instead of a port, and it uses the first free port: ether1 for a router's uplink, a LAN port otherwise.
- **Move a cable** by dragging its plug out of a port and dropping it on another port. **Unplug** it by dropping the plug on empty canvas, or select the cable and press `Delete`.
- New cables take the colour chosen in the toolbar. Select a cable to change its colour.
- Phones, tablets and laptops join a router over **Wi-Fi**: drag from their Wi-Fi symbol to one of the router's antennas.

What the cables do

The cables are not drawings: they decide what happens on every router.

- A port is up only when its cable leads to something that is switched on.
- A router plugged into another router gets its WAN address from that router's DHCP server. It appears in that router's leases, and both routers see each other in `/ip neighbor`.

- A router plugged into the Internet gets an address from your PC's network, like a modem.
- PCs and phones become clients of the router serving their network: they get a lease, appear in the Wi-Fi registration table, and wait at the hotspot login page if there is a hotspot.
- Internet access follows the path. Pull an uplink, power off the router in front, or put a router behind a hotspot it has not logged in to, and everything behind it goes offline, including `/tool fetch` and its TapTap Link.
- The dots on the cables are the simulated traffic. Port LEDs blink with it, and a WireGuard tunnel to TapTap shows as a violet arc to the internet.

The inspector

The panel on the right shows details and actions for whatever you select: a router, a device or a cable.

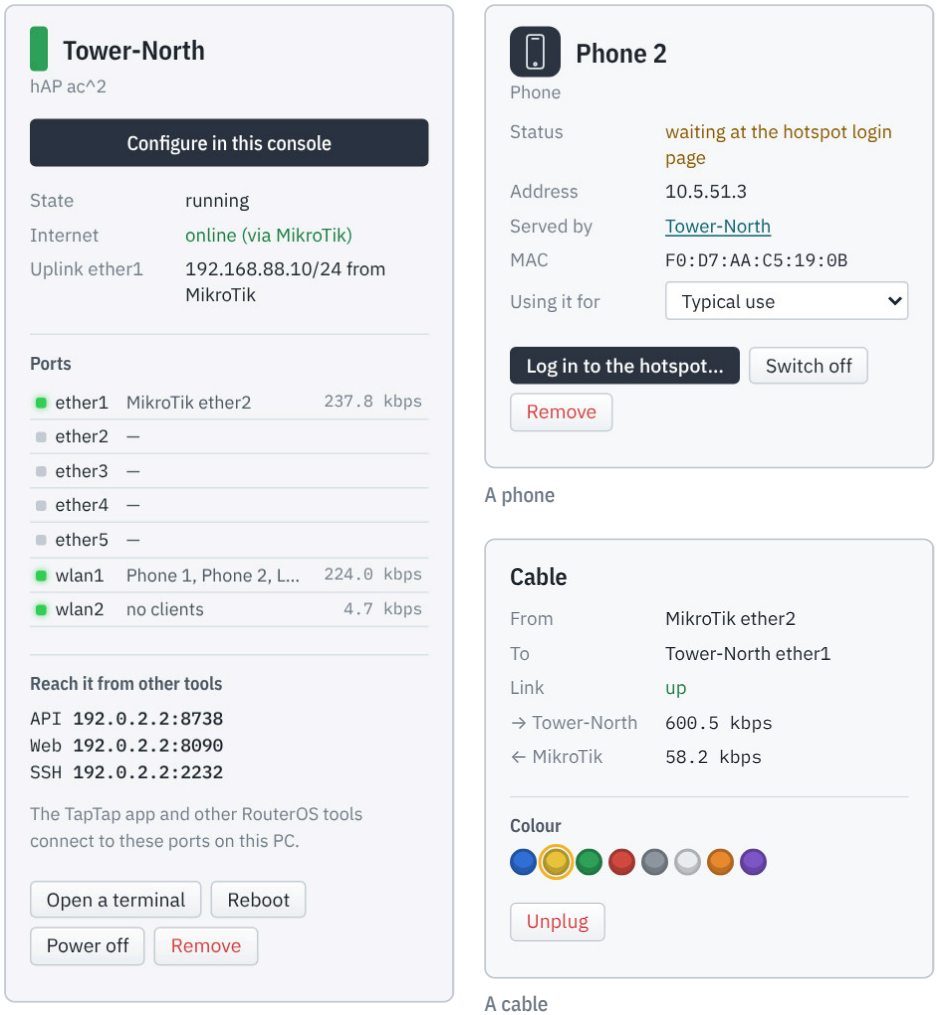


Figure 8.3 The inspector for a router, a device and a cable.

- **Routers: Configure in this console**, internet status and uplink, every port with what is plugged into it and its traffic, and the ports other tools use to reach this router. Below are **Open a terminal**, **Reboot**, **Power off** and **Remove**.
- **Devices:** status, address, the router serving it, MAC address, what it is used for (typical use, idle, browsing, streaming, downloading a lot), **Log in to the hotspot...**, **Switch off** and **Remove**.

- **Cables:** both ends, link state, traffic in each direction, colour and **Unplug**.
- Click a name at the top to rename it. Renaming a router also changes its identity.
- With nothing selected, the inspector shows how many routers, devices and connections there are, how many devices reach the internet, and the list of routers.

Logging a device in to a hotspot

A device behind a hotspot waits at the login page, with an amber dot. Double-click it, or select it and choose **Log in to the hotspot…**. Pick a voucher from that router; the password fills in by itself. The session then appears on that router under IP › Hotspot › Active, and in your app.

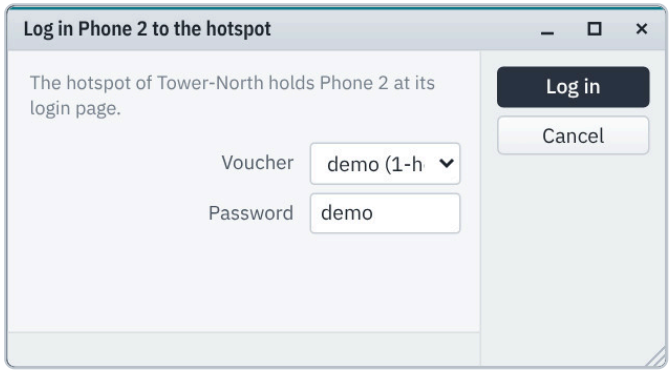


Figure 8.4 Logging Phone 2 in to the hotspot of Tower-North with a voucher.

Configuring several routers

The console configures one router at a time, and you choose which. Every router keeps its own set of windows.

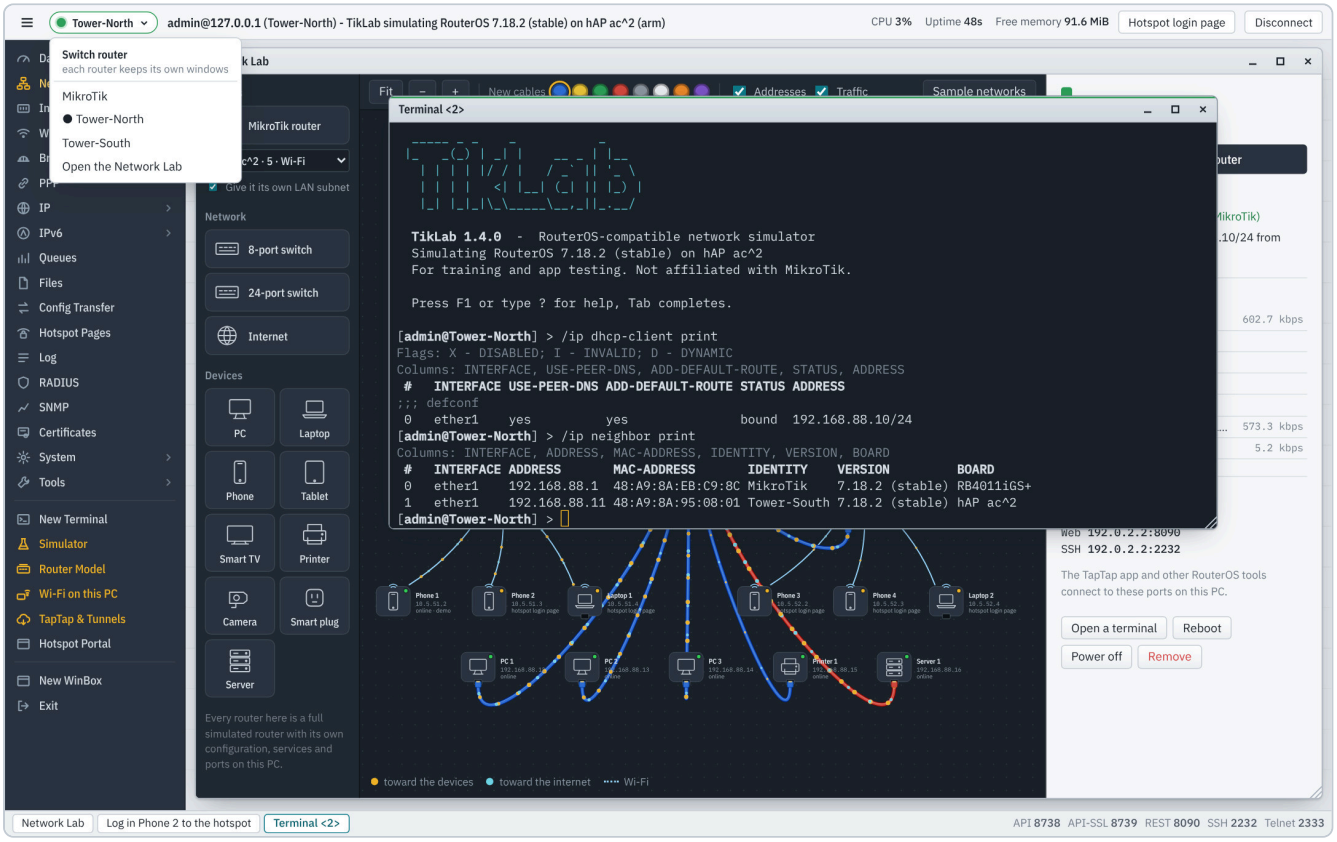


Figure 8.5 The console switched to Tower-North: the router chip shows its name, its terminal shows the DHCP lease from the main router and both neighbours, and the chip's menu lists the routers.

- **Double-click a router**, or choose **Configure in this console**, and the whole console switches to it: menus, Dashboard, terminals and every tool.
- The **router chip** at the top left shows which router you are configuring. Click it to switch to another router or to open the Network Lab.
- Each router keeps its own windows. Switch away and back, and its terminal is still there and still connected.
- The colour of the active window's title bar matches the router's colour on the canvas.

> **Note** A lab router that you gave a password asks for it the first time you switch to it.

Sample networks

Sample networks builds a small network around your main router in one click:

Sample	What it adds
Home	A phone, laptop and tablet on the router's Wi-Fi, a smart TV and a PC on cables (needs a router with Wi-Fi)
Office	An 8-port switch with three PCs, a printer and a server
Hotspot towers	Two hAP ac² towers with their own hotspots, each with two phones and a laptop waiting at the login page
Chain	A router behind a router behind your main router, each with a PC

Reaching lab routers from other tools

Each lab router has its own block of ports on your PC. Router *N* adds 10 × *N* to every port. The inspector lists the exact ports for each router.

Router	API	Console and REST	SSH	Telnet
Main router	8728	8291 / 80	22	23
Lab router 1	8738	8301 / 8090	2232	2333
Lab router 2	8748	8311 / 8100	2242	2343

Point the TapTap app or any RouterOS tool at these ports to manage a whole fleet. Lab routers also announce themselves with MNDP, so they appear in WinBox's and TikLab's neighbour lists.

The lab is saved in `tiklab-data/lab.json` and each router's configuration in `tiklab-data/lab/<id>/`. Removing a router from the canvas deletes its folder.

9

Moving configurations

Build a configuration in TikLab and run it on a real MikroTik, or bring a real router's configuration into TikLab. Open Config Transfer in the menu (also under System › Export / Import).

Export to a real router

The first tab writes a RouterOS script that rebuilds the simulator's configuration on a real router.

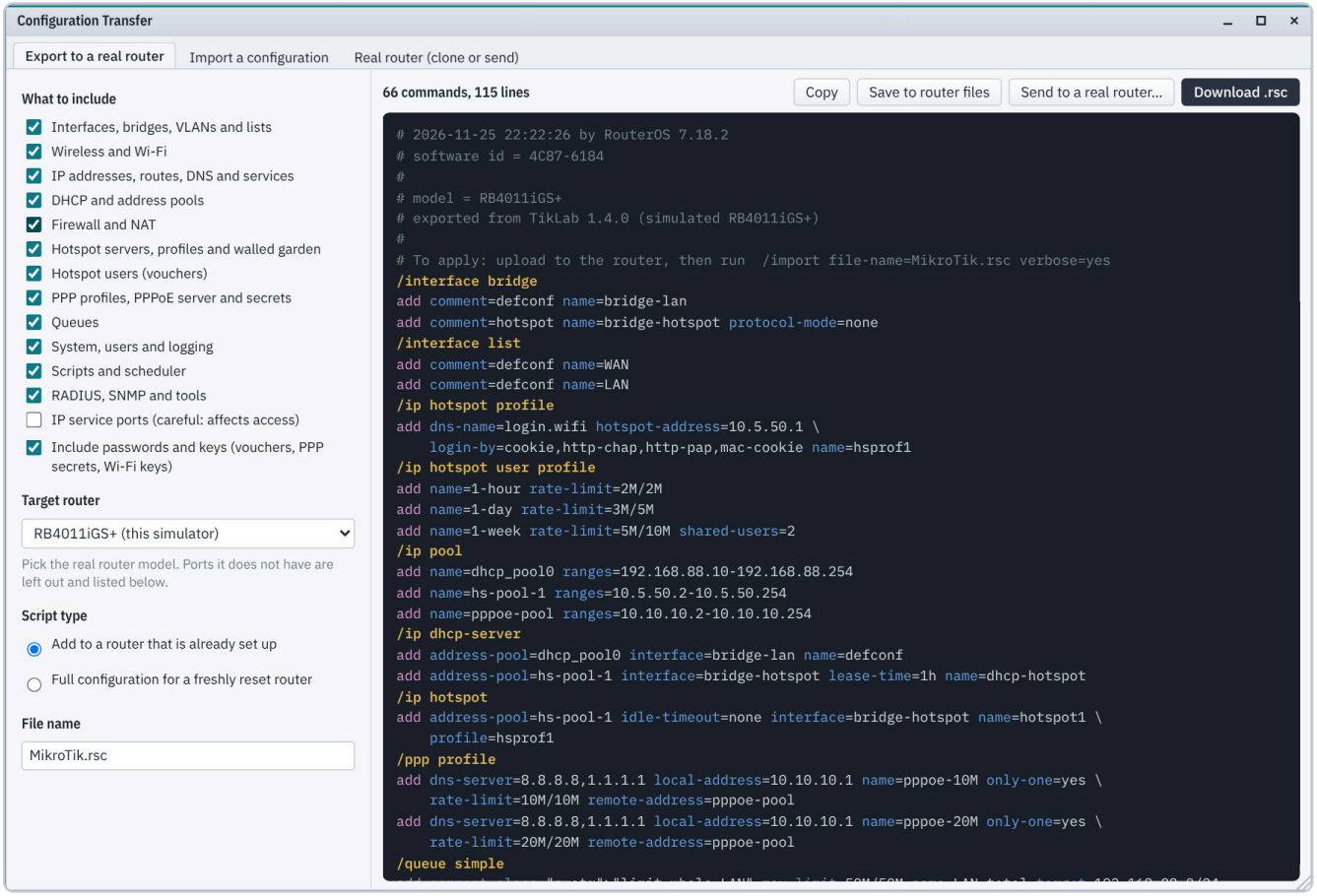


Figure 9.1 Config Transfer › Export to a real router, with the generated script on the right.

1. Tick the parts you want: interfaces, hotspot, vouchers, PPP, firewall and so on.
2. Choose the **target router** model. If it has fewer ports than the simulator, entries for the missing ports are left out and listed. Wi-Fi interfaces are translated between wlanN and wifiN.
3. Choose the **script type**: **Add to a router that is already set up** (upload the file and run `/import file-name=NAME.rsc verbose=yes`) or **Full configuration for a freshly reset router** (run `/system reset-configuration no-defaults=yes skip-backup=yes run-after-reset=NAME.rsc`; the script waits for the interfaces).
4. Press **Download .rsc**, then drag the file into WinBox › Files or upload it with FTP. Or use **Send to a real router**.

- Voucher, PPP and Wi-Fi passwords are included when **Include passwords and keys** is ticked. Router login passwords (`/user`) are never exported.
- IP service ports are not exported by default, because the simulator may use fallback ports such as 8080.
- The output is standard RouterOS script. The command line does the same with `/export file=name show-sensitive`.

Import a configuration

The second tab loads a real router's export into the simulator, so you can practise on a copy of a customer's router.

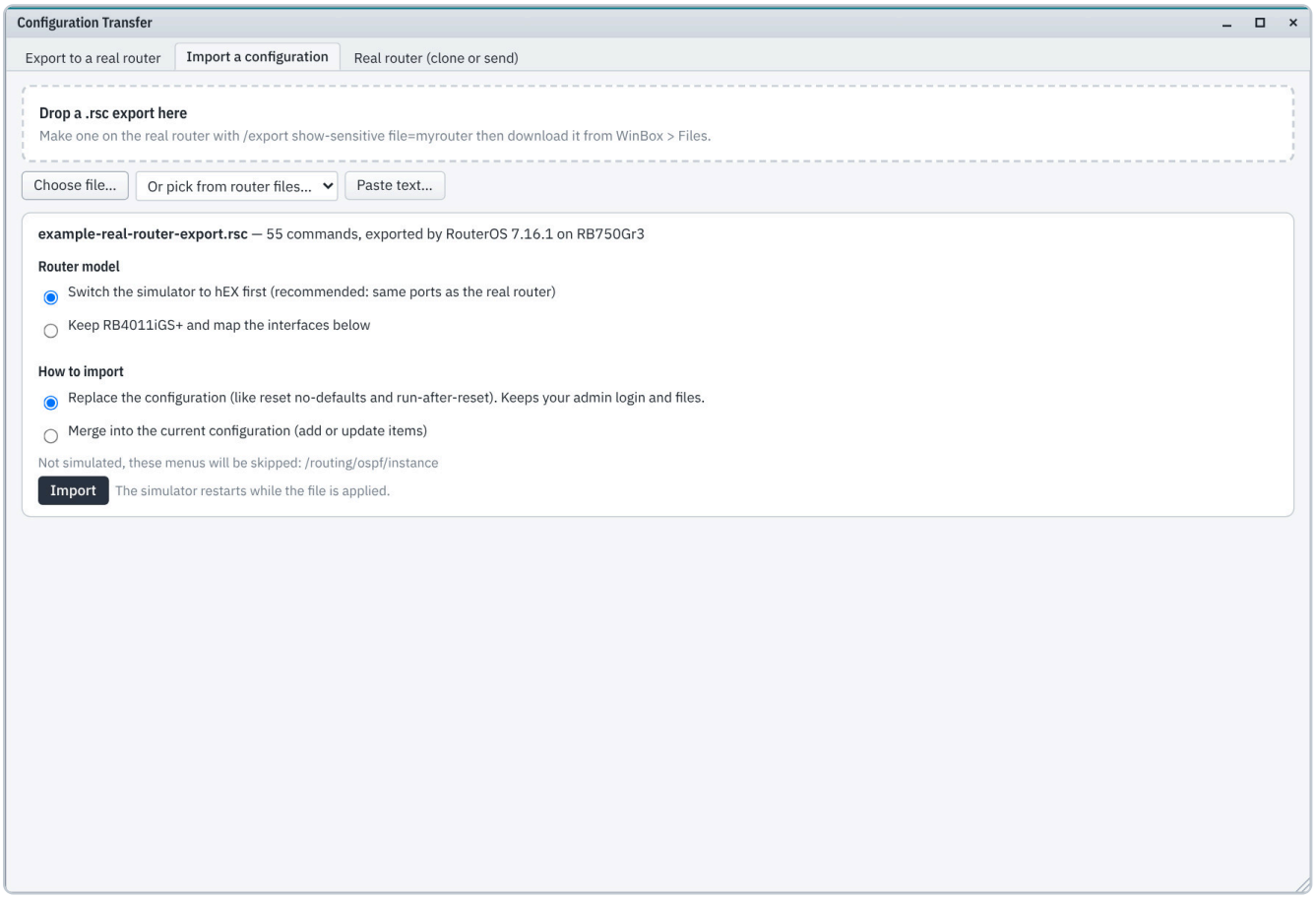


Figure 9.2 Config Transfer › Import a configuration, after choosing a real router's export.

1. On the real router run `/export show-sensitive file=myrouter` and download myrouter.rsc from WinBox › Files.
2. Drop the file on the **Import a configuration** tab, or use **Choose file...**, **Paste text...** or a file already on the router.
3. TikLab reads the `# model =` line and offers to switch the simulator to the same model (recommended). It lists interfaces the simulator lacks, so you can map them or skip the lines that use them.
4. Choose **Replace** (like a no-defaults reset with run-after-reset) or **Merge**, then press **Import**.

You get a line-by-line report: ok, warnings, errors and skipped lines. Menus the simulator does not have (OSPF, BGP, IPsec, CAPsMAN, containers) are skipped and listed; unknown parameters are dropped with a warning. The simulator keeps its own

service ports and never disables its console. RouterOS .backup files are encrypted and cannot be read; use the .rsc export.

Clone or send directly

The third tab talks to a real router over the network: it reads it, copies it into the simulator, or sends a configuration to it.

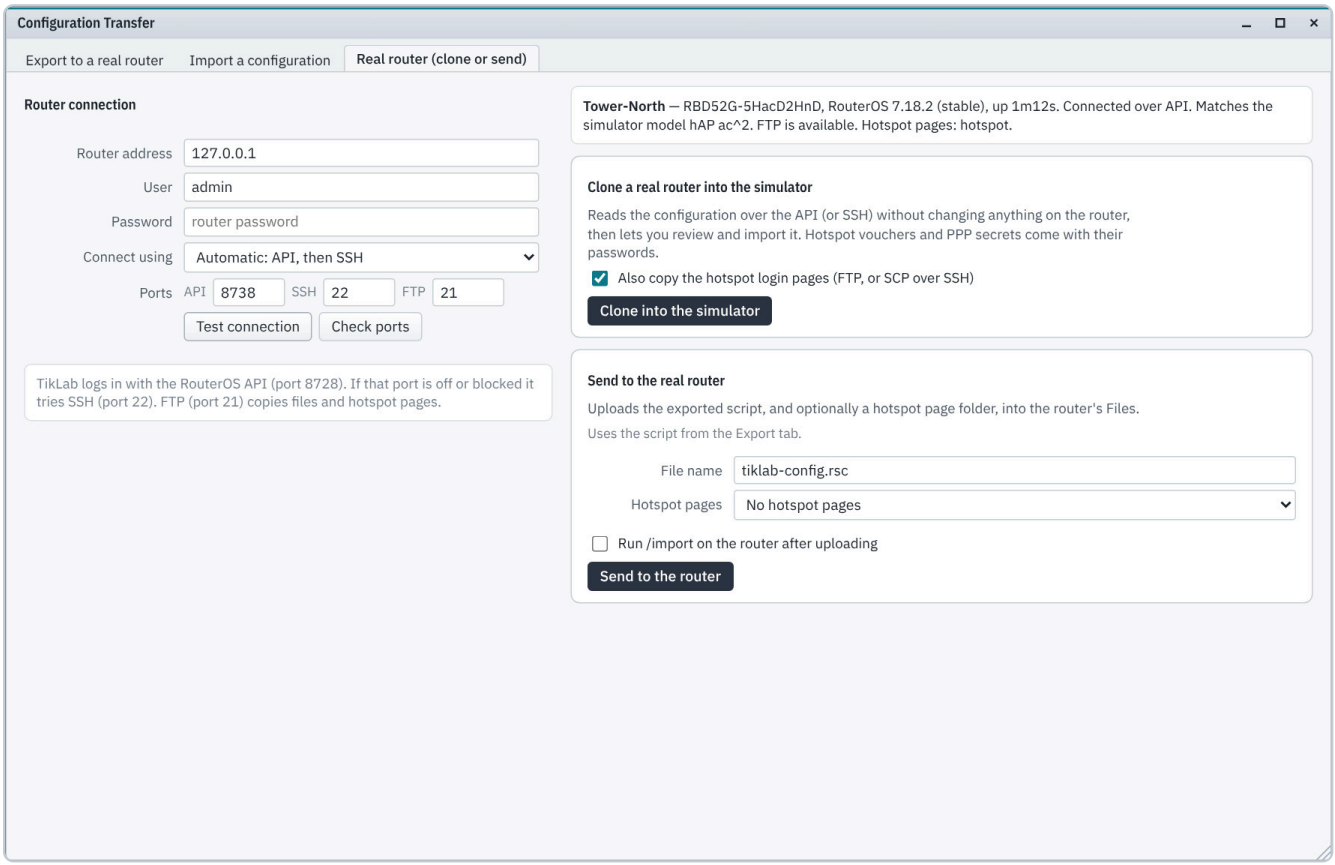


Figure 9.3 Config Transfer › Real router: a successful connection test. TikLab recognised the router as a hAP ac² over the API.

- **Test connection** logs in and shows the identity, model, RouterOS version and uptime, and whether TikLab has a matching model.
- **Clone into the simulator** reads a real router without changing anything on it, over its API (port 8728), or over SSH (port 22) when the API is off or blocked. Vouchers and PPP secrets come with their passwords, and the hotspot login pages are copied over FTP or SCP. You then review and import the result as above.
- **Send to the real router** uploads the exported script and, if you want, a hotspot page folder, into the router's Files over FTP, or SCP when only SSH is open. Running `/import` afterwards is optional, off by default, and asks for confirmation.
- **Connect using** chooses Automatic (API, then SSH), API only or SSH only.

When TikLab cannot reach a router

If a connection fails, TikLab checks the router's ports (API, API-SSL, SSH, FTP, WinBox, WebFig, Telnet), looks the address up in this PC's ARP table and in MikroTik discovery, and explains the problem with RouterOS commands you can copy. **Check ports** runs the same check without logging in.

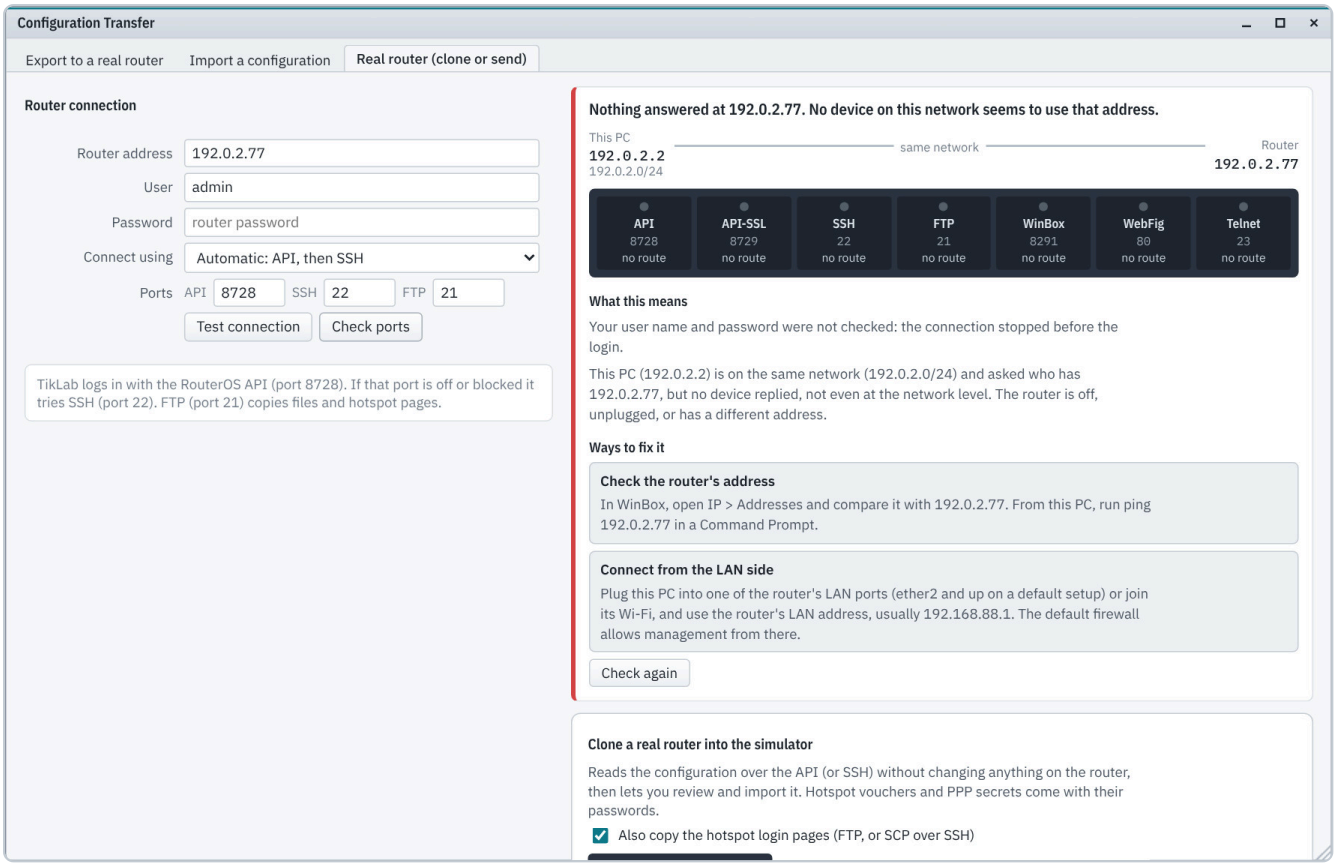


Figure 9.4 Check ports on an address where nothing answers: TikLab explains what is in the way and how to fix it.

What you see	What it means
no reply or **i/o timeout** on every port	The router's firewall drops the traffic. The default MikroTik firewall only allows management from the LAN ports, so this happens when the PC is on the network of the router's ether1. Plug the PC into a LAN port and use 192.168.88.1, or let the PC through (below).
no reply on the API port only	A firewall rule allows only some ports, or the API service is limited to certain addresses. TikLab switches to SSH if SSH answers.
refused on the API port	The API service is off or on another port: `/ip service enable api`.
invalid user name or password	The router answered and rejected the login. Names and passwords are case-sensitive; check the user's Allowed Address.

Your user name and password are only checked after the connection is made, so a timeout says nothing about them. To let one PC through the firewall (TikLab fills in your PC's address):

```
/ip firewall filter add chain=input action=accept protocol=tcp \
    src-address=192.168.1.50 dst-port=8728,8729,21,22 \
    comment="TikLab access" \
    place-before=[:pick [find chain=input dynamic=no] 0]
```

Remove it when you are done: `/ip firewall filter remove [find comment="TikLab access"]`

10

Files and hotspot pages

The router’s files, and a studio for designing and checking hotspot login pages before they go on a real router.

Files

Files shows the router’s file system, like WinBox’s Files window.

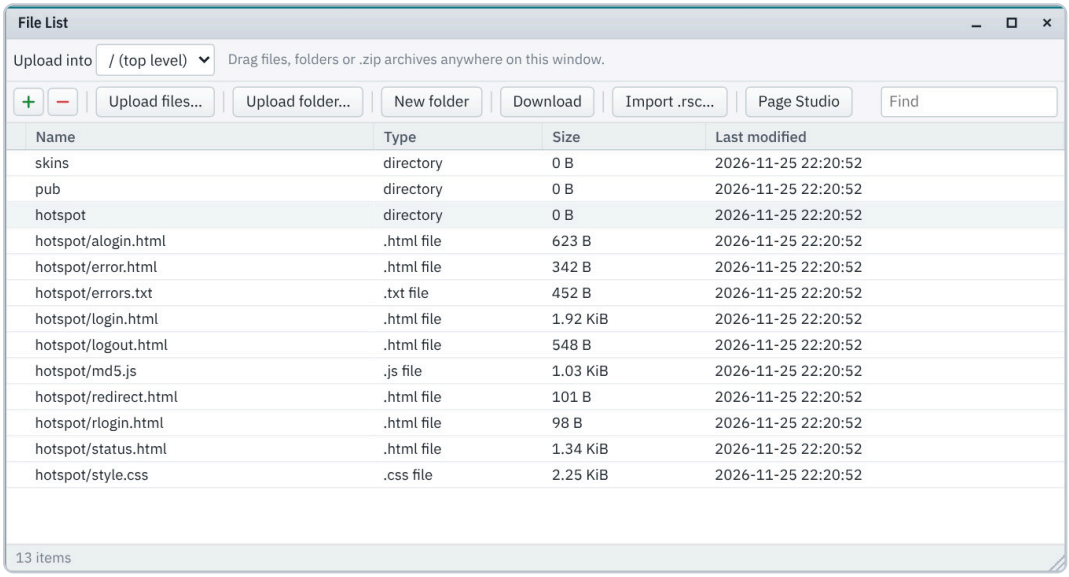


Figure 10.1 The router’s files, with the default hotspot pages.

- Upload files, whole folders or .zip archives: drag them onto the window or use the buttons.
- Create folders, edit text files, and download any file, or a whole folder as .zip.
- The FTP server gives FileZilla, WinSCP and `/tool fetch` the same access a real router gives.

Hotspot Page Studio

Hotspot Pages is a studio for login page designs. It shows them as a phone would and checks them before they go on a router.

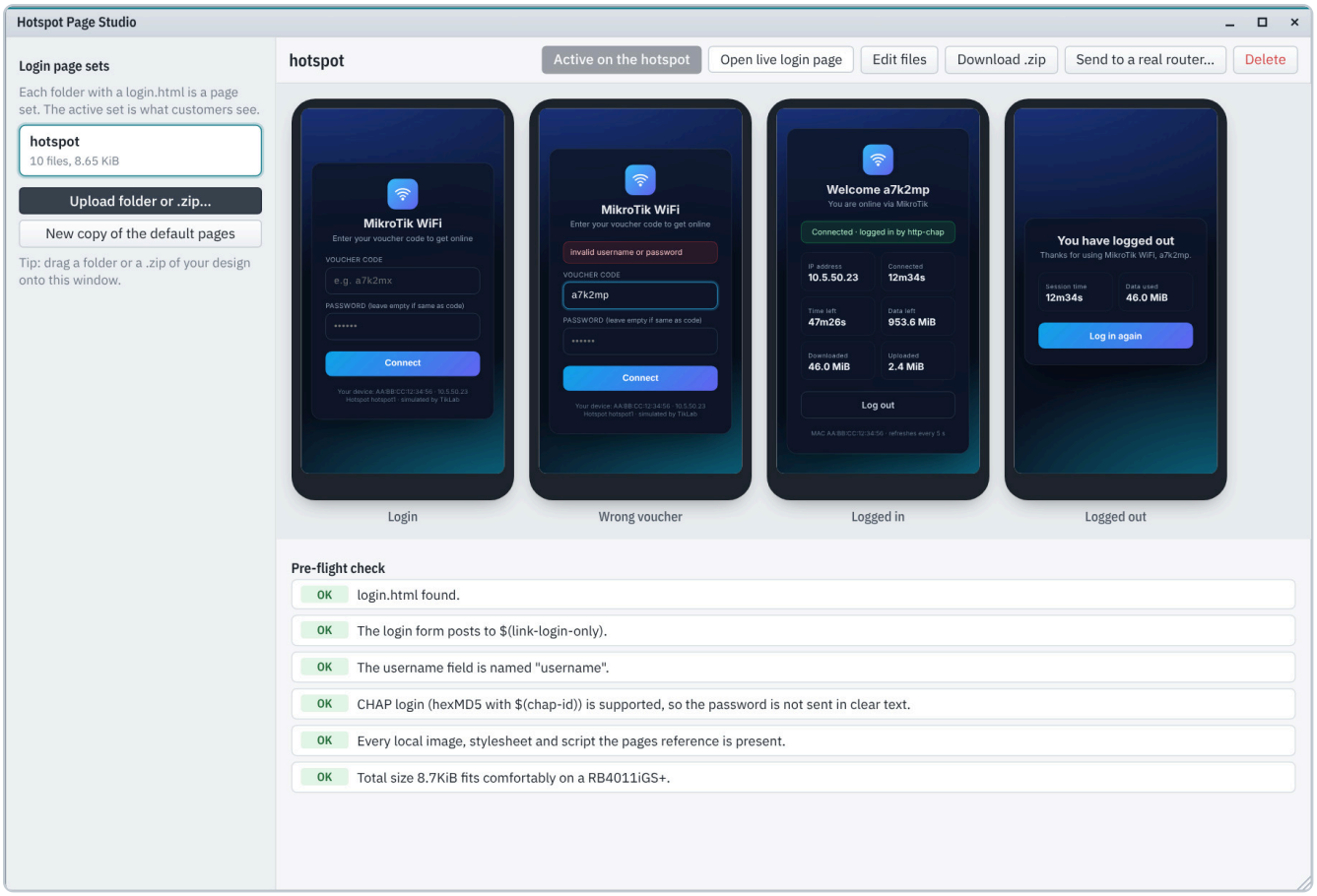


Figure 10.2 Hotspot Pages: four phone previews of the active login page and the pre-flight check.

1. Install a login page design from a folder or a .zip, or start from a copy of the default pages.
2. Look at it in **four phone previews**: login, wrong voucher, logged in and logged out.
3. Read the **pre-flight check**. It catches missing files, broken $(if) blocks, jQuery $() code that MikroTik would mangle, forms that do not post to $(link-login-only), PAP and CHAP mismatches, and external hosts that need the walled garden (one click adds them). It also warns when the pages will not fit the flash of the selected model.
4. **Active on the hotspot** makes it the live page; test it from a phone at `http://PC-address/login`.
5. Download it as .zip, or send it straight to a real router.

Pages are served exactly as a real router serves them: from the hotspot html-directory, with relative links (img/logo.png, css/style.css) and MikroTik variables.

11

Scripts and WireGuard

The terminal and /system script run the RouterOS scripting language, and WireGuard interfaces really connect.

RouterOS scripting

Scripts work in the terminal, in `/system script`, in the scheduler and over SSH. Supported:

- variables (`:local`, `:global`), `:if`, `:while`, `:for`, `:foreach`, `:do … on-error`, `:onerror`;
- functions (`:local f do={…}`), `:return`, `[:parse]`, `:execute` and scheduler jobs with `:jobname`;
- arrays and `->`, `:pick`, `:find`, `:len`, `:typeof`, the `:to…` conversions, `:convert`, `:serialize` and `:deserialize` (JSON), `:resolve`;
- `where` clauses, `print as-value` and `get` without a property.

A small example that lists the users online on the hotspot:

```
:foreach a in=[/ip hotspot active find] do={
    :put ("online: " . [/ip hotspot active get $a user])
}
```

`/tool fetch` makes real HTTP and HTTPS requests. Certificates imported with `/certificate import … trusted=yes` are trusted, and so is the PC's own trust store. A router that is offline (internet switched off, or cut off in the Network Lab) cannot fetch.

WireGuard

`/interface wireguard` is real WireGuard, in userspace: no driver or administrator rights needed. A simulated router can join a WireGuard network (TapTap Tunnel, your own VPN server), and WireGuard clients can connect to it: give the interface a listen-port, add the client as a peer, and the simulator answers ping and its services (API, SSH, console, Telnet) on its tunnel address.

> **Note** Inside tunnels only IPv4 is supported, and FTP data connections do not pass: use SCP or the API to move files.

12

Reference

Command-line options, separate instances and where TikLab keeps its data.

Command-line options

Options go after the program name, for example `tiklab -model hap-ac2 -no-browser`.

Option	Meaning
`-model KEY`	Router model, for example hap-ac2, hap-ax2, hex-s, rb5009, ccr1036
`-list-models`	Print all models with their ports and exit
`-password SECRET`	Admin password for a fresh configuration (empty by default)
`-identity NAME`	Identity for a fresh configuration
`-instance N`	Run several independent copies side by side
`-reset`	Reset the configuration to factory defaults on start
`-no-browser`	Do not open the console in the browser
`-bind IP`	Listen on one address only (all addresses by default)
`-api-port`, `-api-ssl-port`, `-www-port`, `-www-ssl-port`, `-winbox-port`, `-ssh-port`, `-telnet-port`	Choose the ports
`-ros-version "7.18.2 (stable)"`	The RouterOS version reported to apps
`-no-discovery`	Turn off MNDP discovery
`-verbose`	Print the router log in the start window

On Windows, add options by editing Start TikLab.bat, or run `tiklab.exe -model hap-ac2` from a command prompt in the TikLab folder.

Separate instances

Besides the Network Lab, you can run independent copies of TikLab side by side:

```
tiklab -instance 1 -model hap-ac2 -no-browser
tiklab -instance 2 -model hex-s -no-browser
```

Each instance gets its own data folder (tiklab-data-1, …), identity, MAC addresses and ports: instance *N* adds 10 × *N* (API 8738, console 8301, …). They find each other in the Neighbors list but are not wired together. Lab routers use the same port blocks, so

do not run instances on a PC whose Network Lab already has routers.

Data and resetting

Everything TikLab keeps is in the tiklab-data folder next to the program.

In tiklab-data	Holds
config.json	The main router's configuration and its files, including the hotspot pages
lab.json	The Network Lab: devices, positions and cables
lab/<id>/	The configuration and files of each lab router
ssh_host_*.pem, tiklab-cert.pem, tiklab-key.pem	SSH host keys and the TLS certificate for API-SSL and HTTPS

To reset everything, start TikLab with `-reset`, use **System › Reset Configuration**, or delete the tiklab-data folder while TikLab is stopped.

13

Troubleshooting

Common problems and what to do about them.

Problems and solutions

Start here when something does not work as expected.

Problem	What to do
The app (or TapTap) cannot connect to TikLab	Use the PC's address shown in the start window, not 127.0.0.1, when the app runs on another device. On Windows, allow TikLab in Windows Firewall on private networks. Check the API port in the status bar; another program may have taken 8728.
The browser did not open	Open http://127.0.0.1:8291/ yourself. If 8291 was busy, the start window shows the port TikLab used.
Ports 80, 22 and 23 became 8080, 2222 and 2323	That happens on Linux without root. Run with sudo, or give the program the right once with setcap (chapter 2).
Real WinBox lists TikLab but cannot log in	This is expected: WinBox needs MikroTik's own protocol. Use the console in your browser.
Clicking a port on the Dashboard says it is cabled in the Network Lab	Ports with a lab cable are plugged and unplugged on the lab canvas.
A lab router is offline	Select it: the inspector says why. Usually its uplink is unplugged, the router in front is off, or it sits behind a hotspot and needs to log in.
A device shows "no DHCP server on this network"	The network it is plugged into has no DHCP server: plug it into a router's LAN port, or into a switch connected to one.
Phones join the PC's Wi-Fi but the login page does not open	On Windows, allow TikLab in the firewall with the command shown in the Wi-Fi on this PC window, as administrator.
Wi-Fi on this PC is not available	The model must have Wi-Fi, the PC needs a Wi-Fi adapter that can host a hotspot, and on Windows a network connection to share. Only one router at a time can broadcast.
I forgot the admin password	Stop TikLab and start it with -reset (this resets the configuration), or delete config.json in tiklab-data.

14

Limitations and licences

What TikLab does not do, and the software it includes.

Limitations

TikLab simulates a router; it is not one. Keep these limits in mind.

- There is no real packet forwarding. Clients and traffic are simulated, and nothing is routed through the simulator. WireGuard tunnels reach the simulator itself, not networks behind it.
- In the Network Lab, cables carry DHCP, neighbour discovery, hotspot logins and internet reachability. VLANs, PPPoE between lab routers, routing protocols and firewall rules between routers are not simulated, and ping between lab routers does not cross the cables.
- One PC has one Wi-Fi hotspot: only one router at a time can broadcast its network from the PC.
- Inside WireGuard tunnels: IPv4 only, and FTP data connections do not pass.
- Wi-Fi on this PC was tested with a stand-in network on Linux. The Windows Mobile Hotspot part and real adapters still need trying on your PCs. macOS is not supported for it.
- Routing protocols (OSPF, BGP), IPsec, CAPsMAN, containers and the WinBox binary protocol are not emulated. Their lines are skipped, and listed, when you import a real router's configuration.
- RouterOS .backup files cannot be read because they are encrypted. Use .rsc exports.

Licences

TikLab is original code. It includes golang.org/x/crypto (BSD-3-Clause licence), xterm.js (MIT licence) and the IBM Plex fonts (SIL Open Font License 1.1). MikroTik, RouterOS and WinBox are trademarks of MikroTik and are mentioned only to describe compatibility.