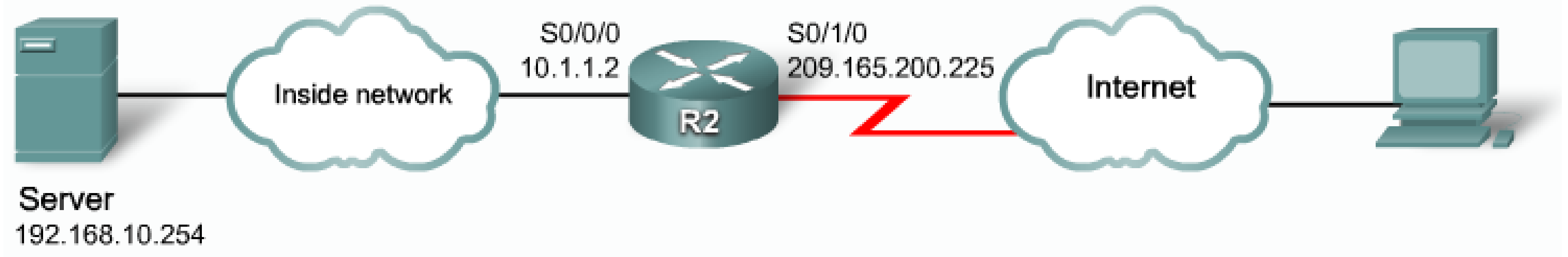
Configuring Static NAT

Step	Action	Notes
1	Establish static translation between an inside local address and an inside global address. `Router(config)#`**`ip nat inside source static`** *`local-ip global-ip`*	Enter the global command **`no ip nat inside source static`** to remove the static source translation.
2	Specify the inside interface. `Router(config)#`**`interface`** *`type number`*	Enter the **`interface`** command. The CLI prompt will change from `(config)#` to `(config-if)#`.
3	Mark the interface as connected to the inside. `Router(config-if)#`**`ip nat inside`**	
4	Exit interface configuration mode. `Router(config-if)#` **`exit`**	
5	Specify the outside interface. `Router(config)#`**`interface`** *`type number`*	
6	Mark the interface as connected to the outside. `Router(config-if)#`**`ip nat outside`**	

Example



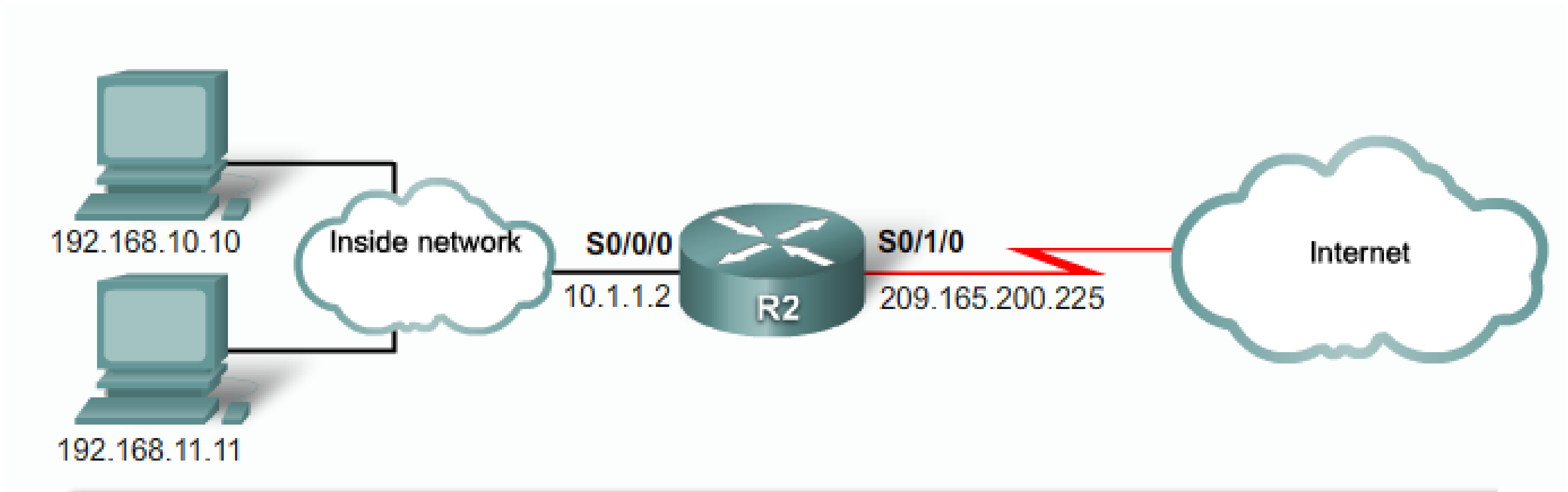
```
ip nat inside source static 192.168.10.254 209.165.200.254
!–Establishes static translation between an inside local address and an inside global address.
interface serial 0/0/0
ip nat inside
!—Identifies Serial 0/0/0 as an inside NAT interface.
interface serial 0/1/0
ip nat outside
!—Identifies Serial 0/1/0 as an outside NAT interface.
```

With this configuration, 192.168.10.254 will always translate to 209.165.200.254

Configuring Dynamic NAT

Step	Action	Notes
1	Define a pool of global addresses to be allocated as needed. `Router(config)#`**`ip nat pool name`** *`start-ip end-ip`* `{`**`netmask`** *`netmask`*`|`**`prefix-length`** *`prefix-length`*`}`	Enter the global command **`no ip nat pool`** *`name`* to remove the pool of global addresses.
2	Define a standard access list permitting those addresses that are to be translated. `Router(config)#`**`access-list`** `access-list-number` **`permit`** `source [source-wildcard]`	Enter the global command **`no access-list`** *`access-list-number`* to remove the access list.
3	Establish dynamic source translation, specifying the access list defined in the prior step. `Router(config)#`**`ip nat inside source list`** *`access-list-number`* **`pool`** *`name`*	Enter the global command **`no ip nat inside source`** to remove the dynamic source translation.
4	Specify the inside interface. `Router(config)#`**`interface`** *`type number`*	Enter the **`interface`** command. The CLI prompt will change from `(config)#` to `(config-if)#`.
5	Mark the interface as connected to the inside. `Router(config-if)#`**`ip nat inside`**	
6	Specify the outside interface. `Router(config)#`**`interface`** *`type number`*	
7	Mark the interface as connected to the outside. `Router(config-if)#`**`ip nat outside`**	
8	Exit interface configuration mode. `Router(config-if)#` **`exit`**	

Example



```
ip nat pool NAT-POOL1 209.165.200.226 209.165.200.240 netmask 255.255.255.224
!—Defines a pool of public IP addresses under the pool name NAT-POOL1
access-list 1 permit 192.168.0.0 0.0.255.255
!—Defines which addresses are eligible to be translated
ip nat inside source list 1 pool NAT-POOL1
!—Binds the NAT pool with ACL 1
interface serial 0/0/0
  ip nat inside
!—Identifies interface Serial 0/0/0 as an inside NAT interface
interface serial 0/1/0
  ip nat outside
!—Identifies interface Serial 0/1/0 as the outside NAT interface
```

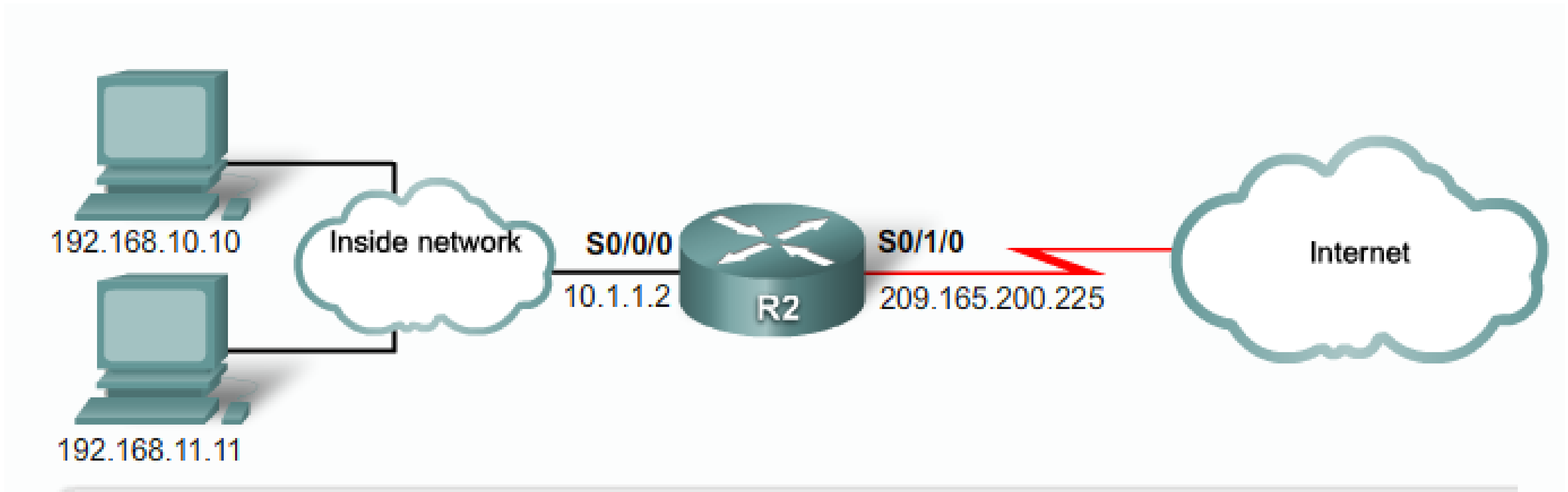
Configuring NAT Overload for a Single Public IP Address

Step	Action	Notes
1	Define a standard access list permitting those addresses that are to be translated. `Router(config)#`**`access-list`** *`acl-number`* **`permit`** *`source [source-wildcard]`*	Enter the global command **`no access-list`** *`access-list-number`* to remove the access list.
2	Establish dynamic source translation, specifying the access list defined in the prior step. `Router(config)#`**`ip nat inside source list`** *`acl-number`* **`interface`** *`interface`* **`overload`**	Enter the global command **`no ip nat inside source`** to remove the dynamic source translation. The overload keyword enables PAT.

3	Specify the inside interface.
	`Router(config)#`**`interface`** *`type number`*
	`Router(config-if)#`**`ip nat inside`**
		Enter the **`interface`** command. The CLI prompt will change from `(config)#` to `(config-if)#`.

4	Specify the outside interface.
	`Router(config-if)#`**`interface`** *`type number`*
	`Router(config-if)#`**`ip nat  outside`**	

Example



```
access-list 1 permit 192.168.0.0 0.0.255.255
```
!—Defines which addresses are eligible to be translated
```
ip nat inside source list 1 interface serial 0/1/0 overload
```
!—Identifies the outside interface Serial 0/1/0 as the inside global address to be overloaded
```
interface serial 0/0/0
  ip nat inside
```
!—Identifies interface Serial 0/0/0 as an inside NAT interface
```
interface serial 0/1/0
  ip nat outside
```
!—Identifies interface Serial 0/1/0 as the outside NAT interface

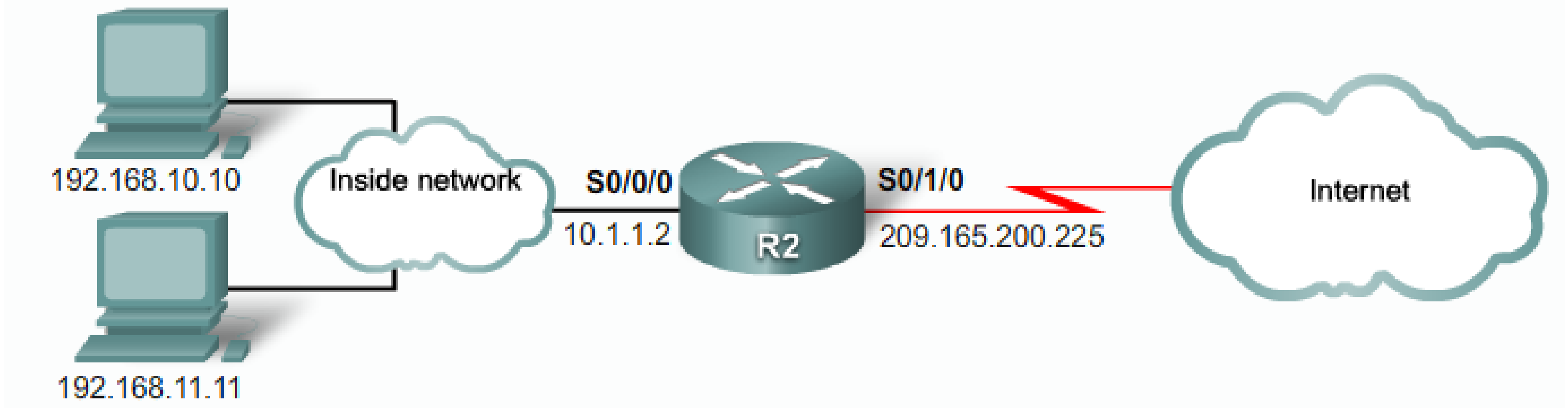
NAT Overload Configuration Using a Pool of Public Addresses

Step	Action	Notes
1	Define a standard access list permitting those addresses that are to be translated. `Router(config)#access-list acl-number permit source [source-wildcard]`	Enter the global command `no access-list access-list-number` to remove the access list.
2	Specify the global address, as a pool, to be used for overloading. `Router(config)#ip nat pool name start-ip end-ip {netmask netmask | prefix-length prefix-length}.`	
3	Establish overload translation. `Router{config}#ip nat inside source list acl-number  pool name overload.`	

4	Specify the inside interface.
	`Router(config)#interface type number`
	`Router(config-if)#ip nat inside`
		Enter the **interface** command. The CLI prompt will change from `(config)#` to `(config-if)#`.

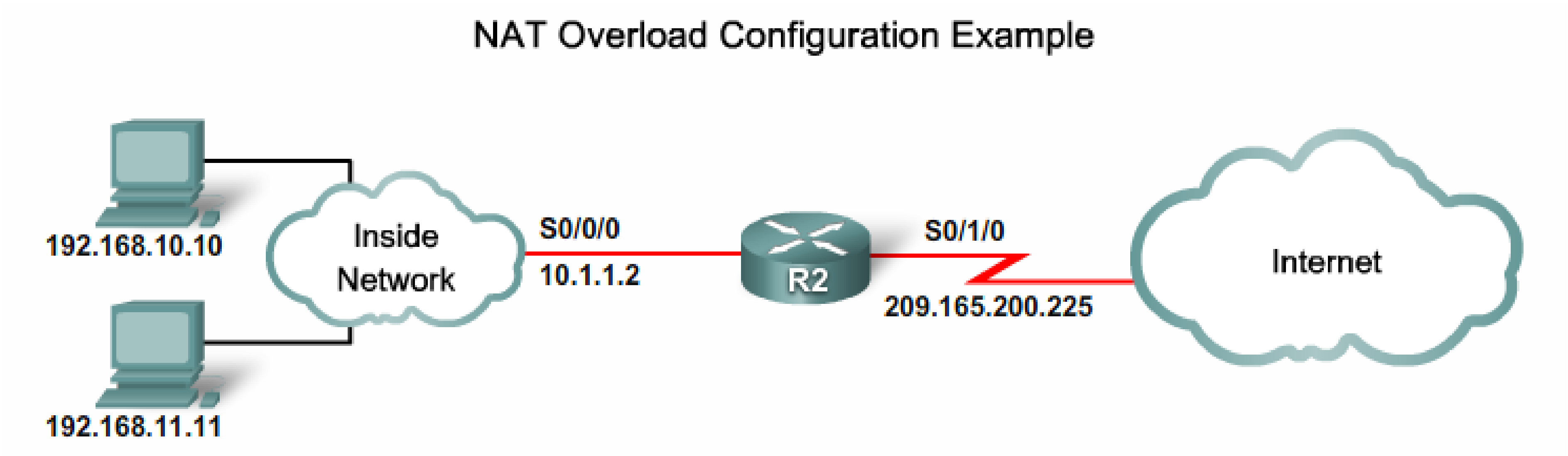
5	Specify the outside interface.
	`Router(config-if)#interface type number`
	`Router(config-if)#ip nat  outside`	

Example



```
access-list 1 permit 192.168.0.0 0.0.255.255
```
! – Defines which addresses are eligible to be translated
```
ip nat pool NAT-POOL2 209.165.200.226 209.165.200.240
```
! – Defines a pool of addresses named NAT-POOL2 to be used in NAT translation
```
ip nat inside source list 1 pool NAT-POOL2 overload
```
! – Binds the NAT pool with ACL 1
```
interface serial 0/0/0
ip nat inside
```
! – Indentifies interface Serial 0/0/0 as an inside NAT interface
```
interface serial 0/1/0
ip nat outside
```
! – Indentifies interface Serial 0/1/0 as an outside NAT interface

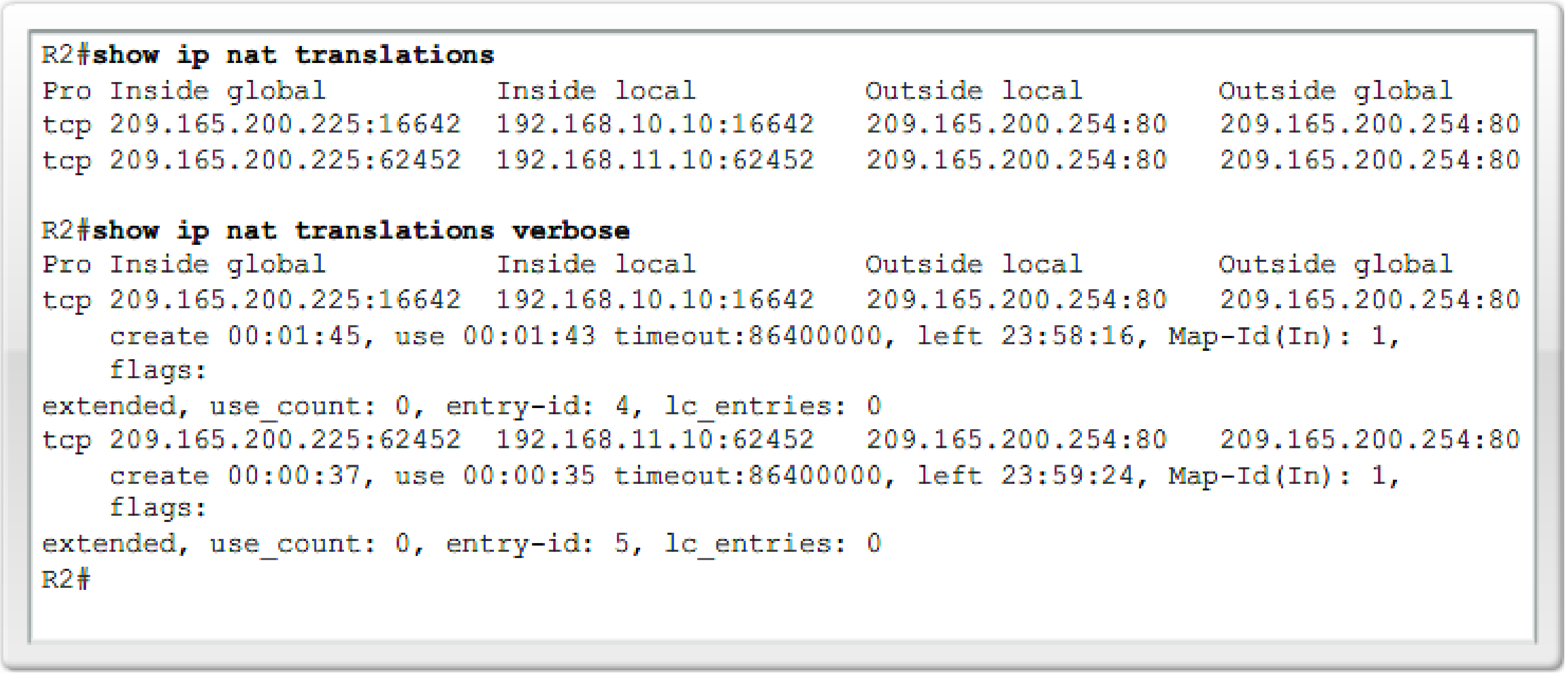
Verifying NAT and NAT Overload



```
access-list 1 permit 192.168.0.0 0.0.255.255
ip nat inside source list 1 interface serial 0/1/0 overload
interface serial 0/0/0
  ip nat inside
interface serial 0/1/0
  ip nat outside
```

Verifying NAT and NAT Overload

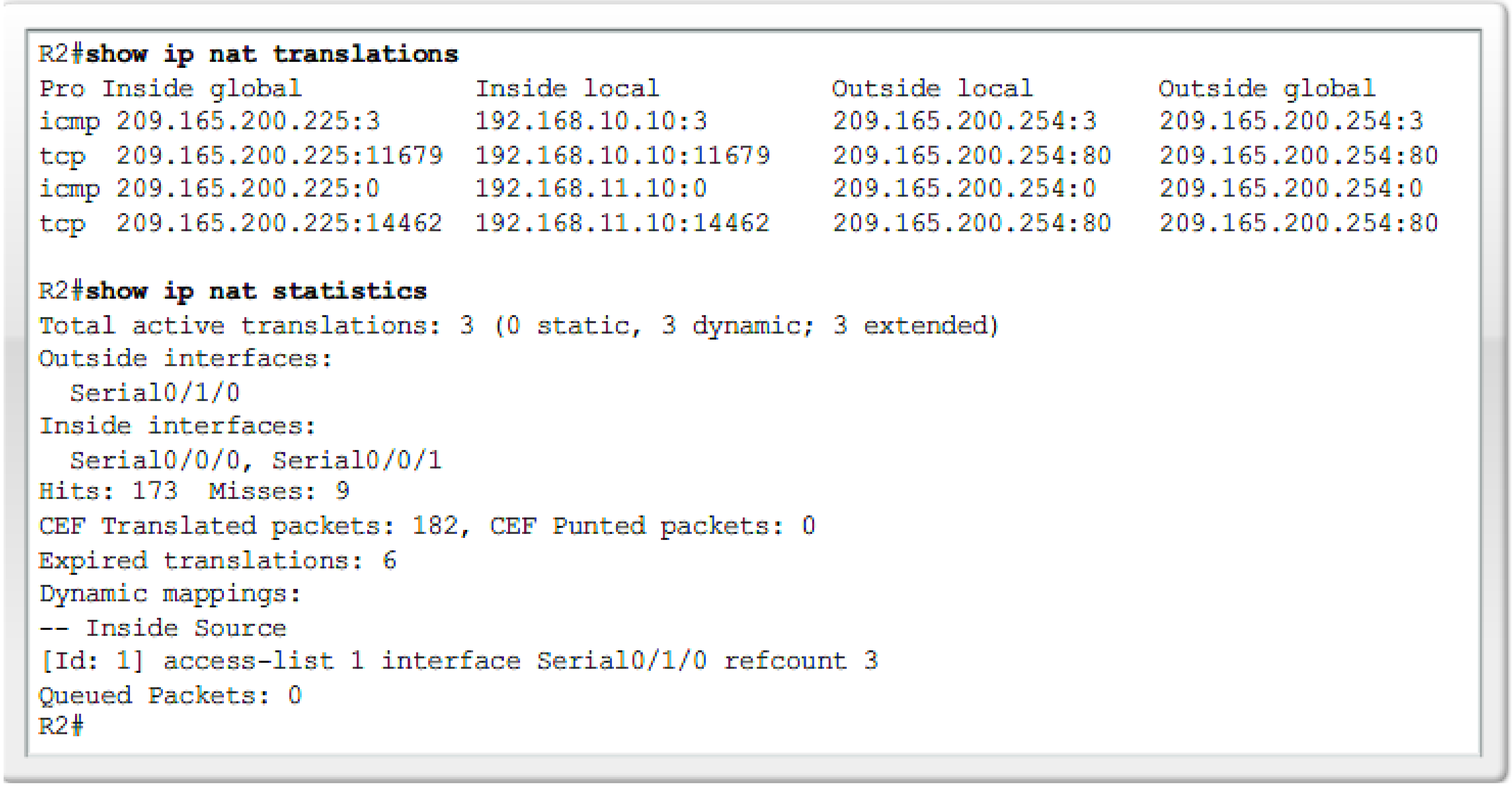
NAT Translations Example



```
R2#show ip nat translations
Pro Inside global          Inside local         Outside local        Outside global
tcp 209.165.200.225:16642  192.168.10.10:16642  209.165.200.254:80   209.165.200.254:80
tcp 209.165.200.225:62452  192.168.11.10:62452  209.165.200.254:80   209.165.200.254:80

R2#show ip nat translations verbose
Pro Inside global          Inside local         Outside local        Outside global
tcp 209.165.200.225:16642  192.168.10.10:16642  209.165.200.254:80   209.165.200.254:80
    create 00:01:45, use 00:01:43 timeout:86400000, left 23:58:16, Map-Id(In): 1,
    flags:
extended, use_count: 0, entry-id: 4, lc_entries: 0
tcp 209.165.200.225:62452  192.168.11.10:62452  209.165.200.254:80   209.165.200.254:80
    create 00:00:37, use 00:00:35 timeout:86400000, left 23:59:24, Map-Id(In): 1,
    flags:
extended, use_count: 0, entry-id: 5, lc_entries: 0
R2#
```

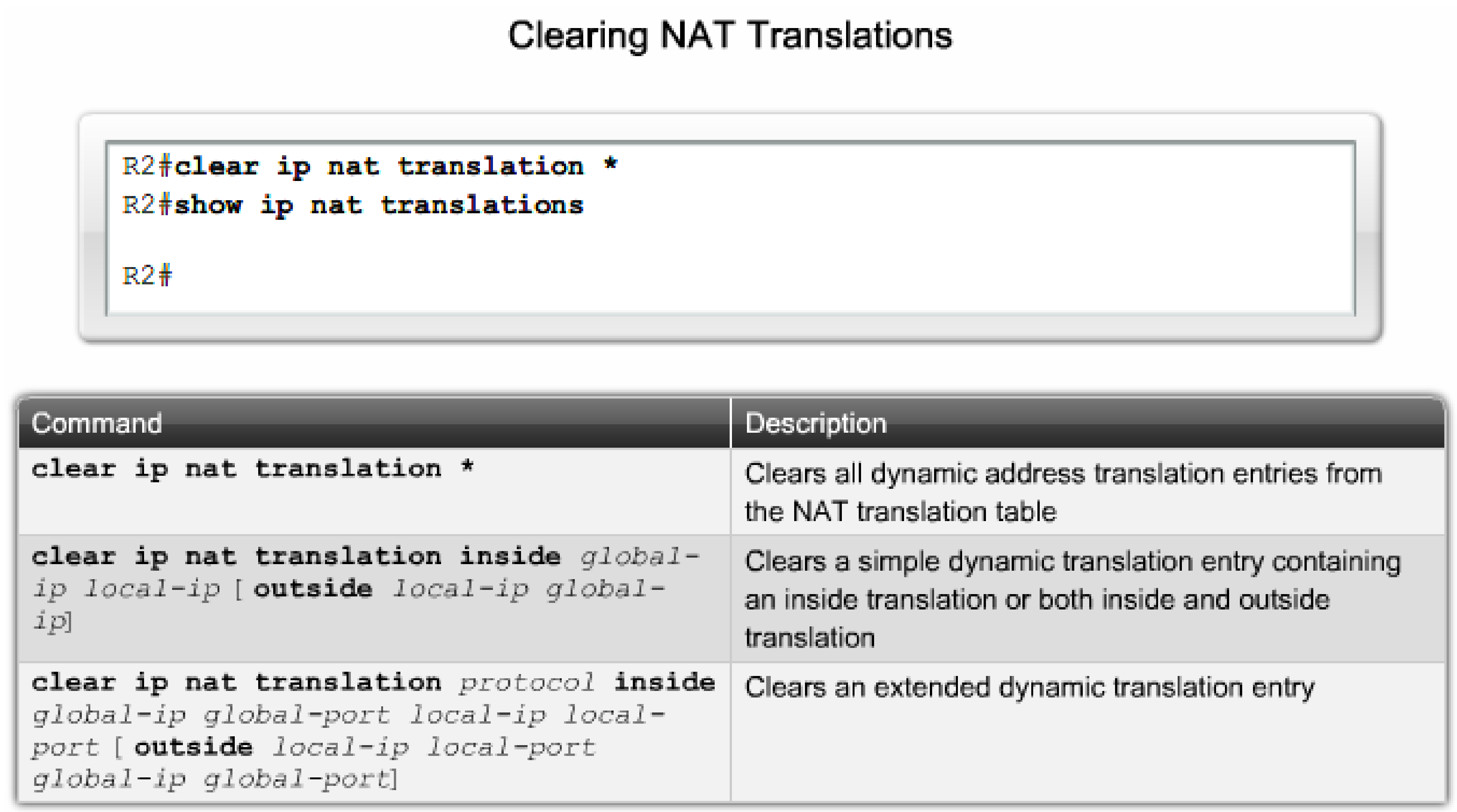
Verifying NAT and NAT Overload

NAT Translations Example



```
R2#show ip nat translations
Pro Inside global          Inside local          Outside local        Outside global
icmp 209.165.200.225:3     192.168.10.10:3       209.165.200.254:3    209.165.200.254:3
tcp  209.165.200.225:11679 192.168.10.10:11679   209.165.200.254:80   209.165.200.254:80
icmp 209.165.200.225:0     192.168.11.10:0       209.165.200.254:0    209.165.200.254:0
tcp  209.165.200.225:14462 192.168.11.10:14462   209.165.200.254:80   209.165.200.254:80

R2#show ip nat statistics
Total active translations: 3 (0 static, 3 dynamic; 3 extended)
Outside interfaces:
  Serial0/1/0
Inside interfaces:
  Serial0/0/0, Serial0/0/1
Hits: 173  Misses: 9
CEF Translated packets: 182, CEF Punted packets: 0
Expired translations: 6
Dynamic mappings:
-- Inside Source
[Id: 1] access-list 1 interface Serial0/1/0 refcount 3
Queued Packets: 0
R2#
```

Verifying NAT and NAT Overload

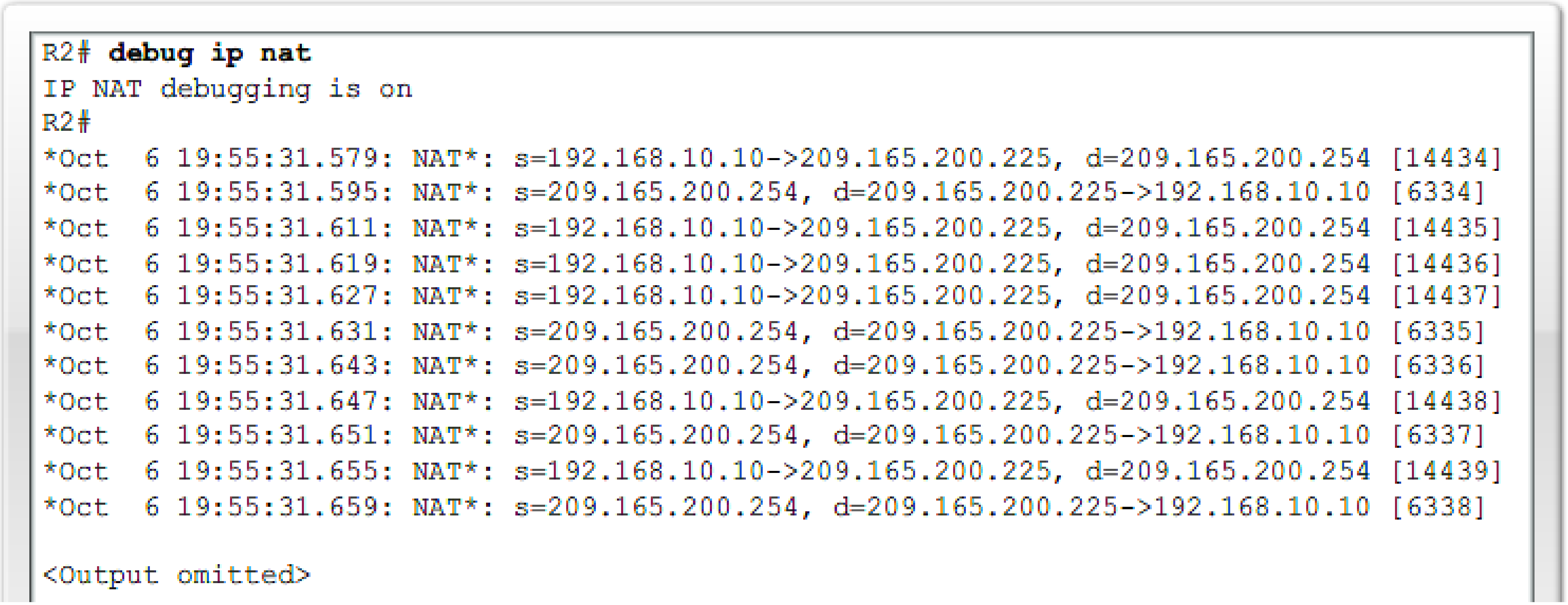


Clearing NAT Translations

```
R2#clear ip nat translation *
R2#show ip nat translations

R2#
```

Command	Description
clear ip nat translation *	Clears all dynamic address translation entries from the NAT translation table
clear ip nat translation inside *global-ip local-ip* [**outside** *local-ip global-ip*]	Clears a simple dynamic translation entry containing an inside translation or both inside and outside translation
clear ip nat translation *protocol* **inside** *global-ip global-port local-ip local-port* [**outside** *local-ip local-port global-ip global-port*]	Clears an extended dynamic translation entry

Debug NAT Translations



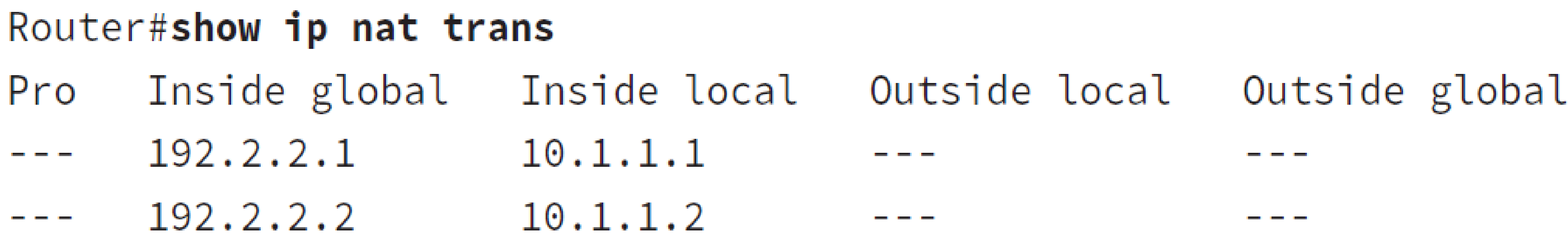
```
R2# debug ip nat
IP NAT debugging is on
R2#
*Oct  6 19:55:31.579: NAT*: s=192.168.10.10->209.165.200.225, d=209.165.200.254 [14434]
*Oct  6 19:55:31.595: NAT*: s=209.165.200.254, d=209.165.200.225->192.168.10.10 [6334]
*Oct  6 19:55:31.611: NAT*: s=192.168.10.10->209.165.200.225, d=209.165.200.254 [14435]
*Oct  6 19:55:31.619: NAT*: s=192.168.10.10->209.165.200.225, d=209.165.200.254 [14436]
*Oct  6 19:55:31.627: NAT*: s=192.168.10.10->209.165.200.225, d=209.165.200.254 [14437]
*Oct  6 19:55:31.631: NAT*: s=209.165.200.254, d=209.165.200.225->192.168.10.10 [6335]
*Oct  6 19:55:31.643: NAT*: s=209.165.200.254, d=209.165.200.225->192.168.10.10 [6336]
*Oct  6 19:55:31.647: NAT*: s=192.168.10.10->209.165.200.225, d=209.165.200.254 [14438]
*Oct  6 19:55:31.651: NAT*: s=209.165.200.254, d=209.165.200.225->192.168.10.10 [6337]
*Oct  6 19:55:31.655: NAT*: s=192.168.10.10->209.165.200.225, d=209.165.200.254 [14439]
*Oct  6 19:55:31.659: NAT*: s=209.165.200.254, d=209.165.200.225->192.168.10.10 [6338]

<Output omitted>
```

Testing and Troubleshooting NAT

- **List of potential causes:**
  - Check the dynamic pools. Are they composed of the right scope of addresses?
  - Check to see if any dynamic pools overlap.
  - Check to see if the addresses used for static mapping and those in the dynamic pools overlap.
  - Ensure that your access lists specify the correct addresses for translation.
  - Make sure there aren't any addresses left out that need to be there, and ensure that none are included that shouldn't be.
  - Check to make sure you've got both the inside and outside interfaces delimited properly.

Testing and Troubleshooting NAT

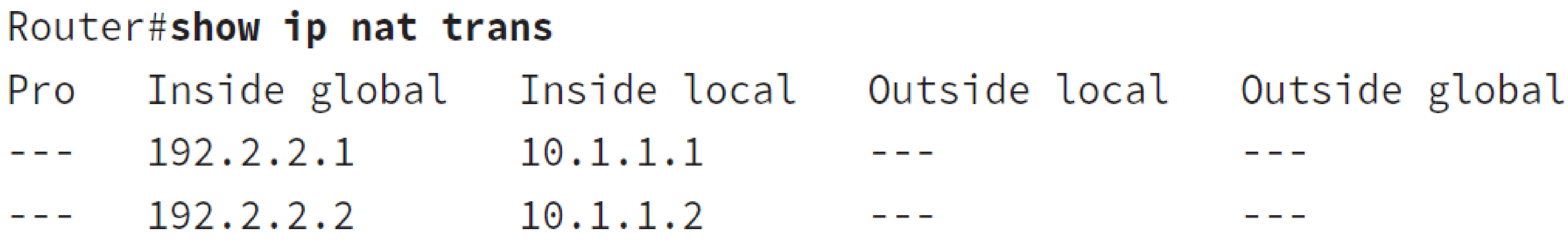
- make sure your router still knows what to do with the new address after the translation!
- The first command you should typically use is the show ip nat translations command:



```
Router#show ip nat trans
Pro  Inside global    Inside local    Outside local    Outside global
---  192.2.2.1        10.1.1.1        ---              ---
---  192.2.2.2        10.1.1.2        ---              ---
```

- After checking out this output, can you tell me if the configuration on the router is static or dynamic NAT?
- The answer is yes

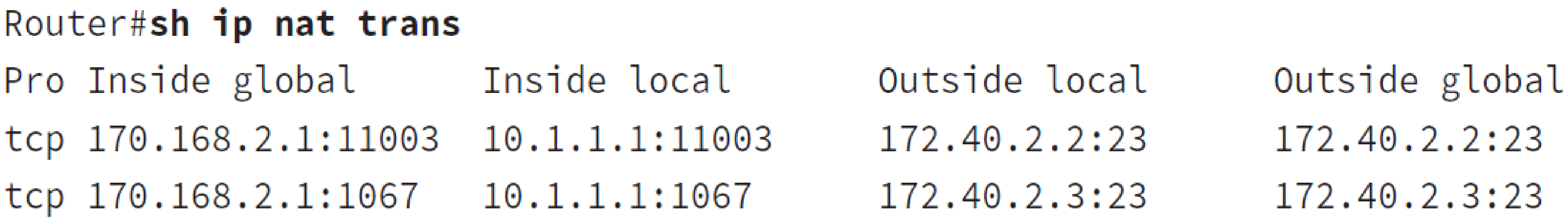
Testing and Troubleshooting NAT



```
Router#show ip nat trans
Pro   Inside global    Inside local    Outside local    Outside global
---   192.2.2.1        10.1.1.1        ---              ---
---   192.2.2.2        10.1.1.2        ---              ---
```

- Either static or dynamic NAT is configured because there’s a one-to-one translation from the inside local to the inside global.
- Basically, by looking at the output, you can’t tell if it’s static or dynamic per se; but you absolutely can tell that you’re not using PAT because there are no port numbers.

Testing and Troubleshooting NAT



```
Router#sh ip nat trans
Pro Inside global       Inside local       Outside local      Outside global
tcp 170.168.2.1:11003  10.1.1.1:11003     172.40.2.2:23      172.40.2.2:23
tcp 170.168.2.1:1067   10.1.1.1:1067      172.40.2.3:23      172.40.2.3:23
```

- you can easily see that the above output is using NAT Overload (PAT).
- The protocol in this output is TCP, and the inside global address is the same for both entries.
- things like memory and CPU, or even the boundaries set in place by the scope of available addresses or ports, can cause limitations on the actual number of entries.

Testing and Troubleshooting NAT

- Consider that each NAT mapping devours about 160 bytes of memory.
- And sometimes the amount of entries must be limited for the sake of performance or because of policy restrictions, but this doesn’t happen very often.
- In situations like these, just go to the **ip nat translation max-entries** command for help.
- Deploying **show ip nat statistics** gives you a summary of the NAT configuration, and it will count the number of active translation types too.

Testing and Troubleshooting NAT

- Also counted are hits to an existing mapping as well any misses, with the latter causing an attempt to create a mapping.
- This command will also reveal expired translations.
- If you want to: check into dynamic pools, their types, the total available addresses, how many addresses have been allocated and how many have failed, plus the number of translations that have occurred, just use the **pool** keyword.

Testing and Troubleshooting NAT

```
Router#debug ip nat
NAT: s=10.1.1.1->192.168.2.1, d=172.16.2.2 [0]
NAT: s=172.16.2.2, d=192.168.2.1->10.1.1.1 [0]
NAT: s=10.1.1.1->192.168.2.1, d=172.16.2.2 [1]
NAT: s=10.1.1.1->192.168.2.1, d=172.16.2.2 [2]
NAT: s=10.1.1.1->192.168.2.1, d=172.16.2.2 [3]
NAT*: s=172.16.2.2, d=192.168.2.1->10.1.1.1 [1]
```

- Notice the last line in the output and how the NAT at the beginning of the line has an asterisk (*).
- **This means the packet was translated and fast-switched to the destination.**

What’s fast-switched?

- fast-switching has gone by several aliases such as cache-based switching and this nicely descriptive name, “route one switch many.”
- The fast-switching process is used on Cisco routers to create a cache of layer 3 routing information to be accessed at layer 2 so packets can be forwarded quickly through a router without the routing table having to be parsed for every packet.
- As packets are packet switched (looked up in the routing table), this information is stored in the cache for later use if needed for faster routing processing.

Testing and Troubleshooting NAT

- you can manually clear dynamic NAT entries from the NAT table.
- You can, and doing this can come in seriously handy if you need to get rid of a specific rotten entry without sitting around waiting for the timeout to expire!
- A manual clear is also really useful when you want to clear the whole NAT table to reconfigure a pool of addresses.
- You also need to know that the Cisco IOS software just won’t allow you to change or delete an address pool if any of that pool’s addresses are mapped in the NAT table.

Testing and Troubleshooting NAT

- The **clear ip nat translations** command clears entries
- you can indicate a single entry via the global and local address and through TCP and UDP translations, including ports, or you can just type in an asterisk (*) to wipe out the entire table.
- But know that if you do that, only dynamic entries will be cleared because this command won't remove static entries.

Testing and Troubleshooting NAT

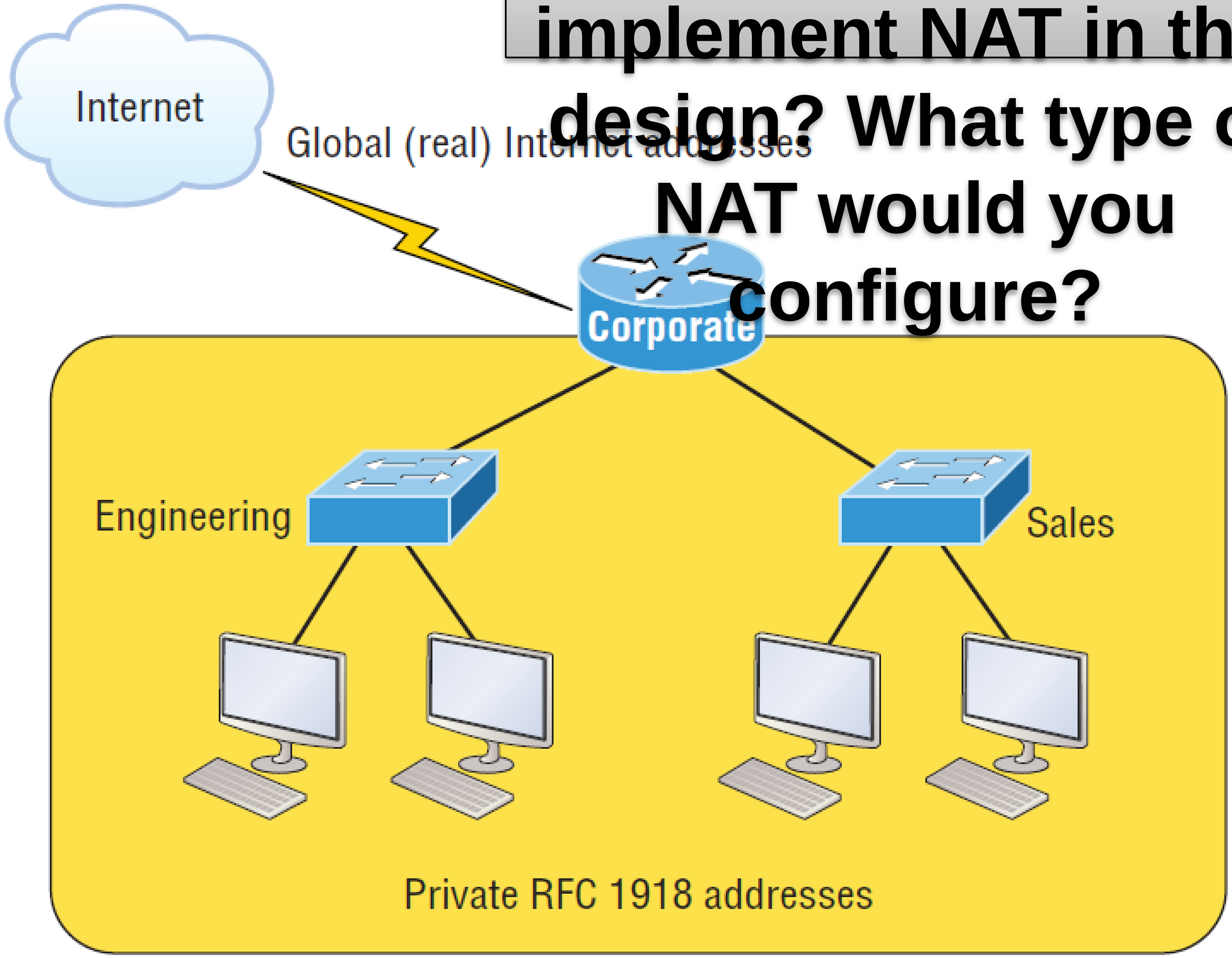
- any outside device's packet destination address that happens to be responding to any inside device is known as the **inside global (IG) address**.
- This means that the initial mapping has to be held in the NAT table so that all packets arriving from a specific connection get translated consistently.
- Holding entries in the NAT table also cuts down on repeated translation operations

Testing and Troubleshooting NAT

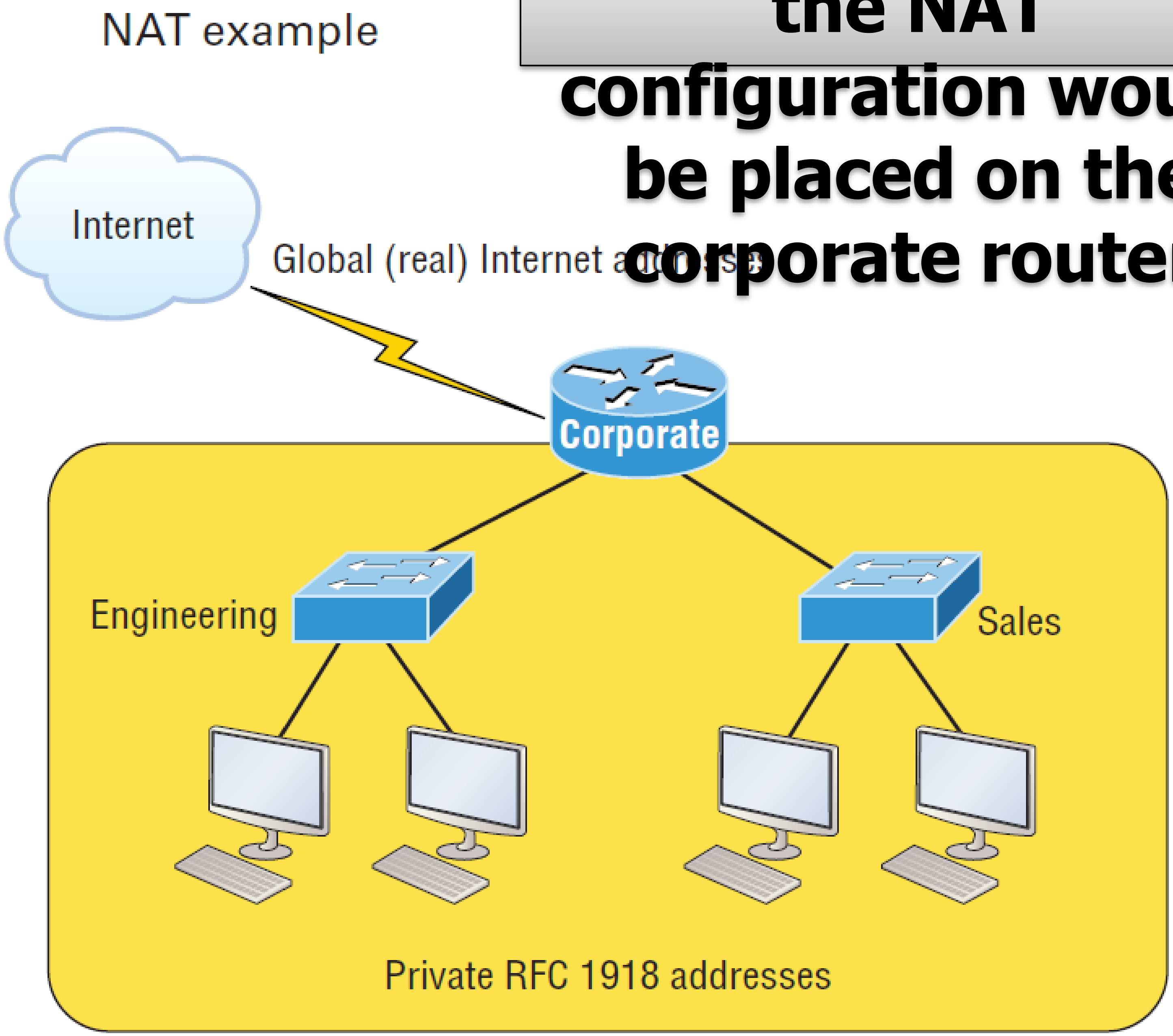
- when an entry is placed into the NAT table the first time, a timer begins ticking and its duration is known as the translation timeout.
- Each time a packet for a given entry translates through the router, the timer gets reset.
- If the timer expires, the entry will be unceremoniously removed from the NAT table and the dynamically assigned address will then be returned to the pool.
- Cisco’s default translation timeout is 86,400 seconds (24 hours), but you can change that with the **ip nat translation timeout** command.

NAT example

Where would you implement NAT in this design? What type of NAT would you configure?



the NAT configuration would be placed on the corporate router



In this next NAT example, what type of NAT is being used?

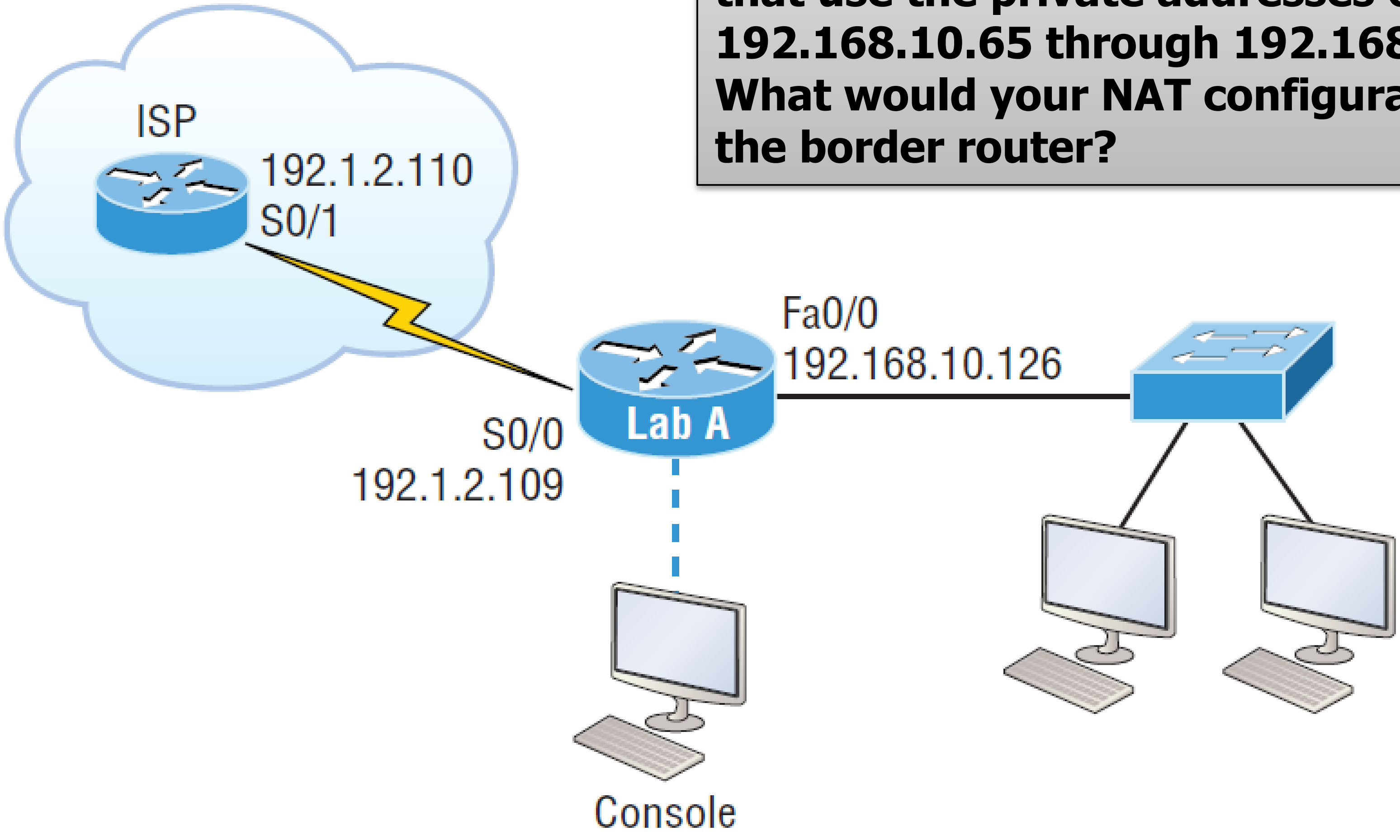
```
ip nat pool todd-nat 170.168.10.10 170.168.10.20 netmask 255.255.255.0
ip nat inside source list 1 pool todd-nat
ip nat pool todd-nat 170.168.10.10 170.168.10.20 netmask 255.255.255.0
ip nat inside source list 1 pool todd-nat
```

The preceding command uses dynamic NAT without PAT.

The pool in the command gives the answer away as dynamic, plus there's more than one address in the pool

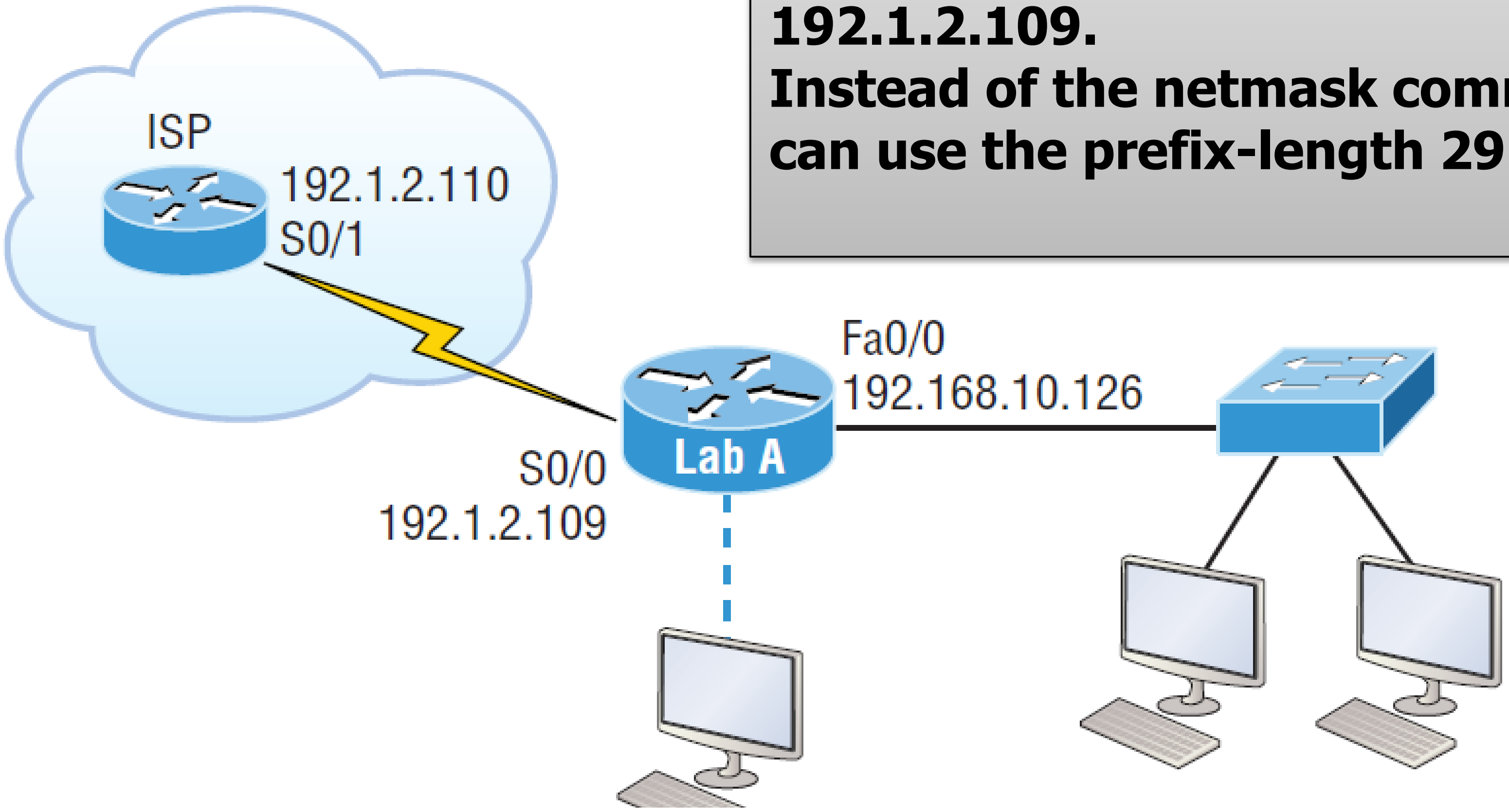
Another NAT example

The border router needs to be configured with NAT and allow the use of six public IP addresses to the inside locals, 192.1.2.109 through 192.1.2.114.
on the inside network, you have 62 hosts that use the private addresses of 192.168.10.65 through 192.168.10.126. What would your NAT configuration be on the border router?



Another NAT example

The command ip nat pool Todd 192.1.2.109 192.1.2.109 netmask 255.255.255.248 sets the pool name as Todd and creates a dynamic pool of only one address using NAT address 192.1.2.109.
Instead of the netmask command, you can use the prefix-length 29 statement.

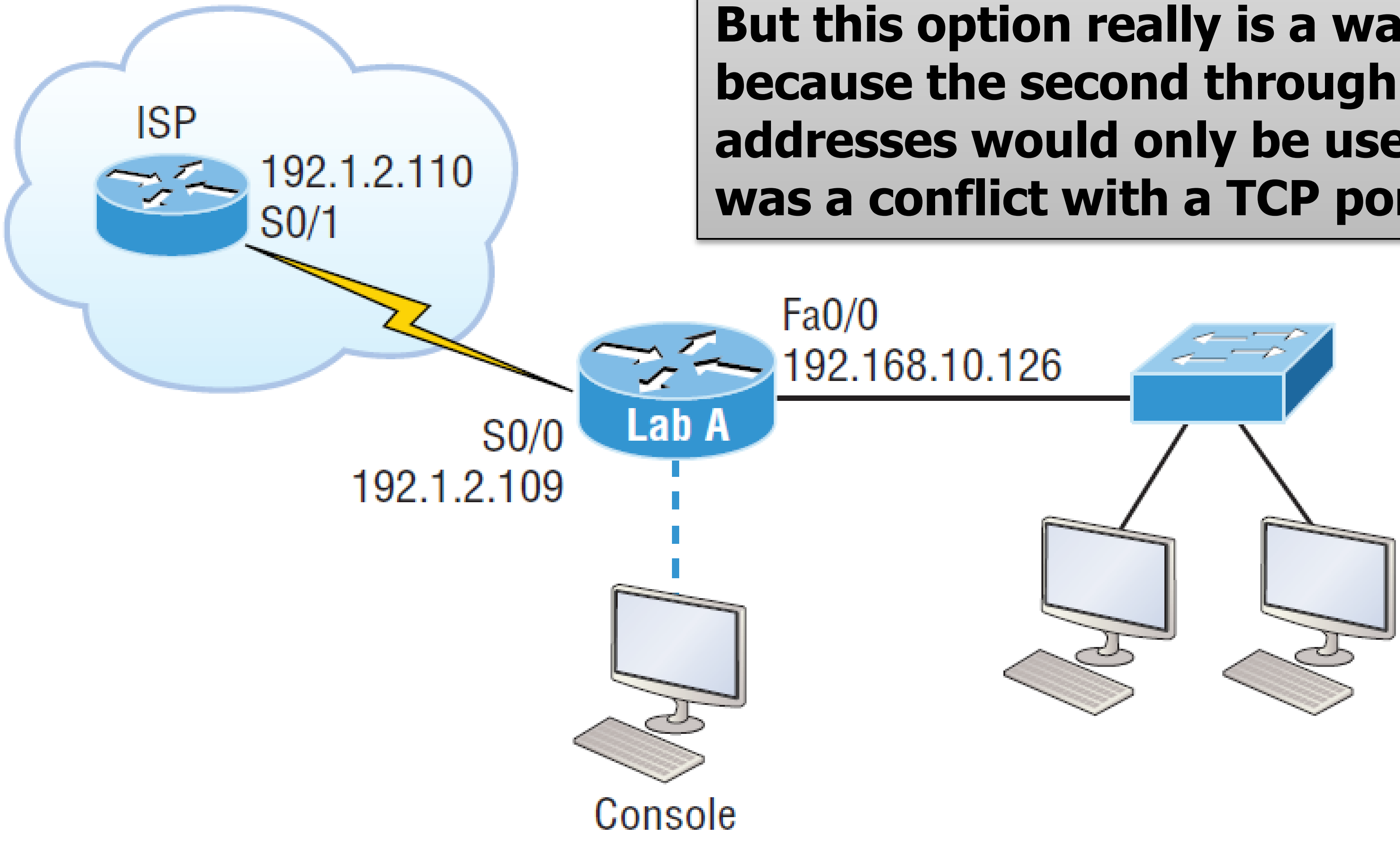


```
ip nat pool Todd 192.1.2.109 192.1.2.109 netmask 255.255.255.248
access-list 1 permit 192.168.10.64 0.0.0.63
ip nat inside source list 1 pool Todd overload
```

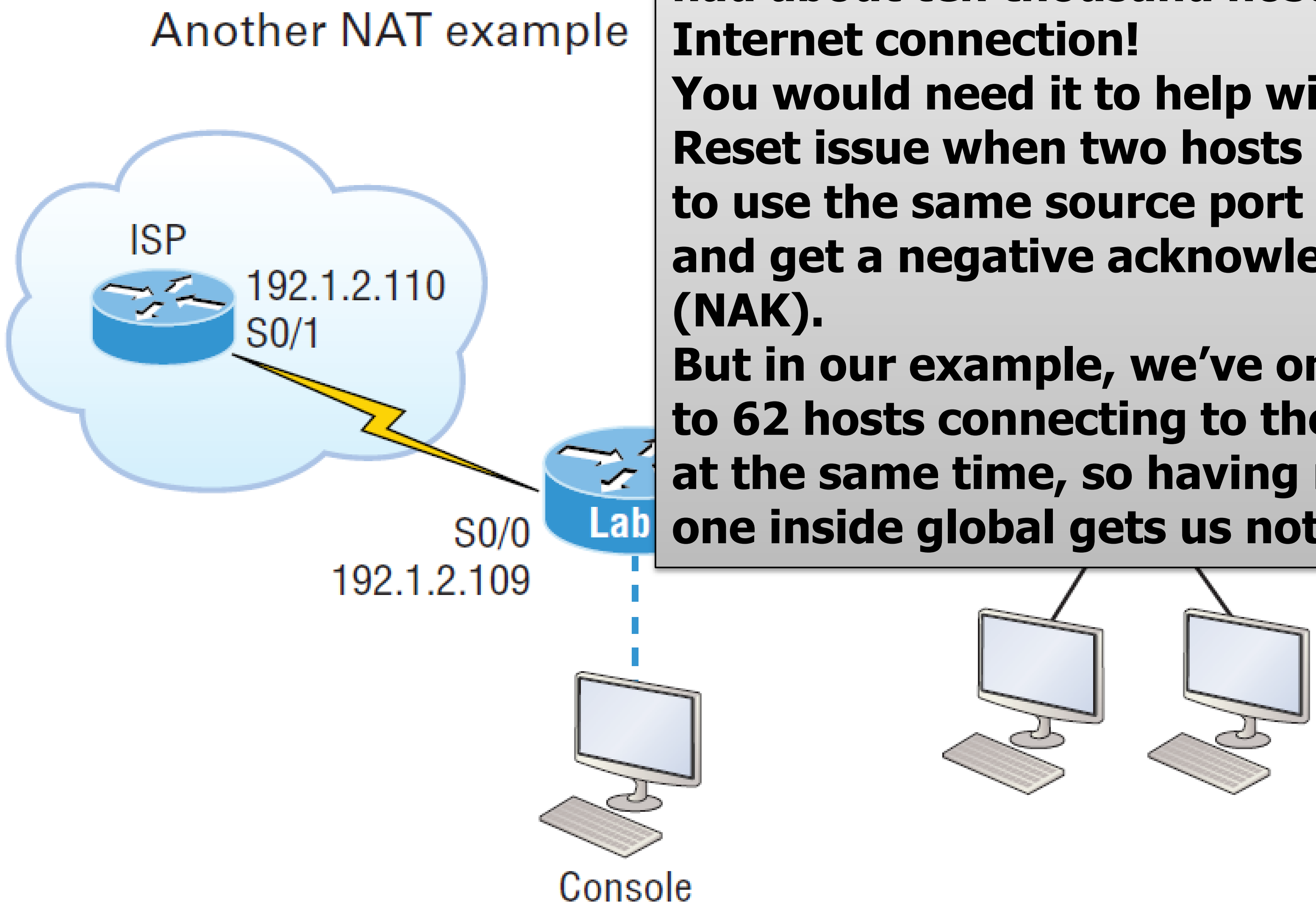
second answer

having only 192.1.2.109 as your inside global, but you can type this in and it will also work: ip nat pool Todd 192.1.2.109 192.1.2.114 netmask 255.255.255.248.
But this option really is a waste because the second through sixth addresses would only be used if there was a conflict with a TCP port number.

Another NAT example



second answer

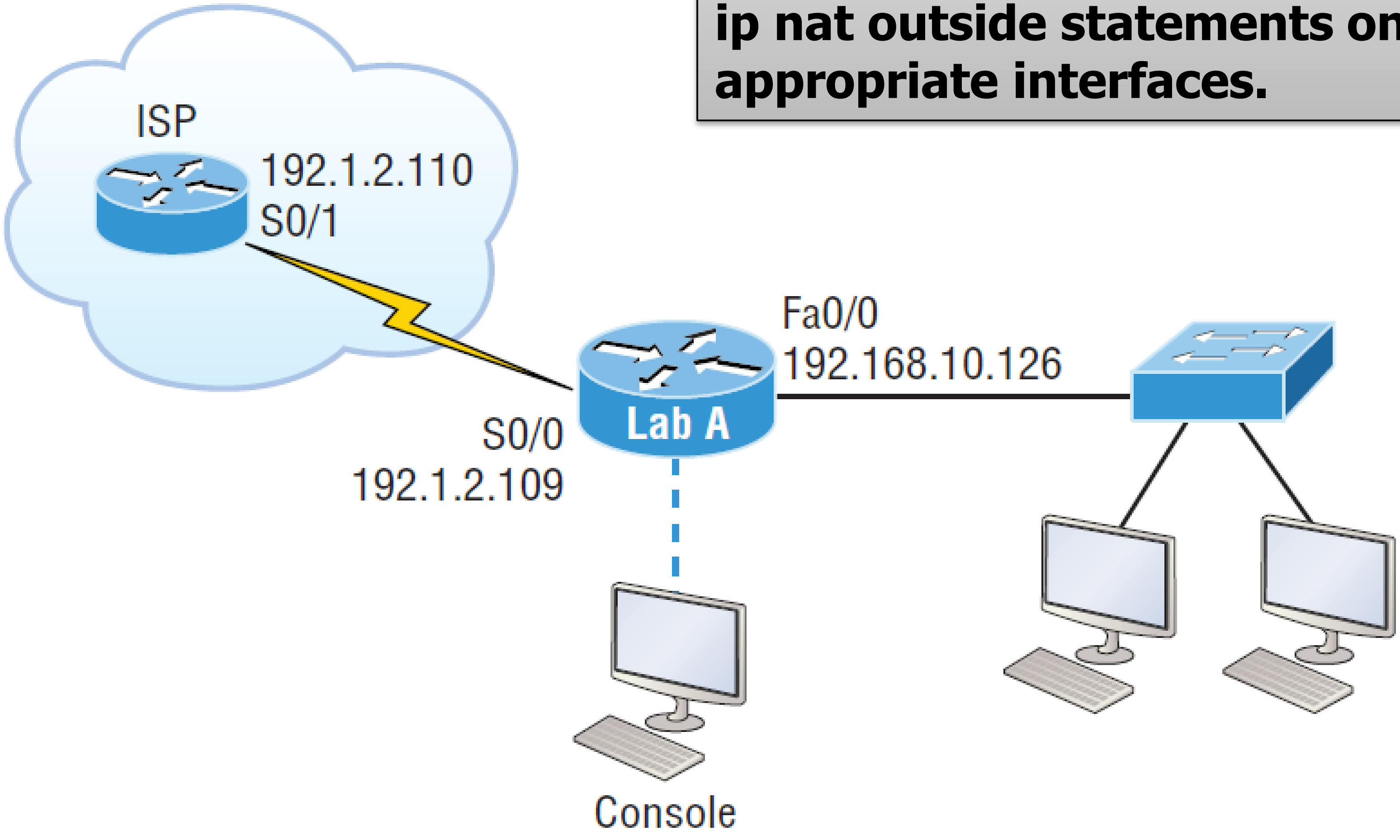


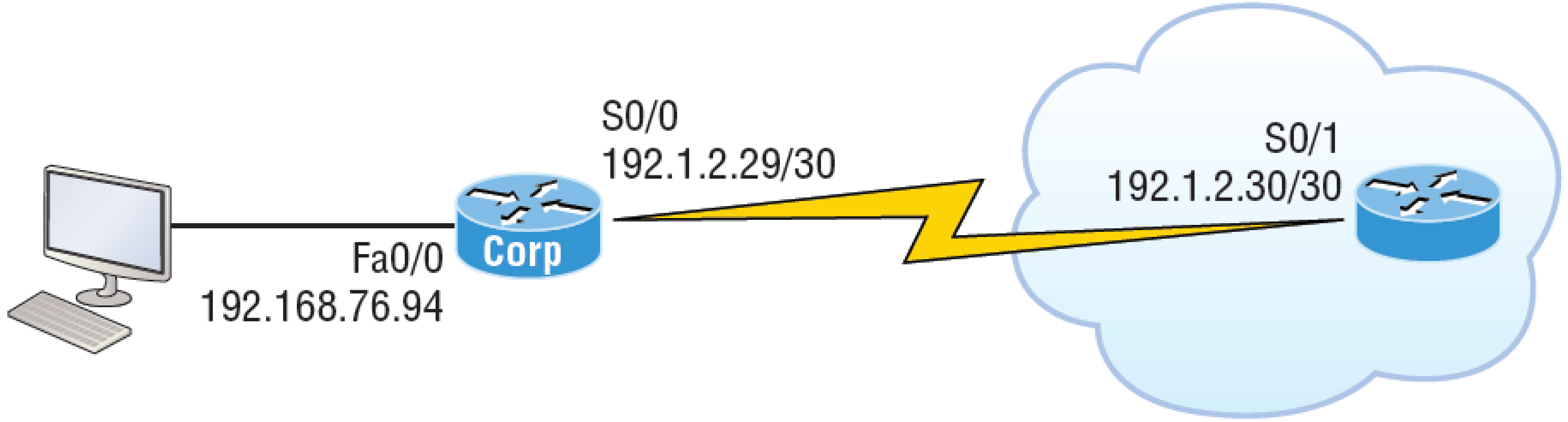
You would use something like what shown in this example if you literally had about ten thousand hosts with one Internet connection!
You would need it to help with the TCP-Reset issue when two hosts are trying to use the same source port number and get a negative acknowledgment (NAK).
But in our example, we've only got up to 62 hosts connecting to the Internet at the same time, so having more than one inside global gets us nothing!

second answer

The command ip nat inside source list 1 pool Todd overload sets the dynamic pool to use PAT by using the overload command.
And be sure to add the ip nat inside and ip nat outside statements on the appropriate interfaces.

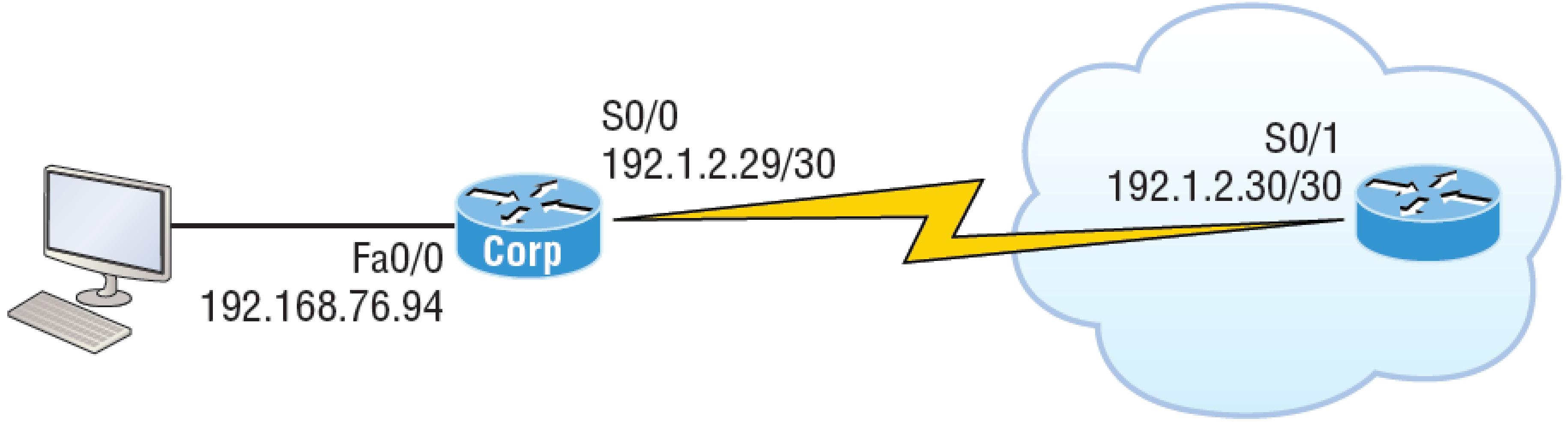
Another NAT example





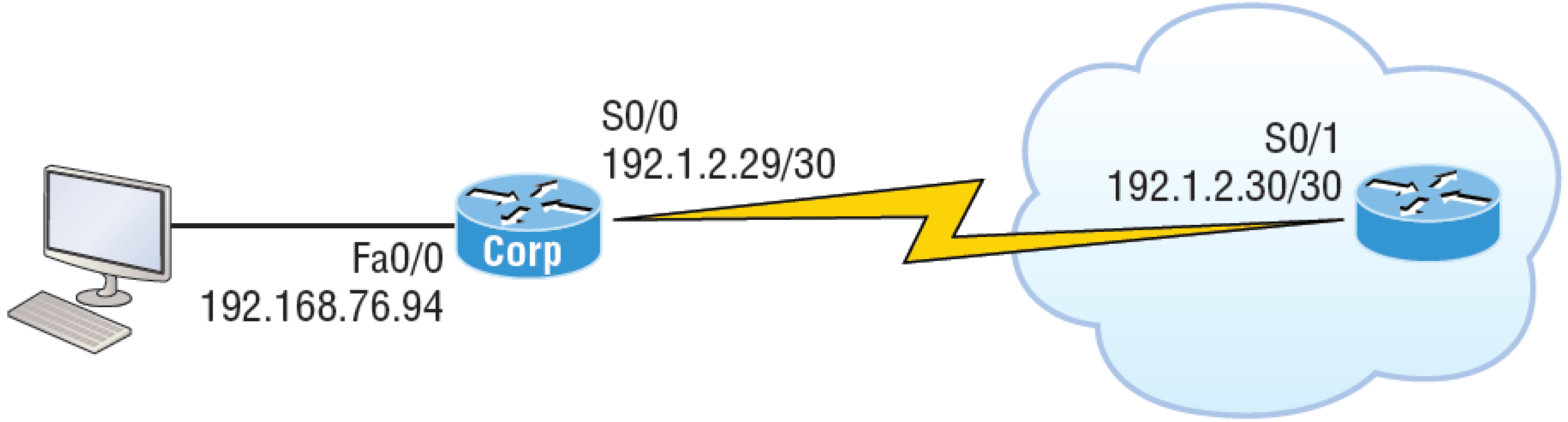
The network is already configured with IP addresses and there is only one configured host.
you need to add 25 more hosts to the LAN.
Now, all 26 hosts must be able to get to the Internet at the same time.
By looking at the configured network, use only the following inside addresses to configure NAT on the Corp router to allow all hosts to reach the Internet:

- **Inside globals: 198.18.41.129 through 198.18.41.134**
- **Inside locals: 192.168.76.65 through 192.168.76.94**



we must first determine what our block sizes are so we can get our subnet mask for our NAT pool.
This will also equip us to configure the wildcard for the access list.
You should easily be able to see that the block size of the inside globals is 8 and the block size of the inside locals is 32.
Know that it's critical not to stumble on this foundational information!
so we can configure NAT now that we have our block sizes:

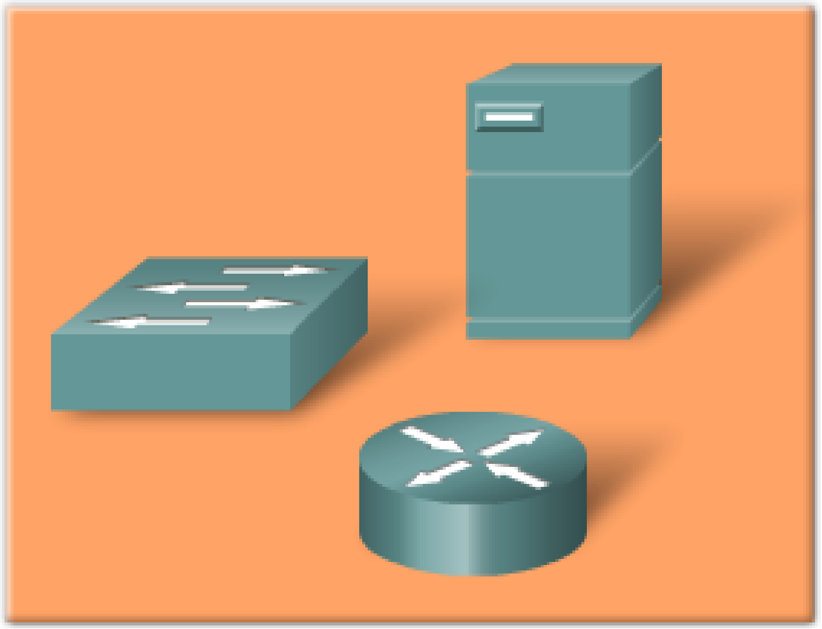
```
ip nat pool Corp 198.18.41.129 198.18.41.134 netmask 255.255.255.248
ip nat inside source list 1 pool Corp overload
access-list 1 permit 192.168.76.64 0.0.0.31
```



Since we had a block of only 8 for our pool, we had to use the overload command to make sure all 26 hosts can get to the Internet at the same time.
There is one other simple way to configure NAT, and I use this command at my home office to connect to my ISP. One command line and it's done! Here it is:

```
ip nat inside source list 1 int s0/0/0 overload
```

Manual Configuration

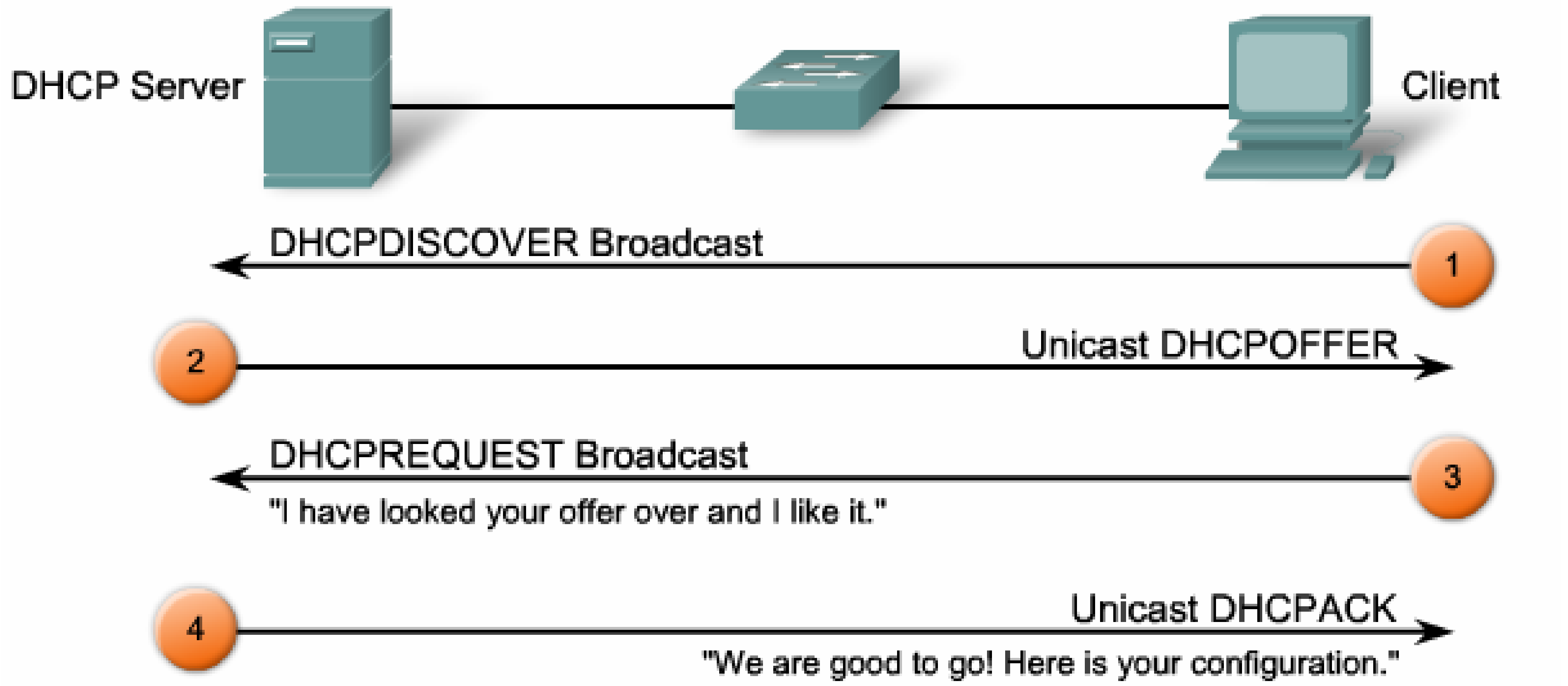


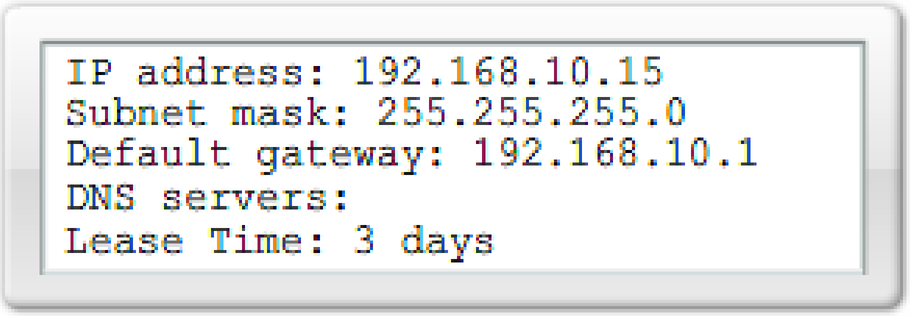
Network devices that remain in the same place (logically and physically) are assigned static IP addresses.

Dynamic Configuration

Network devices that are added, moved or changed (physical and logical) need new addresses. Manual configuration is unwieldy.

DHCP Operation



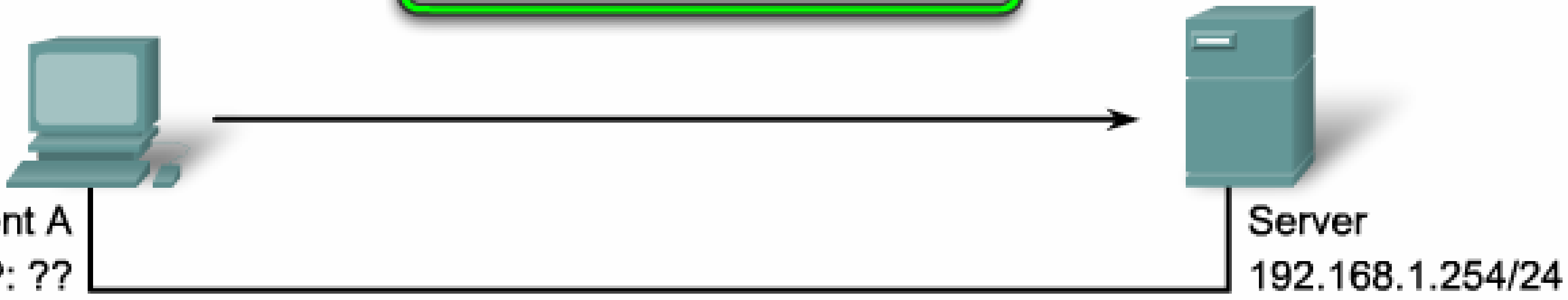


```
IP address: 192.168.10.15
Subnet mask: 255.255.255.0
Default gateway: 192.168.10.1
DNS servers:
Lease Time: 3 days
```

DHCP Message Format

8	16	24	32
OP Code (1)	Hardware type (1)	Hardware address length (1)	Hops (1)
Transaction Identifier			
Seconds – 2 bytes		Flags – 2 bytes	
Client IP Address (CIADDR) – 4 bytes			
Your IP Address (YIADDR) – 4 bytes			
Server IP Address (SIADDR) – 4 bytes			
Gateway IP Address (GIADDR) – 4 bytes			
Client Hardware Address (CHADDR) – 16 bytes			
Server name (SNAME) – 64 bytes			
Filename – 128 bytes			
DHCP Options – variable			

DHCP Discover

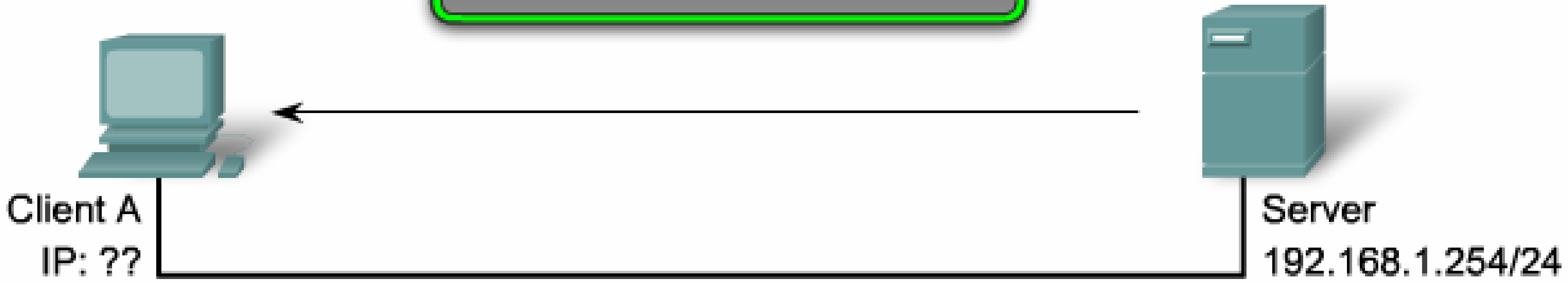


Ethernet Frame	IP	UDP	DHCPDISCOVER
SRC MAC: MAC A DST MAC: FF:FF:FF:FF:FF:FF	IP SRC: ? IP DST: 255.255.255.255	UDP 67	CIADDR: ? GIADDR: ? Mask:? CHADDR: MAC A

MAC: Media Access Control Address
CIADDR: Client IP Address
GIADDR: Gateway IP Address
CHADDR: Client Hardware Address

The DHCP Client sends a directed IP broadcast, with a DHCP discover packet. In the simplest case, there is a DHCP server on the same segment, which will pick up this request. The server notes the GIADDR field is blank, so the client is on the same segment. The server also notes the hardware address of the client in the request packet.

DHCP Offer



Ethernet Frame	IP	UDP	DHCP Reply	
SRC MAC: MAC Serv DST MAC: MAC A	IP SRC: 192.168.1.254 IP DST: 192.168.1.10	UDP 68	CIADDR: 192.168.1.10 Mask: 255.255.255.0	GIADDR: ? CHADDR: MAC A

MAC: Media Access Control Address
CIADDR: Client IP Address
GIADDR: Gateway IP Address
CHADDR: Client Hardware Address

The DHCP server picks an IP address from the available pool for that segment, as well as the other segment and global parameters. It puts them into the appropriate fields of the DHCP packet. It then uses the hardware address of A (in CHADDR) to construct an appropriate frame to send back to the client.