آزمایشگاه شبکه

آزمایش ۷: راه‌اندازی سرویس مسیریابی در Mininet

الف) معرّفی بستر آزمایش

هستهٔ شبکهٔ جهانیِ اینترنت را روترهای قدرتمندی تشکیل می‌دهند که قادرند حجم بزرگی از ترافیک داده را مسیردهی نمایند. این روترها که نوعاً ساخت شرکت‌های مشهوری مثل: Cisco، Huawei یا Juniper هستند از سیستم‌عامل‌های اختصاصی (مثل: Cisco IOS یا JunOS) استفاده می‌کنند که تعامل با آنها از طریق ترمینال‌های کنترلی ویژهٔ پیکربندی شبکه صورت می‌گیرد. دستوراتی هم که برای پیکربندی این روترها استفاده می‌شود معمولاً با دستوراتی که شما در کنسول‌های UNIX یا Linux استفاده می‌کنید، تفاوت دارند.

در حالت ایده‌آل، می‌خواهیم امکان اجرای سیستم‌عاملی مثل IOS در یک محیط مجازی فراهم باشد. اگرچه این امکان به لحاظ فنی میسّر است، امّا منع قانونی دارد چرا که شرکت‌هایی نظیر Cisco یا Juniper اجازهٔ اجرای سیستم‌عامل‌هایشان را روی تجهیزاتی غیر از روترهای خودشان نمی‌دهند.

در عوض، ما از Quagga استفاده خواهیم کرد که یک بستهٔ رایگان نرم‌افزاری برای مسیریابی روی Linux است. Quagga یک ترمینال کنترلی به ما می‌دهد که در آن امکان اجرای دستوراتی است شبیه به آنچه در IOS سیسکو قابل اجراست. بعلاوه، Quagga را می‌توان از طریق MiniNExT با Mininet هم ترکیب کرد. MiniNExT یک گسترش از Mininet است که namespaceهای لازم برای اجرای مستقل Quagga روی هر ماشین مجازی را فراهم می‌آورد.

**توجه! محیط MiniNExT از قبل روی VMهایی که در اختیار دارید، نصب شده است.*

الف-۱) Quagga چیست و چگونه کار می‌کند؟

Quagga یک بستهٔ نرم‌افزاری ویژهٔ مسیریابی است که در آن چندین پروتکل مشهور (مثل: OSPF، RIP و BGP-4) برای محیط‌های UNIX پیاده‌سازی شده‌اند. Quagga از چندین پروسس تشکیل شده که می‌توانند به صورت daemon در پس‌زمینه اجرا شوند. سه مورد از daemonهای Quagga که برای انجام این آزمایش (و بعدی) اهمیت دارند، عبارتند از:

- zebra: که برای مدیریت اینترفیس‌های شبکه‌ای یک روتر استفاده می‌شود. این پروسس امکان پیکربندی اینترفیس‌ها و مانیتور کردن وضعیت آنها را می‌دهد. بعلاوه، در مقایسه با دستور route -n، یک نَمای جزئی‌تر از جداول مسیریابی را فراهم می‌آورد. به بیان دیگر، zebra یک جایگزین پیشرفته‌تر برای دستورات شبکه‌ای Linux است (یعنی: دستوراتی مثل: ifconfig، route و غیره).

- ripd: پیاده‌سازی ورژن 2.0 از پروتکل مسیریابی RIP را فراهم می‌آورد.
- quagga: سرویس اصلی که برای فراخوانی daemonهای فوق بکار می‌روند.

جلوتر، ملاحظه خواهیم کرد که MiniNExT برای هر «روتر مجازی[1]» یک namespace جداگانه ایجاد می‌کند که به این معناست که هر روتر مجازی داخل محیط مجازی Mininet، سرویسِ quaggaی خاص خود را به طور مستقل اجرا می‌نماید.

الف-۲) راه‌اندازی یک پروسهٔ Quagga

هر daemon دارای فایل کانفیگ خاص خودش است که وجود آن برای اینکه Quagga به درستی کار کند، الزامی است (ولو به صورت خالی). در VMای که در اختیار دارید، Quagga را به صورت یک سرویس Linux پیکربندی کرده‌ایم و برای راه‌اندازی آن باید دو فایل کانفیگ به نام‌های daemons و debian.conf را تنظیم نمایید.

در فایل daemons، شما محتوایی شبیه به زیر ملاحظه خواهید کرد:

```
zebra=yes
bgpd=no
ospfd=no
ospf6d=no
ripd=yes
ripngd=no
isisd=no
```

به کمک این فایل، برای Quagga مشخص می‌کنید که تمایل دارید کدامیک از پروسس‌ها را فعال‌سازی نماید. توجه کنید که پروسس zebra باید همواره فعال باشد.

در فایل debian.conf محتوایی نظیر زیر را ملاحظه خواهید نمود:

```
vtysh_enable=no
zebra_options=" --daemon -A 127.0.0.1 -u quagga -g quagga"
bgpd_options=" --daemon -A 127.0.0.1 -u quagga -g quagga"
ospfd_options=" --daemon -A 127.0.0.1 -u quagga -g quagga"
ospf6d_options="--daemon -A ::1 -u quagga -g quagga"
ripd_options=" --daemon -A 127.0.0.1 -u quagga -g quagga"
ripngd_options="--daemon -A ::1 -u quagga -g quagga"
isisd_options=" --daemon -A 127.0.0.1 -u quagga -g quagga"
```

[1] Virtual Router

در این فایل،

- daemon-- به معنای این است که می‌خواهیم پروسس مورد نظر در پس‌زمینه اجرا شود.
- A- برای مشخص کردن آدرس IP ویژهٔ دسترسی به مُد پیکربندی است که به صورت پیش‌فرض همان localhost می‌باشد.
- u- نام کاربر آغازکنندهٔ پروسس را مشخص می‌کند (که به صورت پیش‌فرض همان quagga است).
- g- گروه کاربری (privilege) را مشخص می‌کند (که به صورت پیش‌فرض همان quagga است).

پس از پیکربندی صحیح فایل conf، می‌توانیم سرویس Quagga را مشابه هر سرویس تحت Linux دیگر راه‌اندازی نماییم:

```
/etc/init.d/quagga start
```

الف-۳) خاتمهٔ یک پروسهٔ Quagga

پروسس‌های Quagga را به دو روش می‌توان خاتمه داد:

۱- **توقف کامل سرویس quagga**: با استفاده از دستور سنتی توقف یک پروسس در لینوکس:

```
/etc/init.d/quagga stop
```

۲- **خاتمهٔ انفرادی هر پروسس**: گاهاً ممکن است نیاز به شبیه‌سازی یک crash روی یک روتر باشد و باید بتوانیم که یک پروسس خاص را kill کنیم. برای kill کردن انفرادیِ یک پروسس خاص از Quagga که روی یک روتر مجازی در حال اجراست، دستور زیر را می‌توان در پنجرهٔ ترمینال تایپ نمود:

```
killall <process name 1> <process name 2> ...  <process name N>
```

که در آن، عباراتِ <process name 1> تا <process name N> همان نام پروسس‌هایی از Quagga هستند (مثل: zebra، ripd و غیره) که روی یک روتر مجازی در حال اجرا می‌باشند. توجه کنید که اگر بخواهید پروسس مورد نظر را مجدداً راه‌اندازی نمایید، ناگزیر از reset سرویس quagga هستید چراکه این تنها گزینه برای آغاز کردن پروسس‌های Quagga است.

بالأخره اینکه، همیشه می‌توانید بررسی کنید که آیا یک پروسس روی یک روتر مجازی در حال اجراست یا خیر. برای این منظور، کافی است که لیست پروسس‌های در حال اجرا را با استفاده از دستور زیر ملاحظه نمایید:

```
ps -A
```

ب) ادغام Quagga در Mininet با استفاده از MiniNExT

MiniNExT (یا همان Mininet Extended) یک لایهٔ گسترشی بر Mininet است که namespaceهای پروسس‌ها، namespaceهای ویژهٔ mount کردن filesystem و همینطور، امکان ایزوله‌سازیِ logها و runtime پروسس‌ها را فراهم می‌نماید. این محیط با هدف ایجاد شبکه‌های پیچیده‌تر از یک Mininet تنها توسعه داده شده است. محیط MiniNExT دارای یک سری کتابخانه‌های Python خاص Quagga است که ما از آنها برای ادغام Quagga در محیط مجازیِ Mininet استفاده می‌نماییم.

برای آزمایش ۷، اسکریپت‌های Python مورد نیاز را برای فهم ساده‌تر آنها، به دو بخش تقسیم کرده‌ایم:

۱- چون توپولوژی در حین آزمایش تغییر نخواهد کرد، می‌توانیم از یکی از اسکریپت‌های Python برای ایجاد یک کلاس توپولوژی استفاده کنیم. به کمک آن، خواهیم توانست روترهای مبتنی بر quagga بسازیم، host و switch و link اضافه کنیم.

۲- با اسکریپت دیگر می‌توان محیط مجازی را اجرا نمود؛ یعنی، توپولوژی مورد نظر را فراخوانی کرد و سپس، خط دستور و پنجرهٔ ترمینال برای پیکربندی روترها را اضافه نمود.

هر دو اسکریپت به صورت آماده در اختیار شما قرار داده شده است.

- **فایل‌های lab7_topo.py و lab7_network.py را باز کرده و کامنت‌های برنامه و توضیحات بخش‌های ب-(۱) و ب-(۲) در زیر را مطالعه نمایید.**

ب-۱) کلاس توپولوژی

در خصوص اسکریپت با عنوان lab7_topo.py، توضیحات زیر را مدنظر قرار دهید:

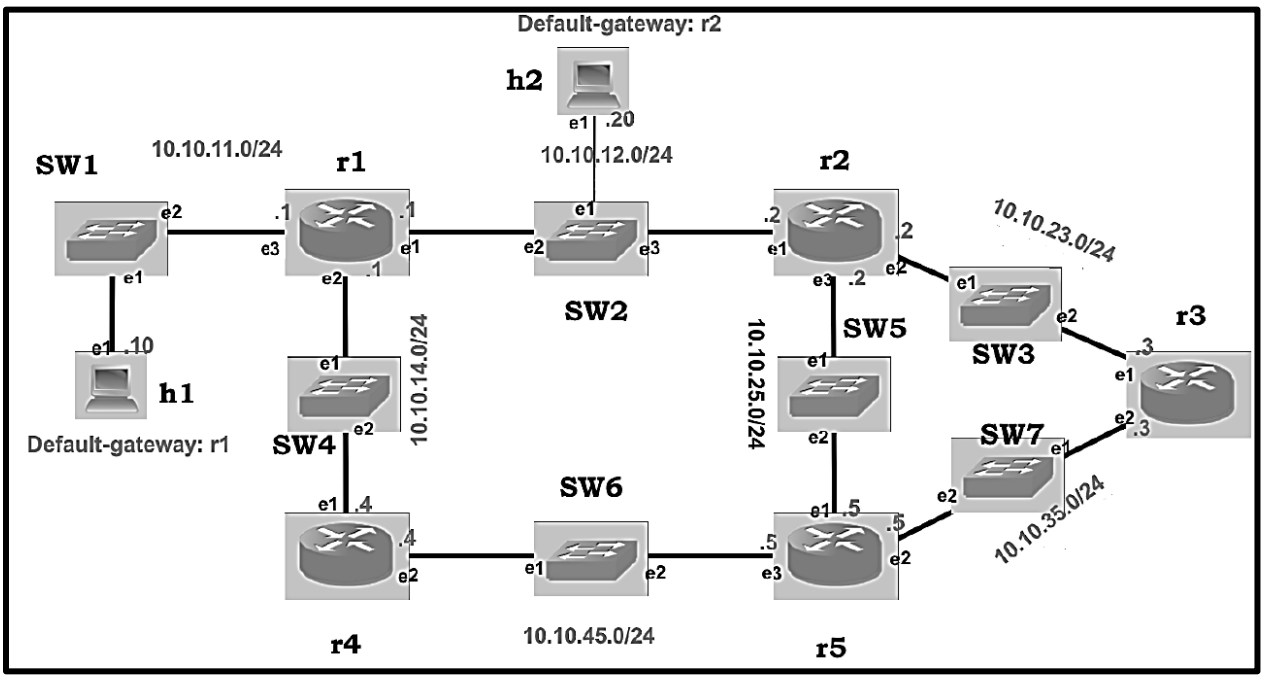
- اگر دقت نمایید، کلاس Topo را از بستهٔ MiniNExT و نه از Mininet، وارد کرده‌ایم. علت این امر آن است که کلاس فراهم شده توسط MiniNExT دارای یک تابع برای اضافه کردن یک node service است که برای عملیات Quagga ضروری می‌باشد (خط ۱۳ را ملاحظه نمایید).
- این اسکریپت فرض کرده است که شما دارای یک فولدر به نام configs در همان مسیر خود اسکریپت هستید و ضمناً داخل این فولدر هم یک فولدر به ازای هر روتر نیاز داریم که دقیقاً همنام با خود روتر است (مثلاً: r1، r2 و غیره). خطوط ۴۷ و ۷۶-۷۵ را می‌توان برای تغییر این رفتار استفاده نمود.
- با استفاده از خط ۴۱ می‌توان پروسس quagga را طوری پیکربندی کرد که به طور خودکار آغاز یا پایان یابد.

- از یک چندتایی نام‌دار (اصطلاحاً named tuple) برای مدیریت attributeهایی که تمایل داریم هنگام اضافه کردن روترهای مجازی در شبکه لحاظ کنیم، استفاده می‌شود. اگر بخواهید این attributeها را اصلاح یا زیاد و کم کنید (مثل: آدرس لوپ‌بَک برای هر روتر)، می‌توانید متغیر خط ۲۲ را اصلاح نموده و بعد به سراغ اصلاح بخشی بروید که با خط شمارهٔ ۵۶ آغاز می‌شود.
- هنگام اضافه کردن یک روتر مجازی به شبکه، باید قابلیت ایزوله‌سازیِ پروسس‌ها، logها و اجرا را فعال نماییم که این کار در خط شمارهٔ ۷۲ صورت گرفته است.

ب-۲) اسکریپت شبیه‌سازی

در خصوص اسکریپت با عنوان lab7_network.py، توضیحات زیر را مدنظر قرار دهید:

- در مستندات MiniNExT پیشنهاد شده است که ماجول isShellBuiltin را پَچ نمایید چراکه ممکن است هنگام اجرای کُد bash از ترمینال MiniNExT ایجاد اشکال نماید. این کُد در خطوط ۱۷ الی ۲۰ نشان داده شده است.
- برخلاف یک اسکریپت متداول Mininet، در MiniNExT از تابع سازندهٔ MiniNExT برای ساخت شبکه استفاده می‌شود (خطوط ۳۶ و ۵۸ را ملاحظه نمایید).
- با وجودی که می‌توانید هر فرمانی را از طریق خط دستور <mininext اجرا نمایید، راحت‌تر خواهیم بود اگر که برای هر روتر یک پنجرهٔ ترمینال جداگانه باز کنیم. کُد نوشته شده در خطوط ۶۸ تا ۷۴ این قابلیت را فراهم می‌سازد.
- تابع stopNetwork() به نحو مناسبی کلیه فایل‌های log را هنگام توقف اسکریپت حذف می‌نماید. برای تغییر این رفتار (یعنی: برای حفظ logها پس از خروج از اسکریپت) باید خطوط ۹۰ الی ۹۷ را کامنت کرد.
- **حال، آمادهٔ استفاده از اسکریپت‌های فوق هستیم. ابتدا بررسی کنید که سرویس quagga را طوری تنظیم کرده باشید که به صورت اتوماتیک شروع نشود و سپس، محیط شبیه‌سازی را اجرا نمایید.**
- **سؤال ۱: از خط دستور Mininet، دستور net را اجرا نموده و بررسی کنید که شبکه مطابق با شکل ۱ ساخته شده باشد. آیا می‌توانید بین همگی روترهای مجازی ping کنید؟ توضیح دهید چرا می‌توانید یا چرا نمی‌توانید؟!**



شکل ۱- توپولوژی آزمایش ۷

ج) پیکربندی شبکه با استفاده از Quagga

برای این بخش، برنامه را کامل ببندید. ادامه توضیحات را مطالعه نموده و مابقی مراحل آزمایش را طی کنید:

هر روتر مجازی، در واقع، مشابه یک ماشین فیزیکی Linux است و بنابراین می‌توان به همان طریق هم آن را پیکربندی نمود؛ یعنی، شما می‌توانید همان مجموعهٔ دستوراتی را که در آزمایش‌های پیشین فرا گرفته‌اید برای پیکربندی اینترفیس‌های شبکه استفاده نموده و وضعیت آنها را مانیتور نمایید یا اینکه محتوای جداول مسیریابی آنها را مشاهده کنید و غیره.

با این حال، در این آزمایش، به جای استفاده از مجموعهٔ دستورات شبکه‌ایِ Linux، از قابلیت‌های فراهم شده توسط Quagga بهره خواهیم گرفت. یکی از مزیت‌های Quagga این است که دستوراتی که توسط آن پشتیبانی می‌شوند، در واقع، زیرمجموعه‌ای از دستورات مورد استفاده برای پیکربندیِ تجهیزات شبکه‌ای Cisco هستند.

مستندات کامل کار با Quagga از طریق www.nongnu.org/quagga/ قابل دستیابی است ضمن اینکه می‌توان از خود سایت Cisco هم به مراجع مفیدی در این رابطه دست یافت.

* توجه! هرگز تلاش نکنید که اینترفیس‌های شبکه‌ای روی یک ماشین را به طور همزمان توسط هر دو نوع دستور ip addr و دستورات پروسس zebra پیکربندی نمایید. در واقع، تعامل این دو همیشه به خروجی قطعی و قابل اطمینانی منجر نمی‌گردد.

ج-۱) پیکربندی اینترفیس‌های شبکه با استفاده از فایل‌های پیکربندی

پیش از اجرای توپولوژی، نیاز است که حداقل برای پروسس zebra، یک فایل پیکربندی ساخته شده و ویرایش گردد. فایل با نام zebra.conf که در اختیار دارید، نمونه‌ای از چنین فایلی است که باید به طور اختصاصی برای هر روتر نوشته شود.

- **فایل zebra.conf ویژهٔ روتر r1 را با استفاده از برنامهٔ leafpad باز کنید و محتوای آن را به همراه توضیحات زیر مطالعه نمایید.**

  - خطوطی که با نشانهٔ ! شروع می‌شوند، کامنت هستند و از آنها صرف‌نظر خواهد شد.
  - password: کلمهٔ عبور مورد استفاده برای login کردن در پروسس zebra و مشاهدهٔ تغییرات.
  - enable password: کلمهٔ عبور مورد استفاده برای فعال‌سازیِ قابلیت پیکربندیِ پویای پروسس zebra.
  - log file: امکان می‌دهد که شما فایل logی را که داخل آن رویدادهای مربوط به zebra ضبط می‌شود، مشخص نمایید (این رویدادها عمدتاً مربوط به اضافه و حذف مسیرها هستند). اطمینان حاصل کنید که مسیر کامل را برای این فایل‌ها مشخص نموده‌اید چراکه در غیر این صورت، سرویس Quagga برای استارت خود به مشکل می‌خورد.
  - debug zebra packet: امکان debugging جزئی‌تری را فراهم می‌نماید؛ یعنی، می‌توانید مشاهده کنید که چه مسیرهایی در جدول مسیریابی اضافه و حذف می‌شوند.
  - interface <interface name>: این دستور مُد پیکربندیِ اینترفیس را آغاز می‌کند.
  - ip address: آدرس IP ورژن ۴ را برای اینترفیس مورد نظر تعیین می‌کند.
  - line vty: قابلیت دسترسی مبتنی بر telnet به پروسس را می‌دهد.

کلیهٔ اسکریپت‌های پیکربندی ویژهٔ روترهای r1، r2 و r4 از پیش تهیه شده‌اند و شما می‌توانید از طریق فولدر lab7 که در اختیار دارید، به آنها دسترسی داشته باشید.

- **فایل‌های پیکربندی لازم برای روترهای r3 و r5 را بسازید (شامل سه فایل: daemons، zebra.conf و debian.conf). برای این منظور، باید با توجه به شکل ۱ و طبق توپولوژی شبکه، اینترفیس‌های روترها را پیکربندی نمایید (آدرس IP تخصیص دهید). همچنین، دقت کنید که در کلیهٔ فایل‌های daemons، صرفاً پروسس zebra فعال شده باشد و مابقی پروسس‌ها را با "no" تنظیم کرده باشید.**

* اطمینان حاصل کنید که اجازهٔ دسترسی (permission) فایل‌های zebra.log طوری باشد که به شما امکان دسترسی read-write بدهد. برای اینکه بتوان یک اسکریپت را برای همه (anyone)، از نوع read-write کرد، می‌توانید دستور زیر را در پنجرهٔ ترمینال تایپ کنید (با داشتن دسترسی root):

```
chmod 666 <script name>
```

ج-۲) مُد مانیتورینگ Quagga

حال اسکریپت با عنوان lab7_network.py را مجدداً اجرا نمایید.

- **سرویس quagga را در کلیهٔ روترها استارت کنید (البته پیش از آغاز پروسس quagga باید مطمئن شوید که در فایل daemons، گزینهٔ zebra=yes وجود داشته باشد). برای آغاز کردن سرویس quagga می‌توان از ترمینال هر روتری، دستور زیر را تایپ نمود:**

```
/etc/init.d/quagga start
```

- **البته به عنوان راهکار دیگر، می‌توانستیم از طریق فایل lab7_topo.py مقرر کنیم که سرویس Quagga به طور خودکار استارت شود.**

به منظور مانیتور کردن فعالیت یک پروسس Quaggaی در حال اجرا، می‌توانید از طریق telnet به پروسس مورد نظر متصل شده و وارد مُد مانیتورینگ شوید.

- **با استفاده از دستور زیر به پروسس zebraی در حال اجرا روی روتر r1 متصل شوید:**

```
telnet localhost zebra
```

- **به منظور مشاهدهٔ محتوای جداول مسیریابی IPv4 در روتر r1، دستور show ip route را تایپ نمایید.**

همواره می‌توانید لیست کامل دستورات مورد پشتیبانی را با تایپ دستور list مشاهده نمایید.

- **سؤال ۲: چه subnetهایی را می‌توان از روی جدول مسیریابی موجود در r1 تشخیص داد؟**
- **حال، وضعیت اینترفیس‌های شبکه روی روتر r3 را با تایپ دستور show interface بررسی کنید.**
- **سؤال ۳: چند اینترفیس نمایش داده می‌شود؟ نام ببرید.**
- **برای مشاهدهٔ پیکربندیِ جاری و در حال اجرا، باید ابتدا با استفاده از دستور enable، مُد پیکربندی را فعال کنید.**

- **در نهایت، پیکربندی در حال اجرایِ روتر r3 را با استفاده از دستور show running-config بررسی کنید.**

ج-۳) پیکربندی Quagga ضمن اجرا !!

در این بخش شرح می‌دهیم که چطور بدون استفاده از فایل‌های پیکربندی، می‌توان یک پیکربندیِ در حال اجرا را مورد تغییر قرار داد. البته پیروی از این روش در حالت کلی، توصیه نمی‌شود. با این حال، برای تکمیل مستند آزمایش، از طریق مثالی، نشان می‌دهیم که گاهاً چگونه پیکربندی پویا می‌تواند مفید واقع گردد.

در فایل کانفیگِ zebra برای روتر شمارهٔ ۴، عمداً یکی از خطوط که مربوط به انتساب آدرس IP به اینترفیسِ r4-eth2 از روتر r4 است، کامنت شده است (با استفاده از نشانهٔ !).

یک راهکار برای رفع این اشکال این است که سرویس quagga را روی این روتر استاپ کرد و سپس خط مورد نظر را اصلاح نمود و در نهایت، مجدداً اقدام به راه‌اندازی سرویس نمود.

امّا ما در این بخش از راهکار دیگری برای رفع این اشکال استفاده می‌کنیم. برای این منظور، روی روتر r4 وارد مُد پیکربندی Quagga شده و به مُد پیکربندی اینترفیس دسترسی می‌یابیم.

برای ورود به «مُد پیکربندی اینترفیس[۲]» در روتر r4 گام‌های زیر را طی نمایید:

- **ابتدا با استفاده از دستور telnet localhost zebra وارد مُد مانیتورینگ zebra بشوید.**
- **با اجرای دستور enable، مُد پیکربندی را فعال نمایید.**
- **با اجرای فرمان configure terminal، به مُد پیکربندی وارد شوید. با این اقدام، اگرچه cursor به همان شکل سابق خود می‌ماند ولی نام پیش از cursor به r4(config) تغییر می‌یابد.**
- **در نهایت، همچنانکه داخل مُد پیکربندی هستید، دستور interface r4-eth2 را تایپ نمایید. نام پیش از cursor به r4(config-if) تغییر می‌یابد که نمایانگر این است که شما وارد مُد پیکربندی اینترفیس شده‌اید.**
- **حال، می‌توانید به اینترفیس r4-eth2 آدرس IP ورژن ۴ اختصاص دهید (با استفاده از دستوری مشابه آنچه در فایل پیکربندی برای این منظور تدارک دیده شده بود). پس کافی است که تایپ کنید: ip address 10.10.45.4/24.**

---

[2] Interface configuration mode

- با دو مرتبه تایپ کردن دستور **exit**، از «مُد پیکربندی اینترفیس» و سپس از «مُد پیکربندی» خارج شوید.
- با تایپ دستور **show interface** مطمئن شوید که تغییراتی که داده‌اید، واقعاً ترتیب اثر داده شده باشند.
- پیش از ترک این بخش از آزمایش، مطمئن شوید که فایل پیکربندی **zebra.conf** برای روتر **r4** را تغییر دهید تا نشانهٔ کامنت از مقابل اینترفیس **r4-eth2** حذف شود چراکه پیکربندی پویا تغییرات داده شده را در فایل پیکربندی ذخیره نمی‌نماید.