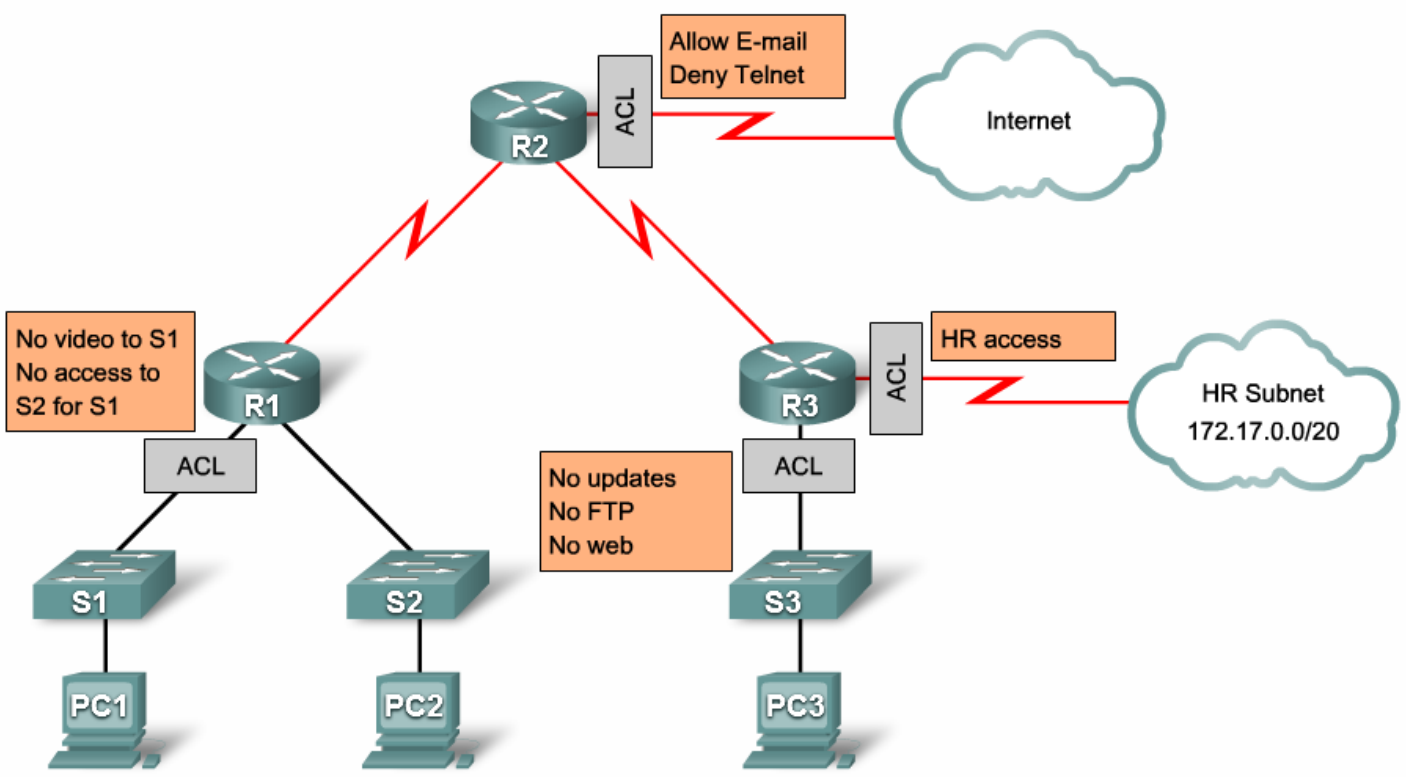
Module 7

Managing IP Traffic with Access Lists and NAT

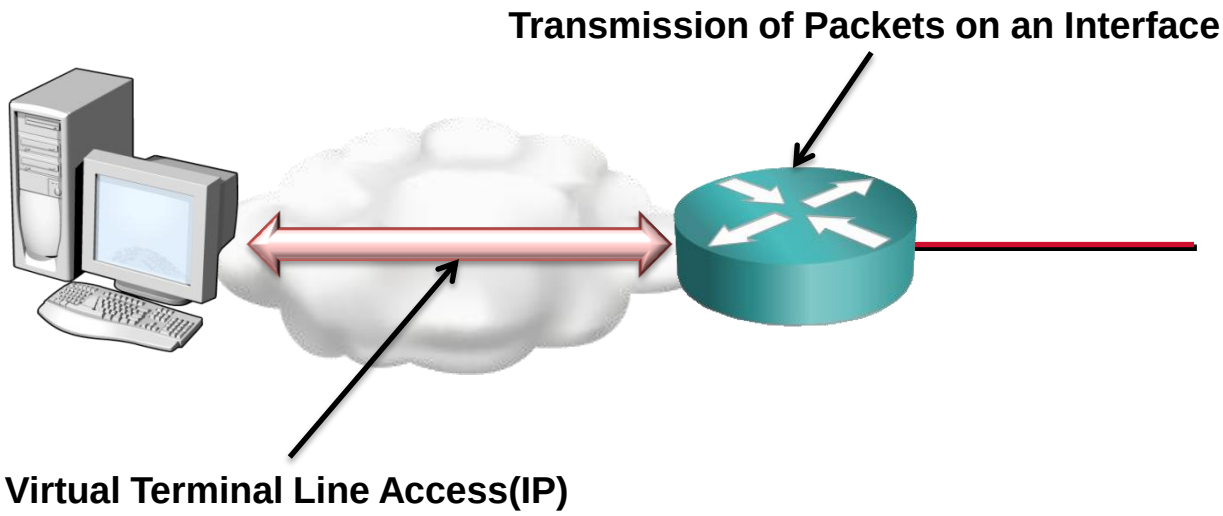
Access List and Their Applications

Why Use Access Lists ?



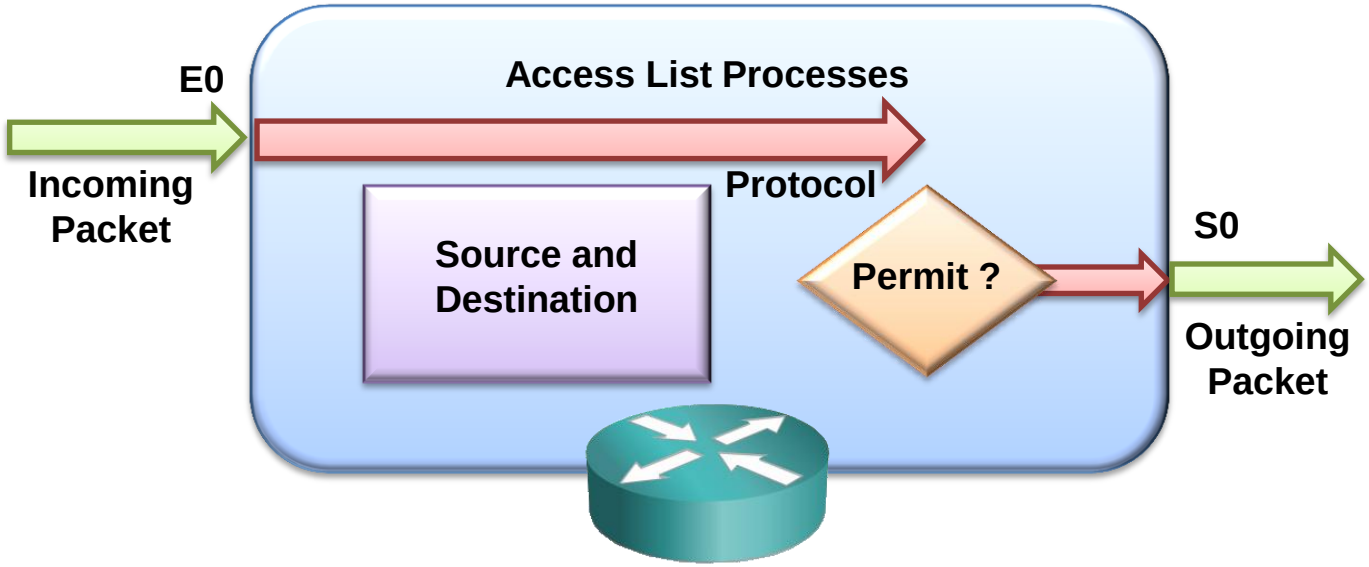
- Router에서는 ACL(Access Control List)을 사용하여 트래픽 식별, 필터링, 암호화, 분류, 변환 작업을 수행할 수 있다
- Router를 경유하는 Packet을 Filtering한다

Access List Applications



- Packet Filtering을 활용하여 네트워크에서의 Packet 이동을 제어할 수 있다
- Router에 VTY 포트로 들어오거나 VTY 포트에서 나가는 Telnet Traffic을 허용(Permit) 하거나 거부(Deny)할 수 있다

Types of Access Lists



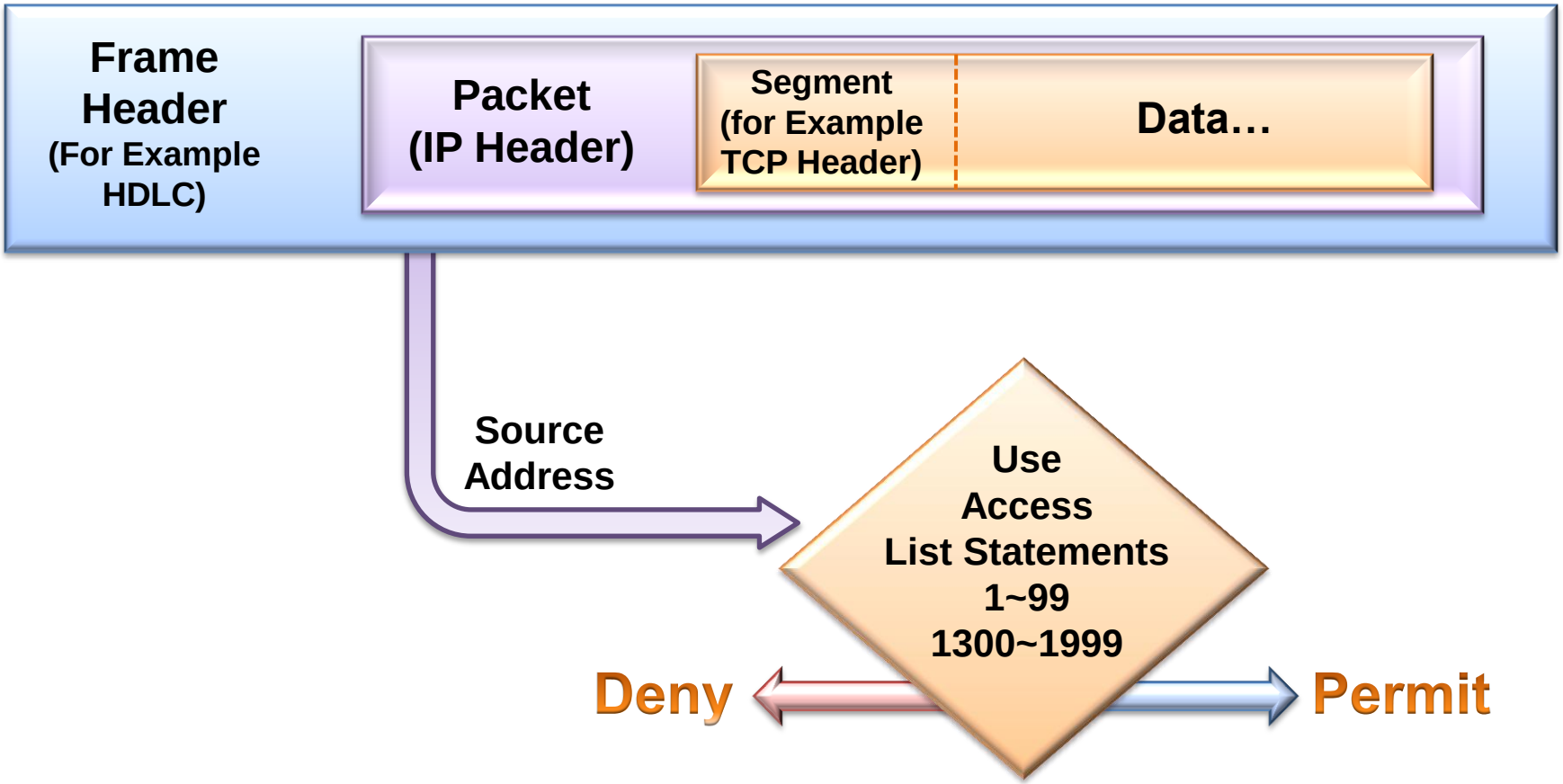
- **Standard Access List :**
  - Source Address를 검사한다
  - 검사결과에 따라 전체 Protocol Suite에 대한 Packet 출력을 허용하거나 거부한다
- **Extended Access List :**
  - Source Address와 Destination Address를 모두 검사한다
  - 특정 Protocol, Port번호, 다른 매개변수를 검사하여 유연하게 제어가 가능하다

How to Identify Access Lists

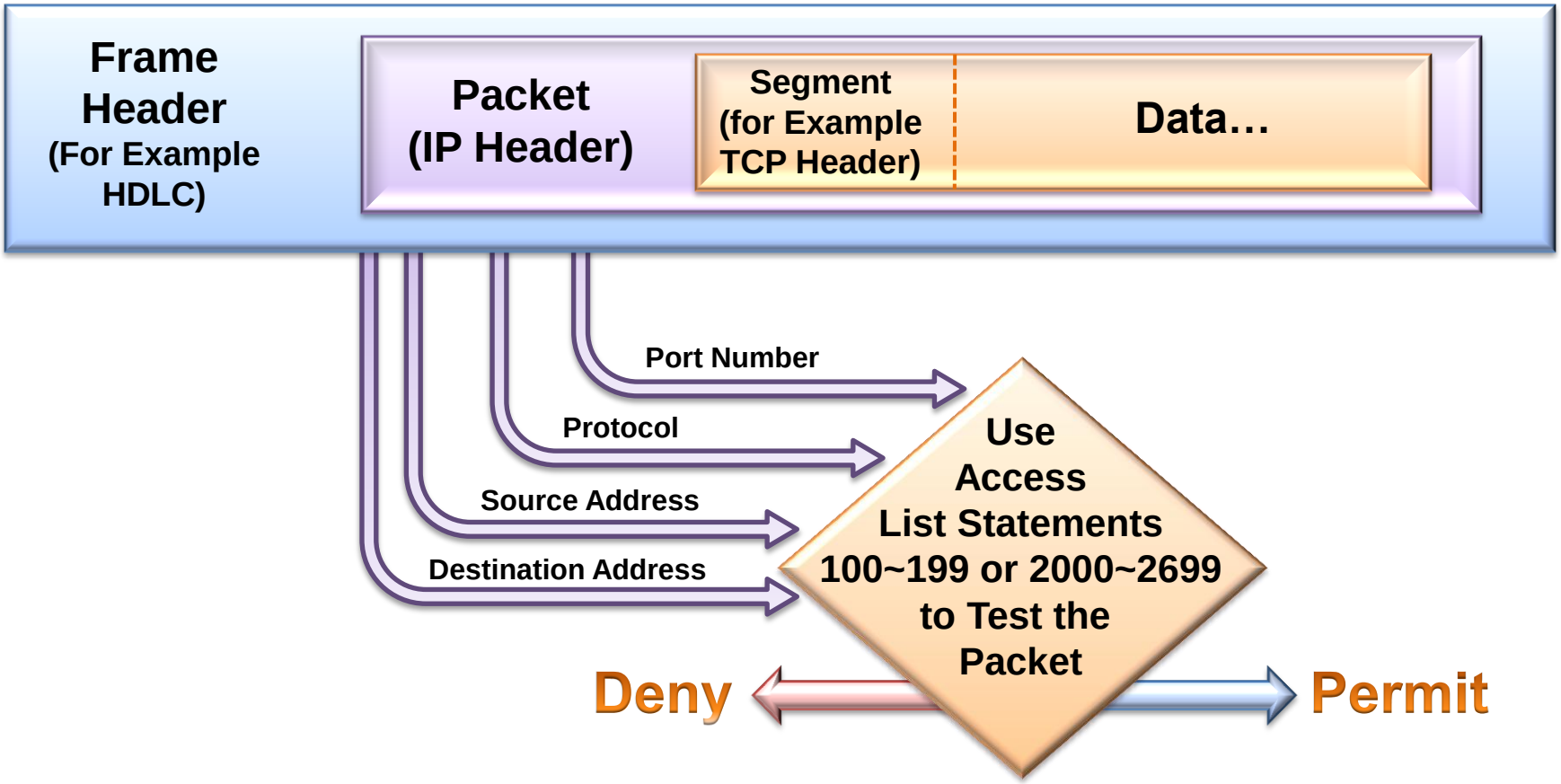
Access List Type		Number Range/Identifier
IP	Standard	1~99, 1300~1999
	Extended	100~199, 2000~2699
	Named	Name

- Standard IP List (1~99)는 IP Packet에 Source Address를 조건으로 가진다
- Extended IP List (100~199)는 Source and Destination Address와 특정 TCP/IP Protocol Suite Protocol과 Destination Port를 조건으로 가진다
- Standard List (1300~1999) (Expanded Range)
- Extended IP List (2000~2699) (Expanded Range)
- Named IP List는 Standard와 Extended 이름으로 선어하여 각 조건을 검사한다

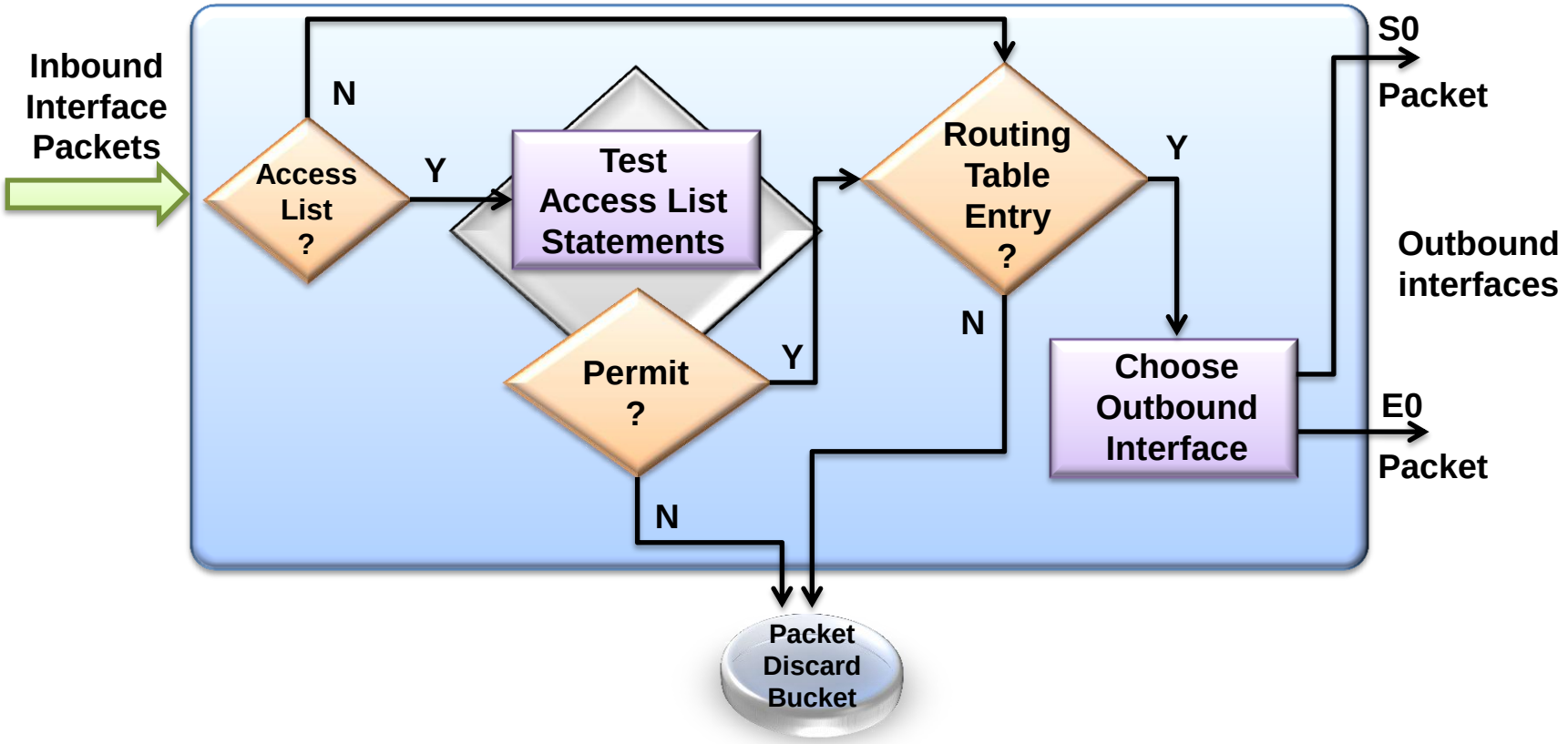
Testing Packets with Standard Access Lists



Testing Packets with Extended Access Lists

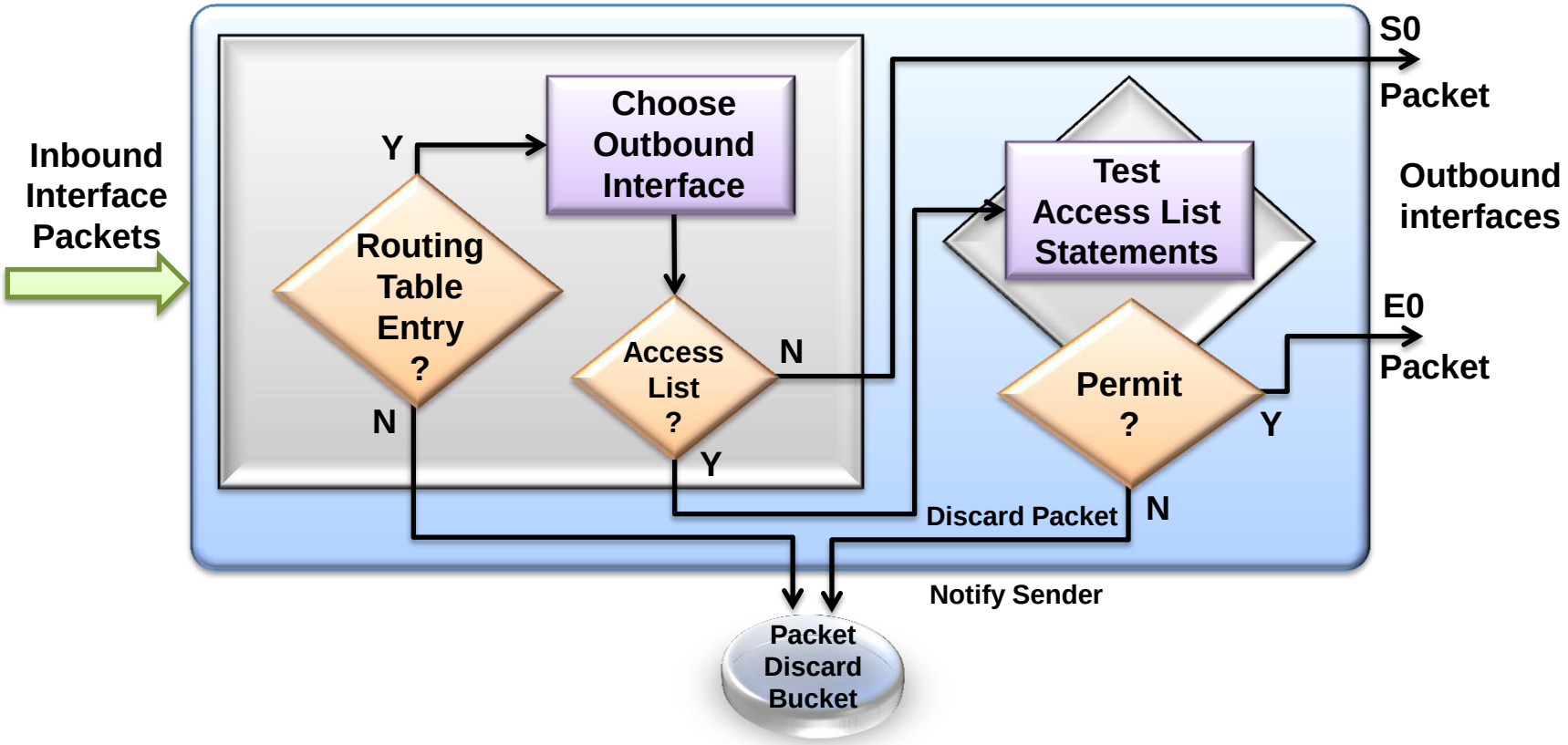


Inbound ACL Operation



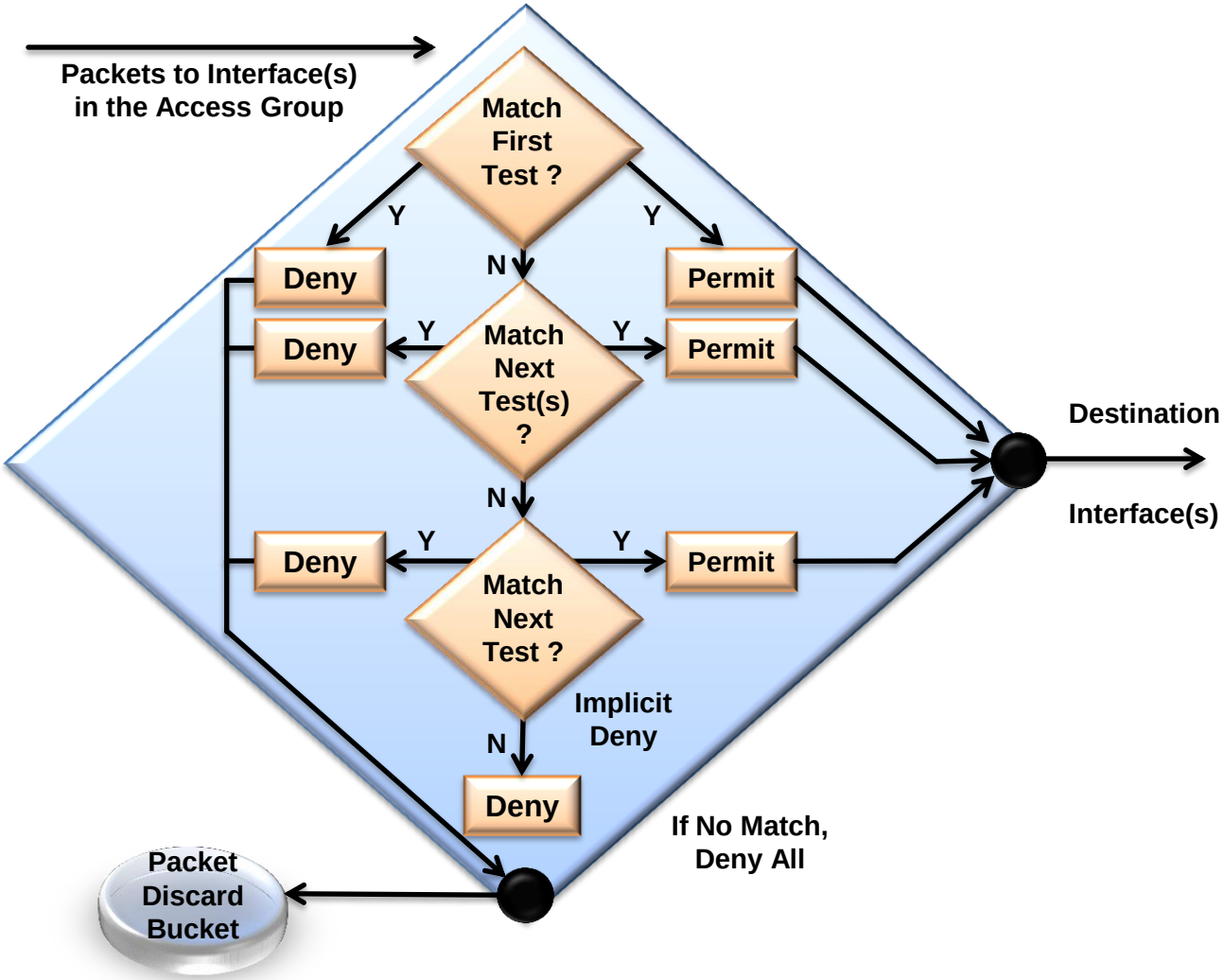
- Access List에 매치되지 않는 모든 Packet은 암시적으로 거부된다

Outbound ACL Operation

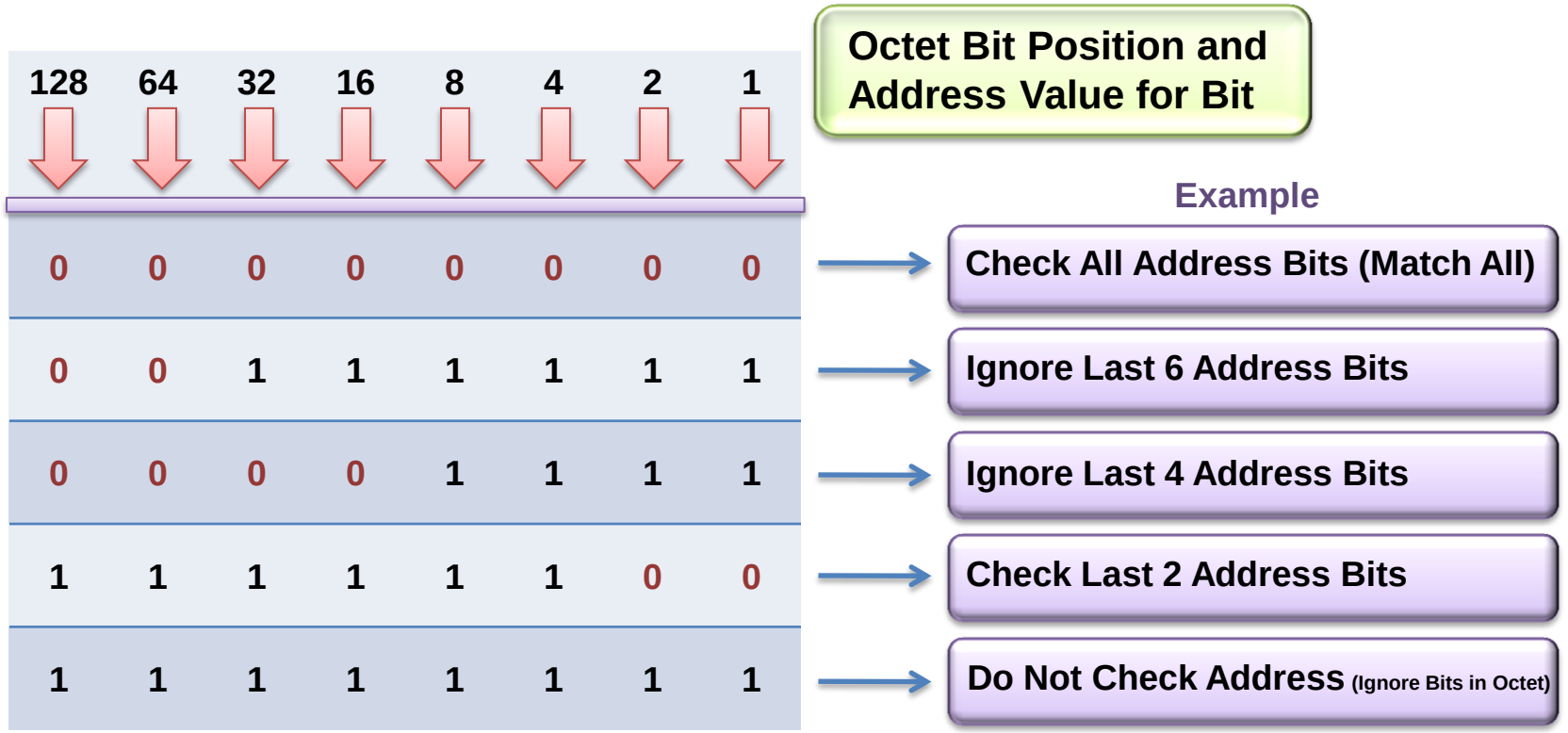


- Access List에 매치되지 않는 모든 Packet은 암시적으로 거부된다

A List of Tests: Deny or Permit



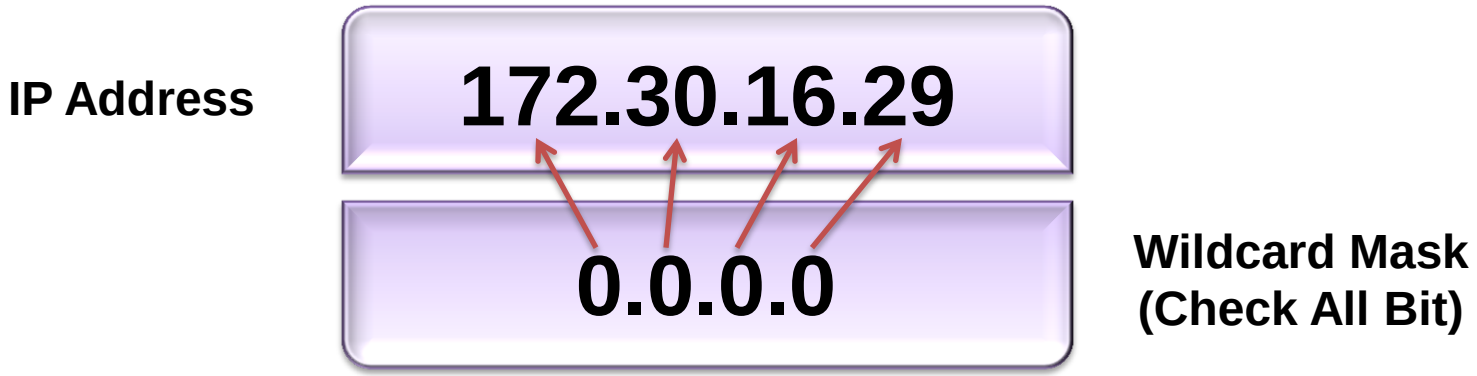
Wildcard Bits : How to Check the Corresponding Address Bits



- Wildcard mask bit 0은 대응 bit값을 검사하라는 것을 의미한다
- Wildcard mask bit 1은 대응 bit값을 검사하지 말고 무시하라는 것을 의미한다

Wildcard Bits to Match a Specific IP Host Address

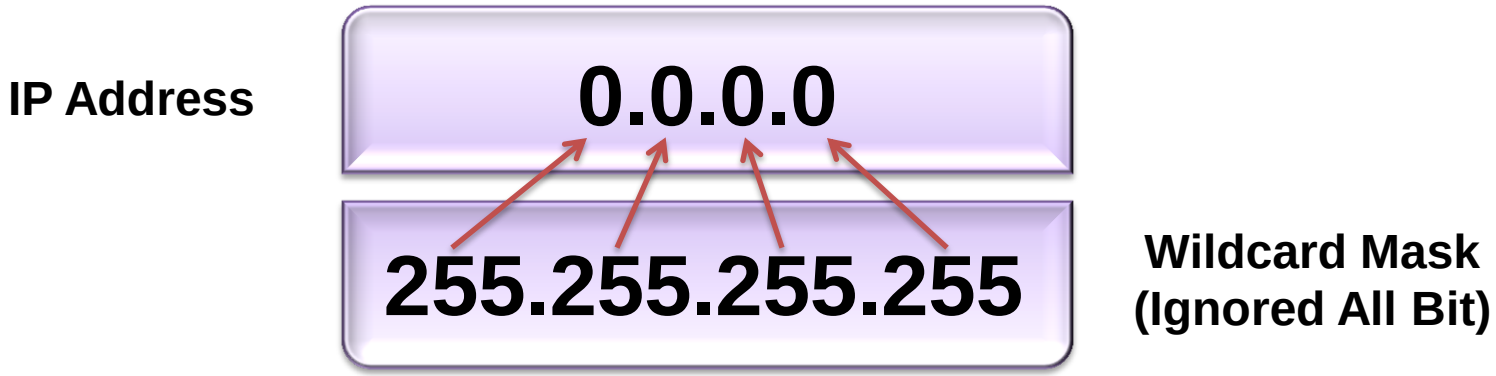
- Test 조건 : 모든 Address Bit 검사 (모두 일치)
  (1개의 IP Host Address만 검사)



- 172.30.16.29 0.0.0.0은 모든 Address를 검사해서 매치되는 주소 즉, 172.30.16.29 IP를 갖는 호스트를 지정한다
- 하나의 IP를 알리기 위해 IP Address 앞에 약어 host를 사용할 수 있다. 예를 들어 “172.30.16.29 0.0.0.0” 대신 “host 172.20.16.29”를 사용할 수도 있다

Wildcard Bits to Match Any IP Address

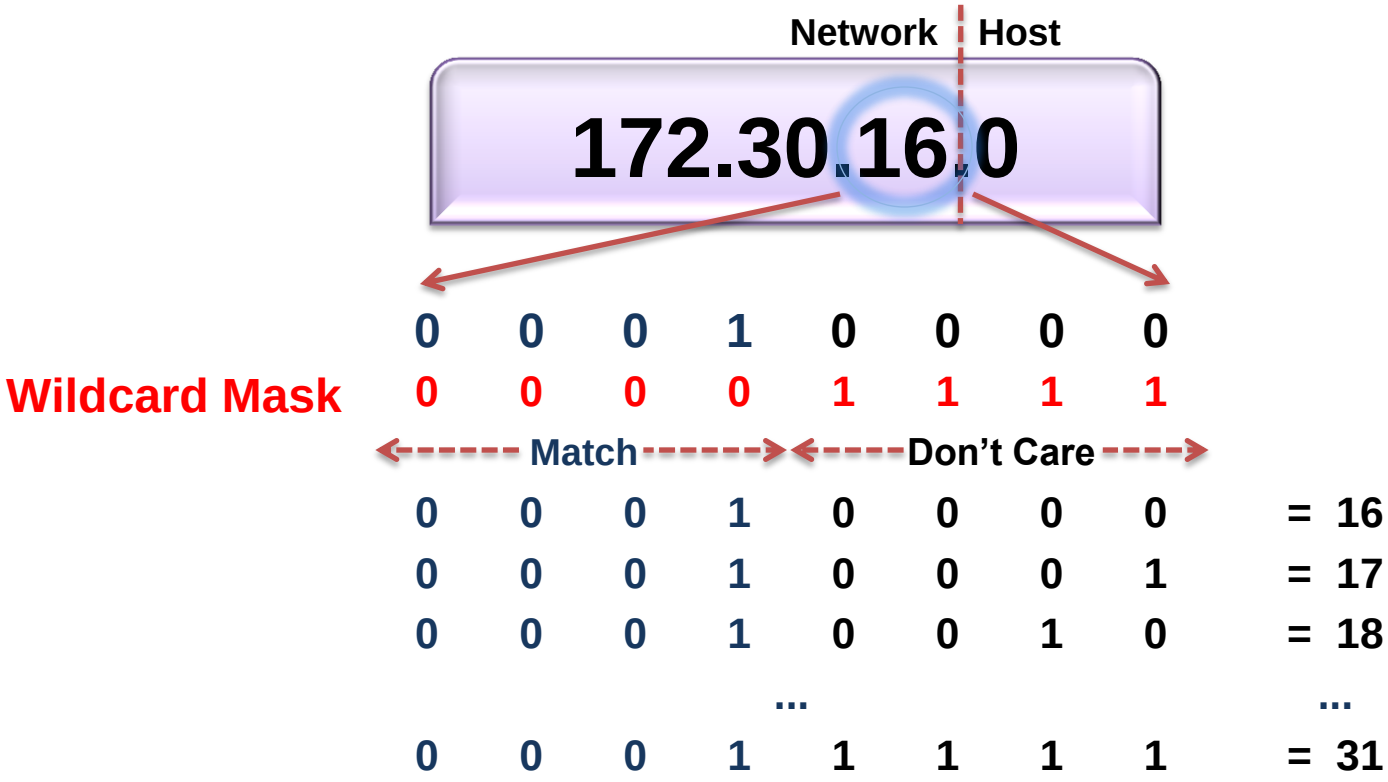
- Test 조건 : 모든 Address Bit 무시 (Match Any)
  (모드 IP Address)



- 모든 Address를 받아들이려면 IP Address는 0.0.0.0을 입력하고 Wildcard mask는 모든 값을 무시(검사 없이 허용)하려면 255.255.255.255를 지정한다
- 관리자는 모든 주소를 지정할 목적으로 0.0.0.0 255.255.255.255를 명시하는 대신 any라는 문자를 사용할 수 있다

Wildcard Bit to Match IP Subnets

- 172.30.16.0/24에서 172.30.31.0/24까지의 IP Subnet 검사하기
  - Address and wildcard mask : 172.30.16.0 0.0.15.255



Configuring IP Access Lists

Access List Command Overview

- **Step 1 : access-list 명령어로 IP Traffic Filter list에 Entry를 만든다**

```
Router(config)#access-list access-list-number  {permit | deny}
{test_conditions}
```

- **Step 2 : IP access-group 명령으로 기존 Access-List를 Interface에 적용한다**

```
Router(config-if)#{protocol} access-group access-list-number
{in | out}
```

Standard IP Access List Configuration

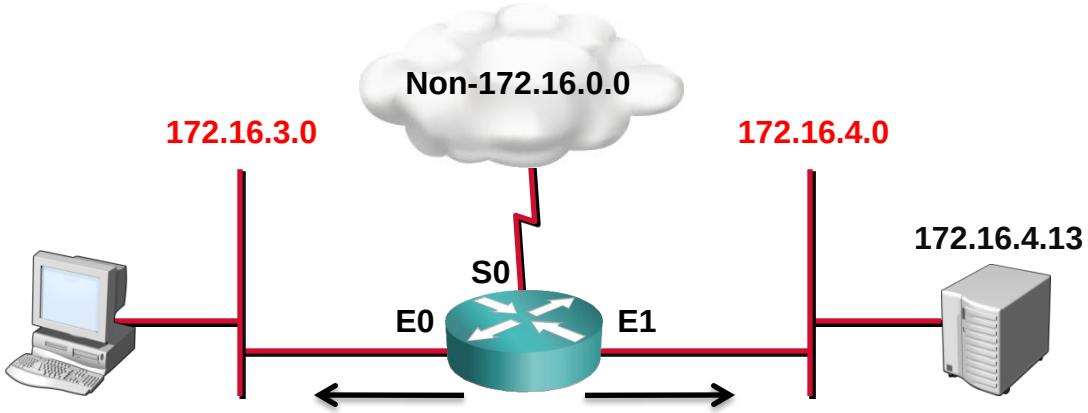
```
Router(config)#access-list access-list-number {permit | deny} source [mask]
```

- Access-list-number : Entry가 속할 list 번호 설정. 1~99, 1300~1999사이의 번호가 들어간다
- permit | deny | remark는 해당 Entry에 매치되면 취할 Action을 정의
- Source는 송신지 IP Address를 정의한다
- mask는 Wildcard mask를 사용하여 Address 필드의 어느 비트들이 일치되어야 하는지 설정한다

```
Router(config-if)#ip access-group access-list-number  {in | out}
```

- List를 적용할 Interface에서 설정한다
- Inbound또는 Outbound시 검사하도록 설정한다
- Default = outbound
- Interface에서 “no ip access-group *access-list-number*”명령을 사용하여 적용된 Access-list를 제거한다

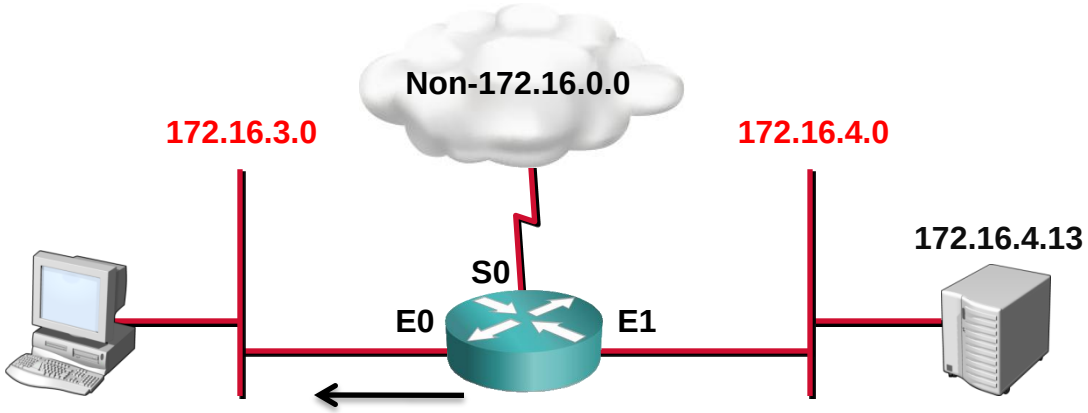
Standard IP Access List Example 1




```
Router(config)#access-list 1 permit 172.16.0.0 0.0.255.255
(implicit deny all – not visible in the list)
(access-list 1 deny 0.0.0.0 255.255.255.255)

Router(config)#interface ethernet 0
Router(config-if)#ip access-group 1 out
Router(config)#interface ethernet 1
Router(config-if)#ip access-group 1 out
```

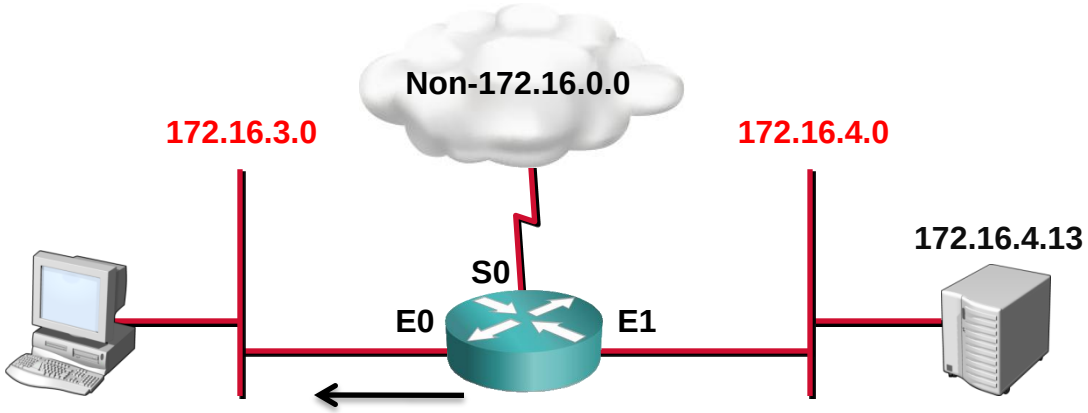
Standard IP Access List Example 2




```
Router(config)#access-list 1 deny 172.16.4.13 0.0.0.0
Router(config)#access-list 1 permit 0.0.0.0 255.255.255.255
(implicit deny all)
(access-list 1 deny 0.0.0.0 255.255.255.255)

Router(config)#interface ethernet 0
Router(config-if)#ip access-group 1 out
```

Standard IP Access List Example 3




```
Router(config)#access-list 1 deny 172.16.4.0 0.0.0.255
Router(config)#access-list 1 permit any
(implicit deny all)
(access-list 1 deny 0.0.0.0 255.255.255.255)

Router(config)#interface ethernet 0
Router(config-if)#ip access-group 1 out
```

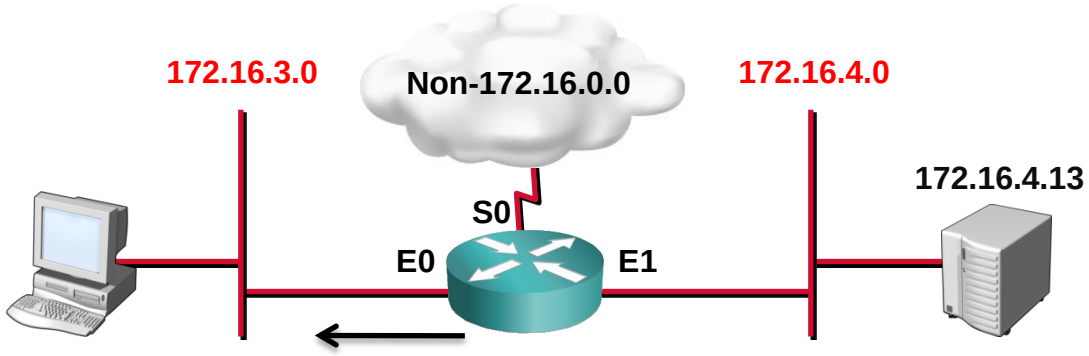
Extended IP Access List Configuration

```
Router(config)#access-list access-list-number
{permit | deny} protocol source source-wildcard [operator port] destination
destination-wildcard [operator port] [established] [log]
```

- Access-list-number : Entry가 속할 list 번호 설정. 100~199, 2000~2699사이의 번호가 들어간다
- permit | deny | remark는 해당 Entry에 매치되면 취할 Action을 정의
- Source와 Destination은 송수신지 IP Address를 정의한다
- mask는 Wildcard mask를 사용하여 Address 필드의 어느 비트들이 일치되어야 하는지 설정한다
- Operator port는 lt(less then), gt(greater than), eq(equal to), neq(not equal to)와 Protocol Port번호를 명시한다
- established는 Inbound TCP에 대해서만 사용된다
- log는 Console로 log Message를 보낸다

```
Router(config-if)#ip access-group access-list-number {in | out}
```

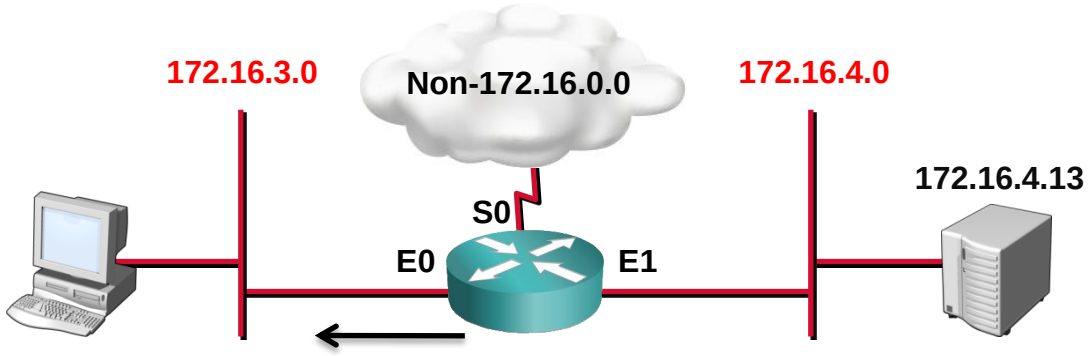
Extended IP Access List Example 1




```
Router(config)#access-list 101 deny tcp 172.16.4.0 0.0.0.255 172.16.3.0 0.0.0.255 eq 21
Router(config)#access-list 101 deny tcp 172.16.4.0 0.0.0.255 172.16.3.0 0.0.0.255 eq 20
Router(config)#access-list 101 permit ip any any
(implicit deny all)
(access-list 101 deny 0.0.0.0 255.255.255.255 0.0.0.0 255.255.255.255)
Router(config)#interface ethernet 0
Router(config)#ip access-group 101 out
```

- deny list는 172.16.4.0 Subnet에서 172.16.3.0 subnet으로 가는 FTP Traffic을 거부한다
- permit은 다른 모든 IP Traffic이 E0 Interface로 나가는 것을 허용한다

Extended IP Access List Example 2




```
Router(config)#access-list 101 deny tcp 172.16.4.0 0.0.0.255 any eq 23
Router(config)#access-list 101 permit ip any any
(implicit deny all)
(access-list 101 deny 0.0.0.0 255.255.255.255 0.0.0.0 255.255.255.255)

Router(config)#interface ethernet 0
Router(config)#ip access-group 101 out
```

- deny list는 172.16.4.0 Subnet에서 E0 Interface로 나가는 Telnet Traffic을 거부한다
- permit은 다른 모든 IP Traffic이 E0 Interface로 나가는 것을 허용한다

Named IP Access List ?

- **Named IP Access list 고려사항**
  - Named IP Access list는 IOS 11.2 이전 version에서는 호환되지 않는다
  - 여러 개의 액세스 리스트에 같은 이름을 사용할 수 없다
- **Named IP Access list 생성 단계**
  - Named IP Access-list Mode로 이동한다

```
Router(config)#ip access-list {stanard | extended} name
```

  - Test 조건을 입력한다

```
Router(config-{std|ext})#{permit|deny} {test conditions}
Router(config-{std|ext})#no {permit|deny} {test conditions}
```

  - 해당 Access-list를 Interface에 적용하기

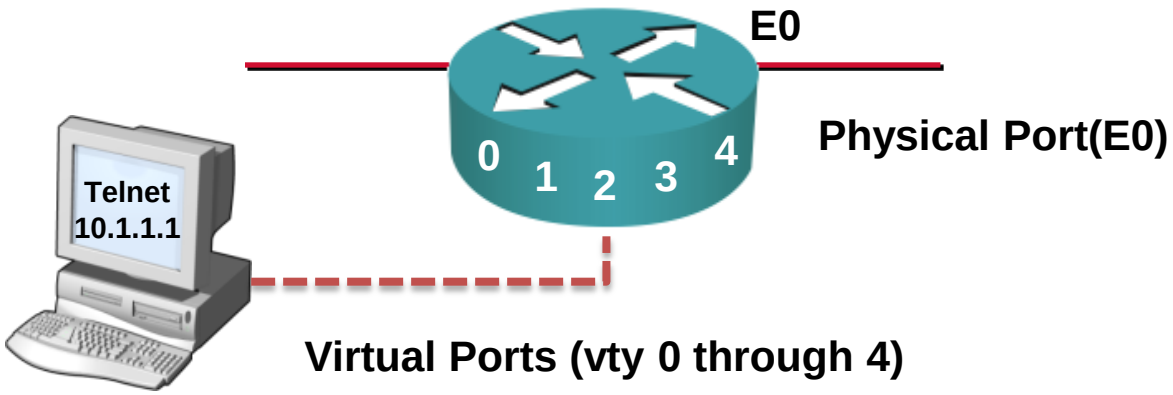
```
Router(config-if)#ip access-group name {in | out}
```

Using Named IP Access Lists

```
Router(config)#ip access-list extended screen
Router(configext-nacl)#deny tcp 172.16.4.0 0.0.0.255 172.16.3.0 0.0.0.255 eq 23
Router(configext-nacl)#permit ip any any
Router(config)#interface ethernet 0
Router(config-if)#ip access-group screen out
```

- deny는 172.16.4.0 Subnet에서 E0 Interface로 나가는 Telnet Traffic을 거부하는 Named Access list이다
- Permit은 다른 모든 IP Traffic이 E0 Interface로 나가는 것을 허용한다

How to Control vty Access



- VTY는 라우터에 Telnet 접속을 위해 할당된 가상 포트이다
- Interface를 경우해서 지나가는 트래픽이 아니기 때문에 Interface에서 제어할 수 없으므로 line vty 0 4에서 제어한다

vty Commands

- 접속 제어할 포트 번호를 활성화 한다

```
Router(config)#line vty {vty# | vty-range}
Router(config)#
```

- 적용할 Access-list를 적용한다

```
Router(config-line)#access-class access-list-number {in | out}
Rotuer(config-line)#
```

vty Access Example

- **Controlling Inbound Access**

```
Router(config)#access-list 12 permit 192.168.1.0 0.0.0.255
(implicit deny all)

Router(config)#line vty 0 4
Router(config-line)#access-class 12 in
Router(config-line)#
```

- **192.168.1.0/24 Subnet에 해당하는 IP Address를 갖는 호스트만 접속을 허용한다**

Monitoring Access List Statements

```
Router#show {protocol} access-list {access-list number}
```

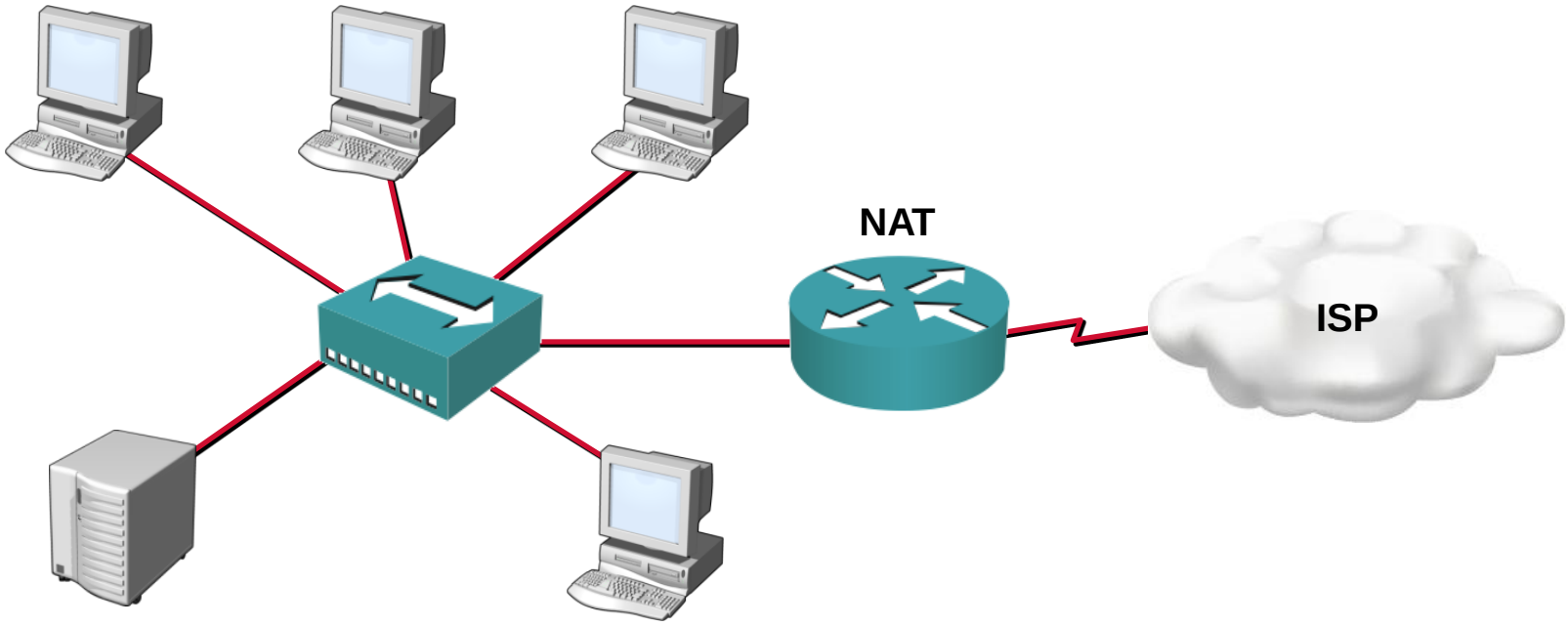
```
Router#show access-list {access-list number}
```

```
Router#show access-lists
Standard IP access list 1
      permit 10.2.2.1
      permit 10.3.3.1
      permit 10.4.4.1
      permit 10.5.5.1
Extended IP access list 101
      permit tcp host 10.22.22.1 any eq telnet
      permit tcp host 10.33.33.1 any eq ftp
      permit tcp host 10.44.44.1 any eq ftp-data
```

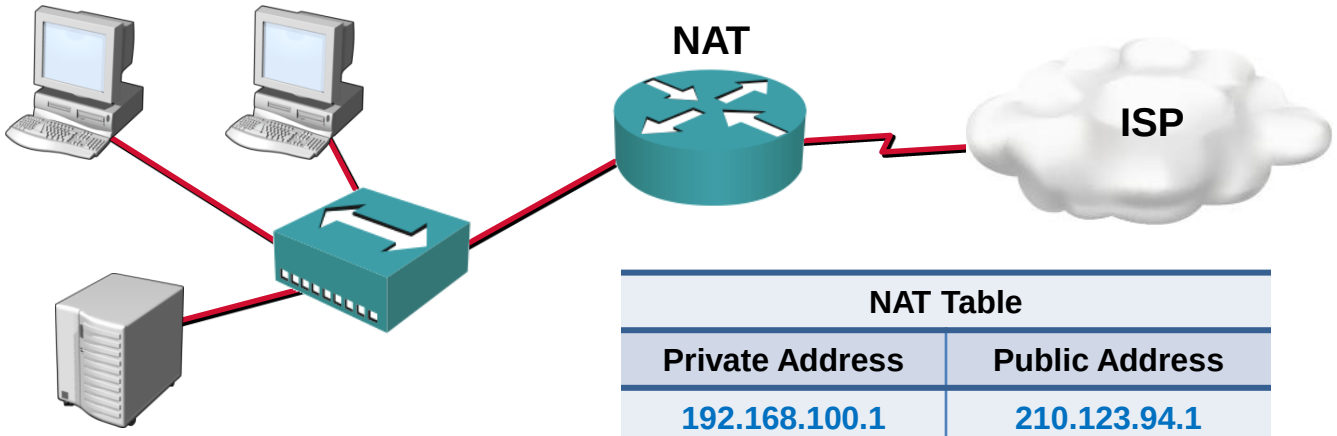
Network Address Translation

NAT(Network Address Translation)

- NAT는 RFC 1631에 정의된 것으로 IP Header의 주소를 다른 주소로 바꾸는 기술이다. NAT는 사설주소를 사용하는 호스트들이 인터넷에 서비스를 이용할 수 있도록 하기 위해 사용한다



Dynamic & Static NAT



NAT Table	
Private Address	Public Address
192.168.100.1	210.123.94.1
192.168.100.2	210.123.94.2

- 동적 NAT는 호스트가 요구하는 Traffic을 받으면 IP 주소내에서 사설 IP를 라우터에 설정된 주소풀에 있는 공인 IP로 변환한 후 외부로 전달한다. 외부에서 응답 신호가 라우터로 돌아오면 NAT라우터는 NAT Table에 있는 이전 정보로 목적지로 들어온 주소를 사설 IP로 변환해서 내부망으로 전달한다
- 정적 NAT는 외부주소로 들어온 요청을 내부 서버로 전달 될 수 있도록 목적지 주소를 변환하는 기능이다. 이 방법으로 사설망 서버를 구현하고 외부 주소로 들어오는 연결을 내부 서버로 전달할 수 있다

Dynamic NAT Configuration

- IP 변환에 사용할 전역 주소풀을 설정한다

```
Router(config)#ip nat pool name start-ip end-ip {netmask Netmask | prefix-length
Prefix-length}
```

- 내부에서 IP변환을 허용할 주소를 Standard Access-list로 정의한다

```
Router(config)#Access-list number permit source-address [Wildcard-mask]
```

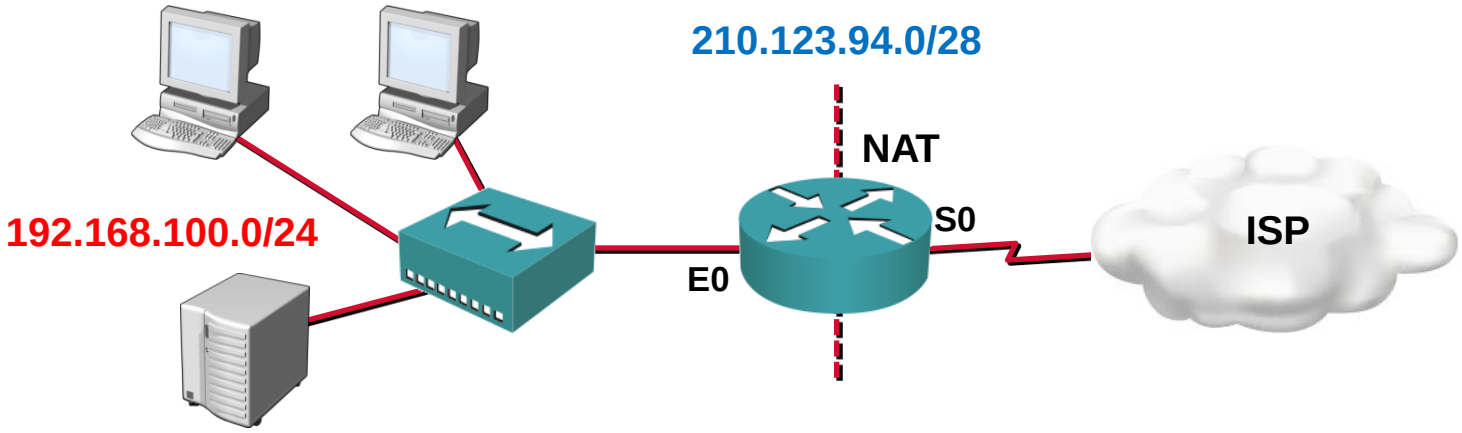
- 동적 변환을 수립하기 위한 NAT 설정을 한다

```
Router(config)#ip nat inside source list Access-list-number pool name [overload]
```

- 각 인터페이스로 이동 후 내부와 외부를 각각 설정한다

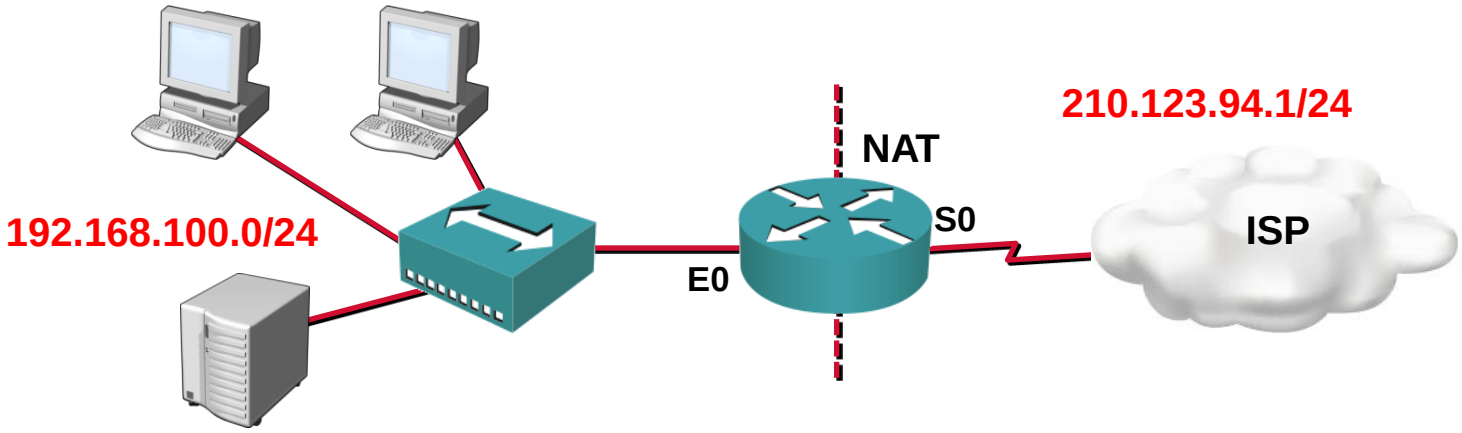
```
Router(config-if)#ip nat inside
Router(config-if)#ip nat outside
```

Dynamic NAT LAB



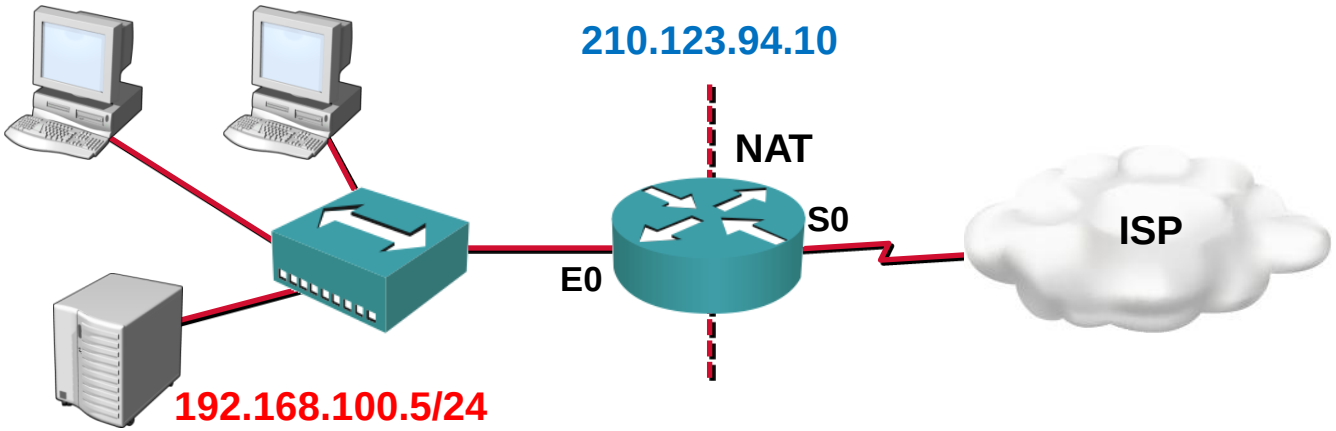

```
NAT(config)#ip nat pool Pub_IP 210.123.94.1 210.123.94.14 netmask 255.255.255.240
NAT(config)#access-list 50 permit 192.168.100.0 0.0.0.255
NAT(config)#ip nat inside source list 50 pool Pub_IP
NAT(config)#int e0
NAT(config-if)#ip nat inside
NAT(config-if)#int s0
NAT(config-if)#ip nat outside
```

NAT-PAT Example




```
NAT(config)#ip nat pool myhome 210.123.94.1 210.123.94.1 netmask 255.255.255.0
NAT(config)#access-list 50 permit 192.168.100.0 0.0.0.255
NAT(config)#ip nat inside source list 50 pool myhome overload
NAT(config)#int e0
NAT(config-if)#ip nat inside
NAT(config-if)#int s0
NAT(config-if)#ip nat outside
```

Static NAT Configuration



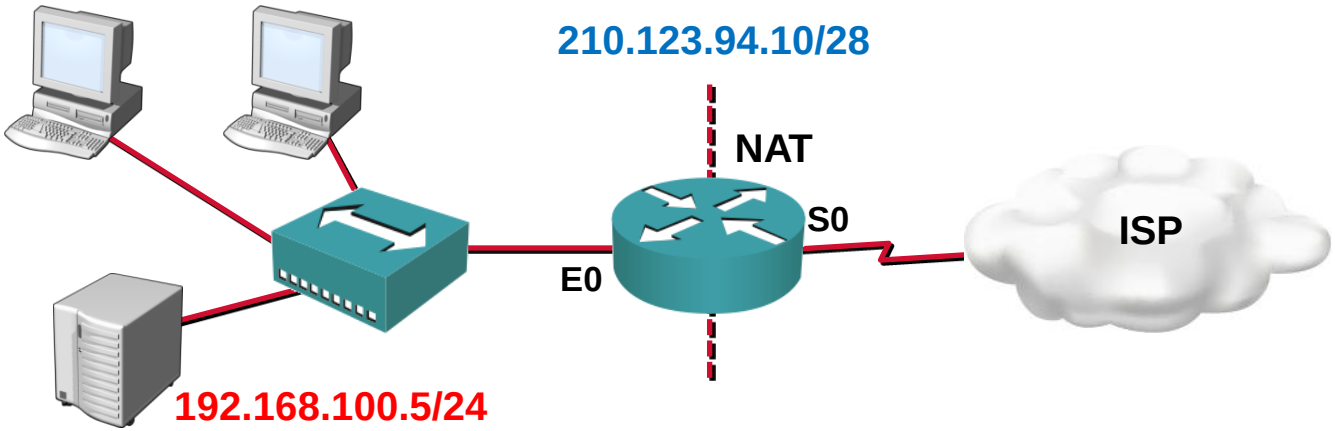
- 정적 변환을 수립하기 위한 NAT 설정을 한다

```
Router(config)#ip nat inside source Static local-ip global-ip
```

- 각 인터페이스로 이동 후 내부와 외부를 각각 설정한다

```
Router(config-if)#ip nat inside
Router(config-if)#ip nat outside
```

Static NAT Example




```
NAT(config)#ip nat inside source static 192.168.100.5 210.123.94.10
NAT(config)#int e0
NAT(config-if)#ip nat inside
NAT(config-if)#exit
NAT(config)#int s0
NAT(config-if)ip nat outside
```

NAT Table 보기

```
R1#show ip nat translations
Pro    Inside global                 Inside local                Outside local
Outside global
udp    210.1.0.1:1438                10.10.10.1:1438       210.1.1.2:69        210.1.1.2:69
udp    210.1.0.1:1439                10.10.10.2:1438       210.1.1.2:69        210.1.1.2:69

R1#
```

```
R1#show ip nat translations
Pro    Inside global                 Inside local                Outside local
Outside global
       210.1.0.11   10.10.10.2

R1#
R1#
```

NAT Monitoring

- 정적 변환을 수립하기 위한 NAT 설정을 한다

```
Router#clear ip nat translation *
```

- 내부 및 외부변환을 모두 포함하는 단순 동적 변환 주소 엔트리를 제거하기

```
Router#clear ip nat translation inside global-ip local-ip outside global-ip local-ip
```

- 활성화된 변환 정보 보기

```
Router#show ip nat translation [verbose]
```

- 변환된 통계 정보 보기

```
Router#show ip nat statistics
```

- NAT 변환 상태 모니터링

```
Router#debug ip nat
```