<PPP> - 2계층
-주요 구성 요소
1) PPP 캡슐화 방법 : IP 데이터그램 같은 상위 계층 메시지를 받아서 하위 물리 계층 링크로 전송하기 위해 캡슐화
2) 링크 제어 프로토콜(LCP) : 장비간 링크의 수립, 유지, 종료를 책임진다 (장비간 협상)
3) 네트워크 제어 프로토콜(NCP) : 서로 다른 3계층 데이터그램 유형을 캡슐화하는 기능 제공
(LCP에서 일반 링크 구성을 완료하면, PPP링크를 통해 전달되는 3계층 프로토콜에 해당하는 NCP로 제어가 넘어간다)
-PPP 기능 그룹 (PPP의 기본 동작을 지원)
1) LCP 지원 프로토콜 : 링크 협상 과정에서 관리나 옵션을 설정
(챌린지 핸드셰이크 인증 프로토콜 CHAP, 비밀번호 인증 프로토콜 PAP)
1.PAP : 사용자 이름, 비밀번호를 평문 형태로 링크로 전송한다 (보안상 위험)
2.CHAP : 초기화 장비에게 '챌린지' 텍스트 프레임을 전송하고, 초기화 프레임이 그 것을 암호화하고 다시 전송한다. 그것을 받은 후에 승인 OR 승인거절을 전송한다.
(링크로 비밀번호를 보내지 않음)

2) LCP 선택적 기능 프로토콜 : PPP 링크가 수립되고 데이터그램이 장비 간에 전송될 때의 동작을 향상시키기 위해
(PPP 압축 제어 프로토콜 CCP, PPP 암호화 제어 프로토콜 ECP)

-일반 동작
1) 링크 수립과 구성 :
1. 물리연결이 되면 장비A가 물리 링크를 통해 LCP 설정요청 메시지를 장비B에게 보냄
2. 장비 B가 승인하면 링크 상태는 개방으로 바뀌고 인증단계로 넘어간다.
3. 인증(LCP 지원 프로토콜)이 성공하면 NCP (네트워크 제어 프로토콜) 단계로 넘어간다
4. 링크 개방 단계에서 LCP 링크와 NCP 링크는 열린 상태에서 동작한다 (이 때, 각 NCP를 통하여 데이터가 전달될 수 있다.)

LCP를 통해 두 장비간 링크 수립에 필요한 인자(데이터 어떻게 전송?)에 동의. 필요하다면 LCP 지원 프로토콜의 도움을 받음. 링크가 수립되면 링크 구성을 마치기 위해 링크를 통해 전달되는 3계층 프로토콜을 위한 적절한 NCP가 호출된다. (ex. IP용 NCP : IPCP)
2) 링크 동작 : 장비들은 링크를 사용하여 데이터그램 전송. 수신의 경우 PPP 헤더를 제거하고 3계층으로 올려보낸다 (압축과 같은 프로토콜을 사용할 수 있음)
3) 링크 종료

<NCP> : NCP는 LCP의 경량 버전처럼 동작한다. (NCP링크 수립, 유지, 종료)
1) 링크 설정(LCP 수립 다음) : 설정요청,승인,등의 메시지를 통해 수행 (네트워크 인자들)
2) 링크 유지
3) 링크 종료 : NCP링크를 닫는다고 해서 LCP 링크가 닫히지는 않는다.

-NCP(ex. IP용 NCP : IPCP) 설정요청, 설정승인 후, IP 데이터가 링크를 통해 전송될 수 있다.

<IPSEC> : IP 네트워크를 위한 완전한 보안 솔루션을 제공하는 서비스 프로토콜의 모음
-IPSEC은 IP계층에서 동작하기 때문에 자체 보안 기능이 없는 상위 TCP/IP 계층 보호가능
-IPSEC은 주로 가상 사설 통신망(VPN)을 구현하는데 쓰인다.
-AH, ESP는 별도의 분리된 프로토콜이 아니라, IP 데이터그램의 헤더에 삽입되는 방식으로 구현

<IPSEC 핵심 프로토콜>
1) AH(IPSEC 인증 헤더) : IPSEC을 위한 인증 서비스 제공. 메시지의 송신자로 표현된 장비가 실제로 그 메시지를 송신했다는 것을 수신자가 검증할 수 있도록 한다. 그리고 허가받지 않은 사용자가 메시지를 캡쳐한 뒤 나중에 재송신하는 재생 공격에 대한 보호 기능도 제공

2) ESP(보안 페이로드 캡슐화) : AH는 데이터 무결성은 보장, 프라이버시는 보장X
ESP를 통하여 암호화한다.

<IPSEC 보조 구성 요소 – 여러 기타 프로토콜과 서비스의 도움으로 핵심 프로토콜이 동작>
1) 암호화/해싱 알고리즘, 키 교환 방법
2) 보안 정책, 연관, 관리방법 : IPSEC은 서로 다른 장비가 서로 보안을 유연하게 구현하도록 함

<IPSEC 구조와 구현 방법 – 어떤 구현 구조를 네트워크 상의 어떤 위치에서 사용?>
-네트워크 상의 위치
1) 종단 호스트 구현 : 모든 호스트 장비에 설치하는 방식 (보안성좋지만, 작업량 많다)
2) 라우터 구현 : IPSEC를 구현한 라우터 쌍 사이만을 보호
-구현 구조
1)통합 구조 : IP자체에 직접 통합(IPV6부터) – 기존의 장비 IP 구현을 변경하는건 쉽지않음
2) 스택 삽입 구조(BITS) : IP와 데이터 링크 계층 사이에서 별도의 계층으로 존재
3) 라인 삽입 (BITW): IPSEC 서비스를 제공하는 하드웨어 장비를 추가, 두 사이트의 라우터와 인터넷 사이의 구간에 특수 IPSEC 장비를 끼워 넣을 수 있다.

이들 구조와 관련해서 IPSEC에는 두 가지 동작 모드가 정의되어 있다.
(모드에 따라서 헤더들의 배열이 달라짐)

1) 전송 모드 : IPSEC헤더(AH, ESP)는 오직 IP페이로드에만 적용되고, IP헤더에는 적용X
(주로 IPSEC 직접 실행하는 호스트에서 종단간 보안을 필요로 하는 구현)
2) 터널 모드 : IPSEC는 IP헤더가 이미 추가된 완전히 캡슐화된 IP데이터그램을 보호
(주로 VPN)

-장비가 다른 장비와 안전한 통신을 하기 위해서는 SA를 수립해야 한다(SA는 단방향)
SA는 3개의 인자 모음으로 정의된다
1) 보안 인자 색인(SPI) : SA를 유일하게 식별하는 ID (32비트)
2) IP 목적지 주소 : SA가 수립된 장비의 주소
3) 보안 프로토콜 식별자 : 이 연관이 AH를 위한건지 ? ESP를 위한건지?
<AH 포맷>
1) 다음 헤더 : AH 다음에 오는 헤더의 프로토콜 번호. 헤더를 서로 연결
2) SPI : 목적지 주소, 보안 프로토콜 유형, 이 데이터그램에 쓰이는 SA를 식별
3) 순서 번호 : 두 장비간 SA가 구성될 때 0으로 초기화되고 데이터그램을 송신할 때 마다 증가한다. (재생공격으로부터 IPSEC을 방어)
4) 인증 데이터 : 해싱 알고리즘의 계산 결과인 무결성 검사값(ICV)을 포함

<ESP 필드>
1) ESP 헤더 : SPI와 순서 번호를 포함 (암호화된 데이터 앞에 온다)
2) ESP 트레일러 : 패딩과 패딩 길이 필드를 이용해 암호화된 데이터를 32비트로 맞춤
(암호화된 데이터 뒤에 위치)
3) ESP 인증 데이터 : AH 프로토콜과 유사한 방식으로 계산되는 ICV 포함

ESP 필드가 각각 구분된 이유 : 일부 암호화 알고리즘은 암호화될 데이터가 특정 블록크기를 가져야 한다고 요구, ESP 인증 데이터는 암호화하고 남은 데이터그램의 나머지 부분을 인증해야 되서 별도로 분리

<ESP 포맷>
1) ESP 헤더 : 1.SPI : 목적지 주소 + 보안 프로토콜 유형 + 데이터그램에 쓰이는 SA를 식별
2.순서 번호 : 재생 공격 방어 필드
2) 페이로드 : 암호화된 페이로드 데이터
3) ESP 트레일러 : 패딩(암호화, 정렬)위해 추가적인 패딩 바이트, 다음헤더(헤더연결)
4) ESP 인증 데이터 : ESP 인증 알고리즘을 적용하여 계산한 ICV 포함

<모바일 IP>
배경 : 이동 노드는 네트워크 사이를 이동하기 때문에 전통적 ip 라우팅은 ip주소가 호스트의 네트워크에 단단히 결합되어 있어서 문제가 된다.

<모바일 IP 동작과정>
1) 에이전트 통신 : 이동노드는 에이전트 발견 과정을 통해 로컬 네트워크의 에이전트를 발견한다. 장비는 에이전트가 보내는 광고 메시지를 듣고 자신이 어디에 있는지 알아낸다
2) 네트워크 위치 결정 : 에이전트 발견 메시지 내용을 기반으로 자신이 홈 네트워크/외부 판단
3) COA(care of address) 획득 : 임시 주소 획득(패킷을 목적지로 전달할 때 사용)
4) 에이전트 등록 :홈 에이전트에 자신에게 오늘 패킷을 전달해달라고 요청
5) 패킷 전달 : 홈 에이전트는 이동 장비에게 온 패킷을 받아 실제 이동 장비의 위치로 전달
(COA 종류에 따라 노드에게 직접 전달, 외부 에이전트에게 전송을 부탁할 수가 있다)

<COA 주소 획득 방법>
1) 외부 에이전트 COA : 외부 에이전트가 광고메세지로 자신의 IP주소인 COA주소를 실어 보낸다
(홈 에이전트가 보낸 패킷은 외부 에이전트를 통해서 이동 노드에게 전달된다)

2) 공존 COA : 모바일 IP가 아닌 다른 방법(IP주소 직접 입력, DHCP할당)을 통해서
(홈 에이전트가 패킷을 직접 이동 장비에게 보낼 수가 있다)
-> 이동 장비를 위한 외부 네트워크의 IP주소가 부족할 수도 있다
-> 외부 에이전트가 없거나 있더라도 오랫동안 연결을 유지하려 할 때 사용

<ICMP 정보 요청 메시지>
ICMP 정보 요청 메시지 : 인터네트워크 문제를 식별하고 교정, 테스트 기능
(서로 데이터그램을 송신할 수 있는지 확인)

<ICMP 타임스탬프 메시지>
배경 : 인터네트워크의 모든 호스트와 라우터는 서로 독립적으로 동작한다. 각 장비는 별도의 세스템 시간을 관리한다. 따라서, 두 장비의 시스템 시간이 너무 많이 차이가 나면 특정 application 이 제대로 동작하지 않을 수가 있다.

➔ 시간을 동기화 하기 위해 ICMP 타임스탬프 요청/응답 메시지

필드 중, 식별자와 순서 번호 필드는 타임스탬프 요청과 응답 메시지를 대응시키는데 쓰인다.
그러나, ' 장비가 서로에게 데이터그램을 송신하는데 걸리는 시간'이 각 데이터그램별로 달라서 시간을 동기화시키는게 어렵다.

➔ NTP(네트워크 시간 프로토콜)을 사용한다

<ICMP 라우터 광고/ 라우터 정보 요청 메시지>
호스트는 경로에 대한 정보가 별로 없다. 호스트에게 라우터 정보를 제공하는 한가지 방법은 각 호스트의 기본 라우터로 사용할 로컬 라우터 주소를 수동으로 입력

'라우터 발견' : 호스트가 자동으로 로컬 라우터를 식별
-라우터는 정기적으로 '라우터 광고 메시지'를 송신하는데 시간 간격(10분)을 줄일 수 있다.
- 라우터 광고 메시지를 기다리는 대신, 호스트는 라우터에게 '라우터정보요청 메시지' 송신
-이외에도 동적 호스트 설정 프로토콜(DHCP)로 기본 라우터를 알 수 있다.

<ICMP 시간 초과 메시지 응용>
기본적으로 ICMP 시간 초과 메시지는 오류 메시지 이지만, TCP/IP tracerout를 구현하기 위해 쓰인다.
1) 더미 데이터그램의 TTL 값을 1로 설정하여 전송하면 첫번째 홉에서 라우터가 데이터그램을 버리고 ICMP 시간 초과 메시지를 되돌려 보낸다
2) 다시 같은 목적지로 데이터그램을 보내는데 TTL 값을 2로 설정하여 보내면 경로상 두 번째 장비가 시간초과 메시지를 보내는 방식으로 traceroute를 구현한다.

<ICMP 인자 문제 메시지>
목적지 접근 불가, 패킷 크기 초과, 시간 초과 메시지로 설명할 수 없을 때를 대비하여 범용 오류 메시지 유형을 정의하였다.(ICMP 인자 문제 메시지)

<출발지 링크 계층 주소 선택사항 포맷> : ICMP 메시지를 보내는 장비의 링크 계층 주소 포함
-라우터 요청/광고 메시지
<목적지 링크 계층 주소 선택사항 포맷>
-리다이렉트 메시지 대상 주소 필드

장비가 로컬 네트워크에 있는 다른 장비들과 해야 하는 일
1) 직접 데이터그램 전송 2) 2계층 주소 지정 3) 간접전송 할 때 로컬 라우터 식별
IP 주소지정 방식 : 로컬 주소인지 ? 멀리 떨어져 있는 장비의 주소인지 식별
ICMP : 로컬 장비 간의 다양한 통신을 지원하는 메시지 시스템

<IPV6 주변 발견(ND) 프로토콜 > : ICMP와 유사함
-ND 프로토콜은 로컬 장비가 서로의 존재를 발견하는 것과 관련이 깊다.
-주변 장비를 발견, 로컬 네트워크 연결, 데이터 그램 라우팅과 구성에 관한 여러 기능 수행
(호스트와 라우터가 네트워크 기능을 적절히 실행하기 위해서는 ND프로토콜이 필요하다)
1) 호스트 라우터 발견 기능 :
-라우터 발견(RD)
-접두사 발견(RD와 밀접) : 자신이 속한 네트워크 알아내고, 직접or간접전송 한다
-인자 발견 : MTU 습득

2) 호스트 간 통신 기능
-주소 결정 : 로컬 네트워크의 다른장비 3계층주소로 2계층 주소를 알아냄(ARP대신)
-다음 홉 결정 : 목적지 주소를 보고 어디로 보내야 할지 결정

3) 리다이렉트 기능
그러나 ND 프로토콜은 ICMPv6 메시지를 통해 구현한다
(라우터 광고/요청 메시지, 조변 정보 요청/광고 메시지, 리다이렉트 메시지)

IP에서 데이터그램을 전송하는 방식은 직접 전송과 간접 전송으로 나눌 수 있다.
1) 간접 전송 : 라우터가 호스트를 도와야 하므로 호스트 라우터 발견 기능이 필요
2) 직접 전송 : 라우터를 사용X 로컬 호스트 간의 직접 통신관련 IPv6 ND 프로토콜 필요

주소 결정 : 3계층 주소로 2계층 주소를 알아내는 것
-IPv6에서는 ND 프로토콜이 주소 결정을 책임 지는데, 목적지 장비의 ip주소를 ICMPv6 주변 정보 요청 메시지에 실어 전송한다.
-호스트는 주변 장비에 대한 정보를 저장하는 주변 노드 캐시를 관리한다.
-주변 정보 요청 /광고메시지는 '주소 결정기능'과 관련이 깊지만, 호스트가 IPv6 자동 구성을 할 때 호스트가 사용하려 하는 주소가 네트워크에 이미 존재하는 중복 주소 검사로 사용될 수 있다.

<RIP>
개요
-RIP는 경로를 결정하기 위해 거리 기반 벡터 알고리즘을 사용한다
-각 라우터는 다양한 네트워크, 호스트에 대한 정보를 담은 항목을 라우팅 테이블에 모아 관리
-항목은 네트워크나 호스트의 주소와 그곳까지의 거리(라우터수 == 홉수)
-라우터는 정기적으로 자신의 라우팅 테이블을 특별한 메시지를 통해 자신이 접속해 있는 네트워크에게 전송한다 (UDP를 사용)
-구현하기 쉬우며 작은 AS에서 사용하기 좋다. (최대 15홉)
-ipv6을 위한 RIP를 RIPng(RIPv6)라고 부른다
- RIP는 다른 라우팅 프로토콜처럼 경로에 대한 정보를 교환하는 방식을 제공한다
-네트워크 라우팅 정보를 전달하는 일은 정기적OR 네트워크 구조가 바뀌었을 때 일어난다.
-라우터의 RIP 소프트웨어는 RIP 메시지로 서로 통신한다.(UDP사용)
- 라우팅 정보가 즉시 필요한 상황(처음 켜졌을 때)에 RIP 요청 메시지를 전송한다
-RIP 라우터는 30초마다 완료되는 타이머가 있는데, 타이머가 완료되면 라우터는 요청받지 않았다 해도 라우팅 테이블 전체를 RIP 응답 메시지에 실어 브로드캐스트나 멀티캐스트로 전송한다.
- 또 다른 타이머는 '만료 타이머'인데 이것은 라우터가 경로를 라우팅 테이블에 저장하는 시간을 한정한다.(180초)
-만료 타이버가 만료 후 만료된 경로의 길이를 16으로 바꾸고 곧 지울 것이라고 표시한다
그 직후, 'Garbage collection timer'를 시작시킨다(120초)
-RIP요청에 대한 응답, 30초 타이머 만료로 전송, 경로상의 변화 있을 때 RIP응답을 전송한다
'트리거 갱신'이라고 불리는 것은 경로상의 변화를 인터네트워크로 알리는 역할을 한다
(ex. 경로가 만료되어 garbage collection이 시작되면 이 경로가 더 이상 유효하지 않다는 것을 알리기 위해 트리거 갱신을 시작한다.)

라우팅 테이블에서 중요한 정보
1) 네트워크나 호스트의 주소
2) 라우터에서 네트워크나 호스트까지의 거리
3) 라우터에서의 첫 번째 홉

<네임 체계> : 기호 네임을 이용하여 장치를 식별하는 방법을 제공
1)**네임 공간의 정의** : 네임을 어떻게 구성하고 사용할 것인지에 대한 규칙을 포함
-네임에 사용할 수 있는 문자나 기호들의 수를 정한다, 최대 네임 개수
1.단순 네임 구조 : 단순한 기호의 나열, 내부 구조 X, 네임 간에 어떠한 관계도 없다,
시스템 내의 모든 장치에게 유일한 네임을 할당하기 위해 하나의 중앙 기관이 필요하다.

2.계층 네임 구조. : 복잡하지만 지역적인 구조를 이용해 네임을 부여할 수 있ㄷ.

2)**네임 등록** : 한 네임을 특정 장치에 할당하는 과정 (유일하게 할당하는 네임 관리 체계 필요)
-네임 지정과 유일성 보장
-중앙 등록 기관 지정 : 네임 지정의 과정을 책임짐 (네임이 담긴 파일을 관리)
-등록 권한 위임 : 대규모 네임 체계에서는 네임 공간을 분할하여 등록의 권한을 하부 조직으로 위임할 수 있다.

실제로 등록하는 방법
-테이블 네임 등록 : 한 명의 관리자가 하나의 테이블을 가지고 네임 지정을 운용(테이블의 수정) 주로 소규모의 단순 네임 체계를 갖는 곳에서 사용
-브로드캐스트 네임 등록 : 시행착오 방법 (한 장치가 특정 네임을 사용하고자 한다면 네트워크상의 다른 모든 장치에 이미 그 네임이 사용 중인지 확인하는 메시지를 보낸다. 해당 네임을 사용하는 장치가 없으면 등록)
-데이터베이스 등록 : ex) DNS
계층의 각 부분으로 네임 등록 권한을 위임한 상태라면 네임 등록에는 분산 데이터베이스 사용. 각 네임 등록 기관들은 계층 구조에서 해당하는 전체 데이터베이스의 일부를 관리한다.

3)**네임 변환** : 장치의 기호 네임을 숫자 주소로 변환하는 과정을 정의 (중요, 실제로가장많이사용)
-네임 체계에는 컴퓨터가 사용하는 두 개의 식별 시스템이 쓰인다 (기계가 사용하는 숫자 주소와 사람이 사용하는 네임), 네임 체계는 두 방식을 통합하는 역할을 한다.
1)테이블 기반 네임 변환 : 테이블을 통해 기계의 네임을 주소로 변환하는 방법 얻는다.
2)브로드캐스트 네임변환: 브로드캐스트로 질의한다.
3)클라이언트/서버 네임 변환 : 서버는 소프트웨어를 이용해서 클라이언트가 보낸 네임 변환 요청에 응답한다. 서버는 클라이언트의 요청에 담긴 네임을 데이터베이스에서 찾고 그에 대응하는 숫자 식별자를 클라이언트에게 보낸다.

-네임 체계는 사용자가 기계 네임만 입력해도 이를 알아서 숫자 주소로 변환해 주는 소프트웨어를 이용한다.
-클라이언트의 역할을 수행하는 네임 변환기가 네임 변환 요청을 처리하는 네임 서버에 접속함.

<Networking class>

-**Firewall 방화벽** : 서로 다른 보안레벨의 네트웤의 경로 사이에 위치하며 지나 다니는 트래픽을 보고 통과 시킬지 말지를 결정 (트래픽을 집중시킴)
-기본적으로 IP, Port정보를 바탕으로 한다.
-보안 영역중에서 접근제어만 가능하다.
-**accpet된 패킷과 denied된 패킷에 대한 log를 기록한다.**

방화벽 rule : -보안 레벨이 높은곳에서 낮은 쪽으로 packet이 갈 때는 all permit 반대는 all denied (그러나, 방화벽 만드는 벤더사마다 요즘은 다 다름)
-DMZ : 인프라 네트워크 구성중에서 외부 인터넷망과 내부 인트라넷의 사이에 위치한 중간지대

1)Proxy 방식 == 게이트웨이 역할

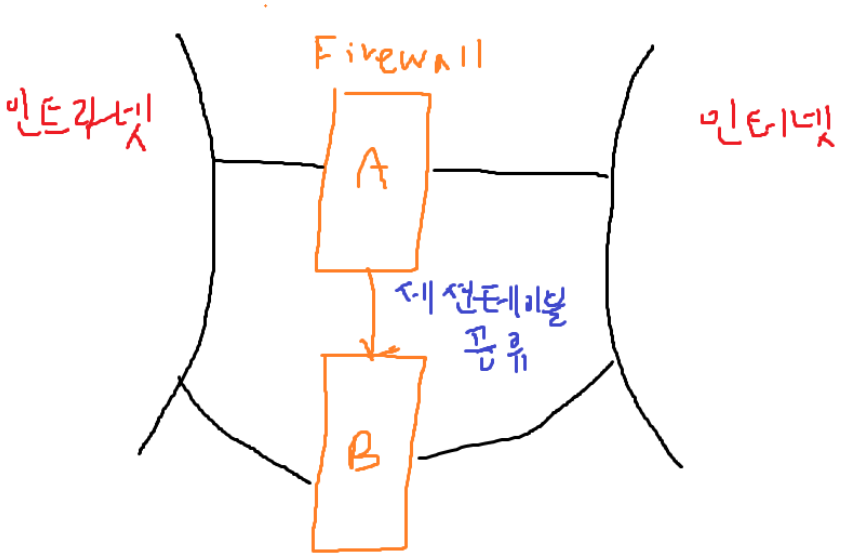
A (인트라넷 내부)　　방화벽　　B (인터넷 외부)
->A가 패킷을 보낼 때 방화벽을 경로로 삼아서 보낸다
ACL방식 : A->B로 보내는 것에 대해서도 rule을 정해야하고, TCP의 경우 A가 B로 tcp syn 보냈을 때 B가 A로 보내는 syn ack packet에 대해서도 rule을 정해야 한다.
즉, ACL 방식은 양방향을 전부 다 고려해서 어떤 것을 accept 해야 할지 결정해야 한다.
(ACL은 인터페이스 별로 작동을 한다 -> 성능은 좋으나 관리가 힘들다)

2)Stateful inspection : 상태정보를 기록하는 세션 테이블을 가진다

-A가 B로 tcp syn packet을 보내면 일단 방화벽의 rule을 확인해서 (accept)이 된 것을 보고 이 패킷에 대한 상태 정보를 방화벽의 세션테이블 메모리 특정영역에 남긴다.
-TCP SYN 패킷이 나가서(rule permit) ACK이 안들어올 가능성도 있기 때문에 방화벽의 세션테이블 메모리의 각 항목에 타이머를 걸어둔다.
-데이터가 계속 왓다갔다 하면 세션테이블의 해당 tcp 항목값을 유지를 하는데, 그렇지 않다면 설정마다 다른데 1시간동안 유지시킨다. Tcp fin이 날라오면 그냥 해당 세션테이블의 항목을 날린다.

3) UTM 방화벽 (Unified threat management): 스팸 정보 확인, URL 필터링, HTTPS 까서 확인, 응용 L7 기능, VPN 기능 까지 필터링을 방화벽에 같이 제공하는 차세대 방화벽
-기존의 방화벽은 IP 베이스로 작동되어왔으나, 사용자 베이스로 작동함.

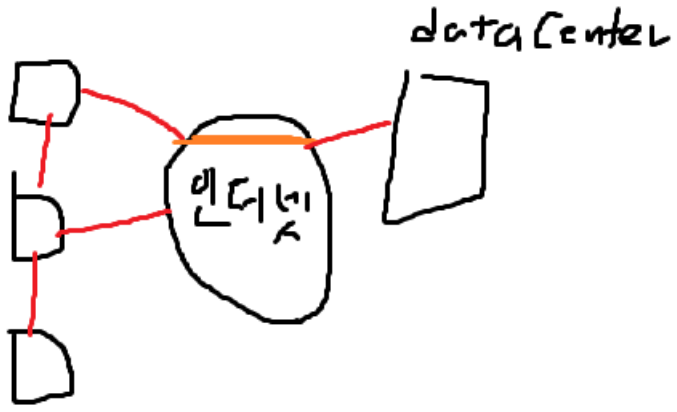
4)이중화 구성 : VRRP + HSRP + a



<VPN>: Virtual private network : 가상 사설망
컨셉 : public 망을 이용해서 private 망 처럼 활용하는 것

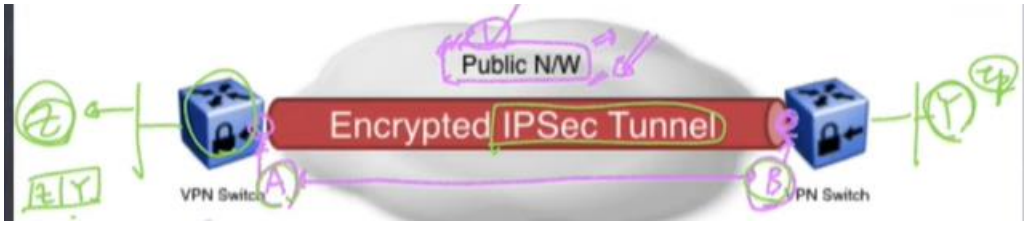
-물리적인 기능들은 이미 제공하는데 가상화된 관점으로 기존의 요구사항 등을 만족시키게 기술적으로 제공한다 (돈 때문에) for 전용회선비 절감
-집에서 통상적으로 쓰는 ADSL 쓰면 보안상으로 위험하다
-

1) **CPE-VPN** : **IPSEC 을 지원하는 VPN 장비들 끼리 터널을 뚫는다** (암호화, 무결성, 인증)
기업 내 물리적으로 떨어진 두 곳끼리 전용회선을 깔면 돈이 비싸기 때문에 장비를둬서 공중 인터넷에서 다른 사람이 못보게 하는 것



-전용회선은 거리, bps 속도에 맞춰서 가격이 정해지는데 가격이 존나 비싸다
(사용기간이 길수록 그나마 덜 비쌈)
그래서 IPSEC을 사용하는데 과정을 설명하면
-지리적으로 떨어져 있는 두 지사끼리 통신을 하려면 통신사업자의 pop 지점까지 짧은 전용회선을 깔고 pop 지점에 IPSEC 장비를 설치한다 – IPSEC 장비는 공인 IP가 있다.
-통신해야 되는 트랙픽 양이 많은 곳이 아닌 데이터는 적은데 실제 전용망을 통해서 정보를 주고받아야 하는 기업 분점들이 많이 사용한다.
-집에 가서 회사의 전용망에 접속을 해야 되는 경우는 pc에 IPSEC 모듈(소프트웨어)를 설치한다.
(내가 집에서 DHCP로 IP를 할당받고, ACCESS VPN에 의해서 기업 내부망에서 통신 할 수 있는 IP로 변환이 된다)

IPSEC
-public 망에서 end to end 간 통신의 보안 유지가 목적 (Secure channel 생성, 데이터 전달)
->상호 인증, 무결성, 암호화
-**IKE**(ISAKMP, Oakley) protocol : UDP 500번을 사용해서 보안채널 생성/키 교환 목적
-**AH**, **ESP** : 실제 사용자 데이터 전달
1) **Tunnel Mode (게이트 웨이 역할)** : 인터넷 상으로는 A와 B가 ipsec 터널을 만들어서 통신을 하는데 그 안의 데이터는 Z와 Y에게 가야된다. 따라서 뒤에 ip헤더가 하나 더 씌워진다.



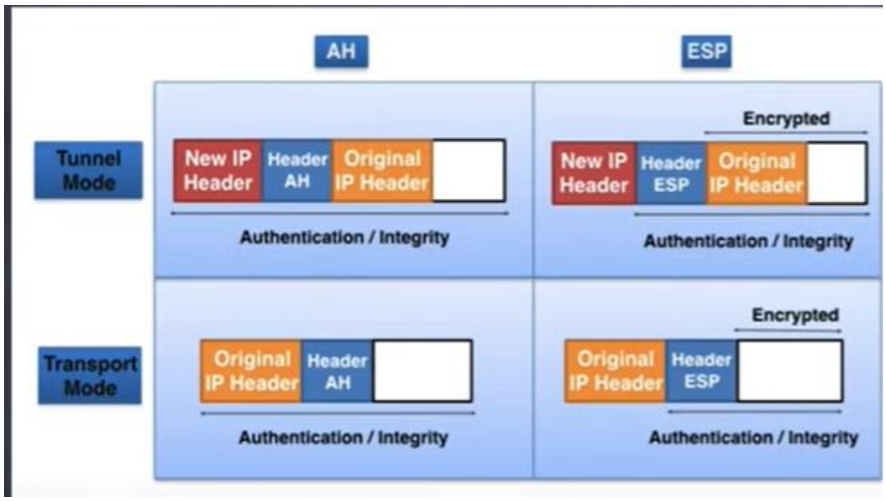
PHASE 1: A와 B가 어떻게 서로를 인증, 키 교환
PHASE 2 : A->B , B->A 에서 각각 보내는 키와 방식, ESP를 사용할 것인지 AP를 사용할 것인지 협상

-**IKE**(Internet key exchange) : IPSEC을 위한 SA(보안 연관)을 생성하며, **키 관리를 수행 하는 프로토콜**

즉, 보안 채널 생성 관리
-SA(Security Association) : IPSec end point 간에 성립된 보안관련 계약. IKE를 통해 협상(생성)
PHASE 1 : 인증 방법, 암호화 알고리즘, 해쉬 알고리즘, Key에 대한 리뉴얼
(main mode or aggressive mode)
PHASE 2 : ESP or AH 선택, 인증 방식 정의, ESP인 경우 암호화 알고리즘 결정

-AH 헤더 및 ESP 헤더 : 데이터를 전달하는 방식
IPSec에서 ip 패킷의 '인증', '암호화'를 위해 2개 헤더를 더 정의

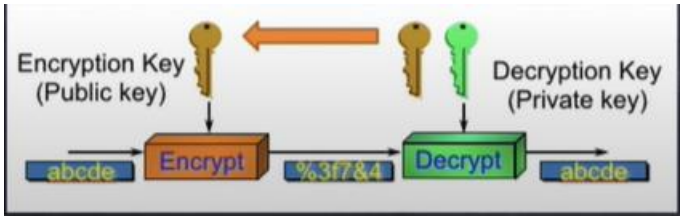
1) AH 헤더 : IP 패킷의 무결성, 인증에만 사용되는 헤더 (IP 50번)



2) ESP 헤더 : IP 패킷의 무결성, 인증, 암호화(기밀성)을 모두 제공하는 헤더 (IP 51번)
-기밀성을 위한 암호화 알고리즘 AES, DES... 사용 하고 수신측에서 IKE로 미리 교환한 Key 값을 이용하여 데이터를 복호화 한다

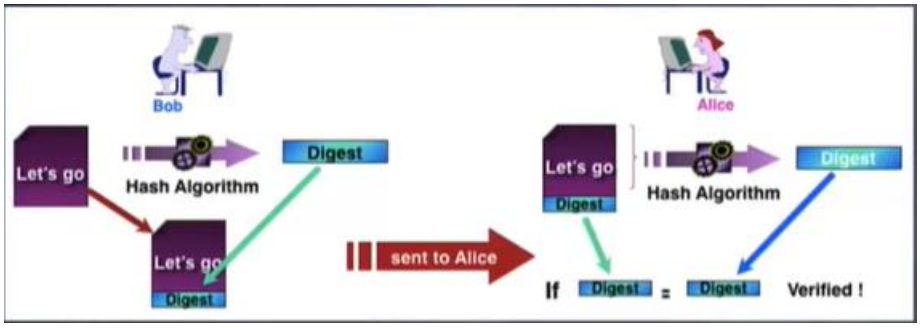
2) transport mode : A와 B가 실제로 통신한다.

<기본적인 암호학>



<암호화>

-Asymmetric cryptography : 데이터를 암호화 할 때 수신측에서 private key, public key를 만들어서 public key를 송신자에게 보낸다. 송신자는 public key로 암호화하고 수신자는 private key로 복호화 한다.

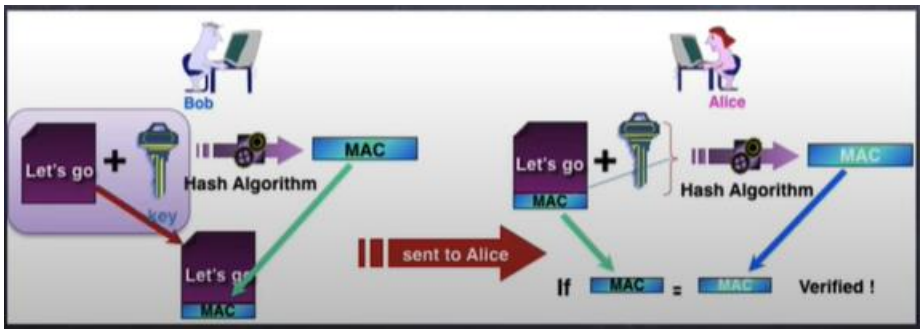


<무결성> : 데이터가 변조되지 않았는지?

1) **Hash function(SHA, MD5) – message digest**

BOB이 Let'sgo(데이터)를 보낼 때 데이터를 해쉬 알고리즘을 돌려서 나온 값을 실제로 보내는 데이터에 붙여서 보낸다. Alice는 동일하게 확인한다.

2) **MAC 방식 : Hash function + Cryptography**



Let'sgo(메시지)와 key를 합쳐서 hash function에 돌린 값이 MAC 인데 그걸 실제로 보내는 데이터에 붙여서 보낸다. 송신, 수신자가 똑 같은 KEY가 있어야 동일한 값이 나온다.

-Symmetric crptography = Hash + secret key -> IPSEC 이용

<IPSec 터널 연결 및 트래픽 전달 방식>
1) IKE Isakmp sa : 최초로 udp500번을 사용해서 서로간 인증하고 (main mode 는 6개의 양방향 메시지, aggressive mode는 3개의 메시지)
Mainmode
1단계 : 한 IPSEC VPN 가 다른 IPSEC VPN에게 인증 방식, 암호화 방식, 해쉬 알고리즘.. 협상송신
2단계 : 수신받은 IPSEC VPN이 1단계 IPSEC VPN에게 똑같이 송신한다
3,4단계 : 어떤 알고리즘을 쓰는 것이 정의가 되었기 때문에 키를 생성한다. 각각의 IPSEC VPN은 자신은 PRIVATE KEY를 가지고 상대방에게 PUBLIC KEY를 던진다.
5,6단계 : 실제 데이터를 암호화하고 인증하는 과정을 거치면서 실제 만든 키를 사용한다.

2) IPSEC Sa 생성 : quick mode (3개)
정상적으로 Iskmp sa를 통해 서로 간의 secure channel은 이미 성립된 상태임
ESP를 쓸건지 AP를 쓸건지 상호간에 협의한다.
//PFS 서칭.

대칭키 : 암호화 키와 복호화 키가 서로 동일한 방식
비대칭키 : 암호화 키와 복호화 키가 서로 다른 방식 : RSA, Diffie-Hellman

->실제 데이터 암호화 복호화 인증에 사용되는 건 대칭키를 사용하고, 이 대칭키 정보를 비대칭키를 사용해 교환한다. (대칭키가 더 빠르다)

2) SP-VPN : MPLS(레이블) 기반 라우팅임 IP 기반이 아님
-위의 그림에서 공중망(인터넷은 아님)을 보면 공중망 구간마다 인터넷 사업자(EX. Kt)가 pop이라는 지점을 둬서 고객(PE) 들에게 제공하는 VPN서비스다. 인터넷 사업자가 모든 보안, 라우팅 관련된 서비스를 처리하며 다른 고객들과는 통신할 수 없다 (라우팅 분리)

<인터넷 구간에서 NAT(PAT)시 문제>

해결 : NAT-Traversal : main mode 하는 도중에 NAT가 되는지 Message를 보낸다
Ike 과정에서 NAT 여부를 확인하고, esp 헤더 뒤에 UDP4500 으로 Wrapping하여 PAT 가능

*NAT(Network address translation) : 네트워크 주소 변환
-ip 패킷이 트래픽 라우팅 장비를 거치면서 이동할 때, 헤더에 있는 ip 정보를 수정하는 작업이다. (보통 라우터나 방화벽에 설정)
->공인 ip 개수가 부족하기 때문에 1개의 공인 ip주소에 여러 개의 사설 ip 주소를 할당한다.
1)Static Nat : IP주소들을 1 : 1 매핑으로 변환해주는 가장 간단한 NAT 유형이다.
2) Dynamic NAT : Static NAT 처럼 NAT 라우터가 사설 IP와 공인 IP 간의 매핑 정보를 생성한다. 그리고는 IP 패킷이 네트워크에서 나가거나 들어올 때 그 패킷 헤더의 IP 주소를 변환한다.

-그런데 이 때, 사설 IP와 공인 IP를 매핑하는 것을 정적인 정보에 기반으로 하지 않고 동적으로 수행한다. 사용가능한 공인 IP주소들의 pool을 만들어 두고,

*PAT : Dynamic NAT의 한 종류.
-공인 IP 주소 1개에 사설 IP 주소 여러 개를 매핑하는 것
-변환된 IP 주소로는 사내망 호스트들을 구분할 수 없기 때문에 포트번호를 부여하여 구분한다.
(대부분 홈 네트워크는 PAT을 사용하고 있다)

EX) ISP는 홈 네트워크 라우터에 하나의 공인 IP 주소를 할당한다. 컴퓨터 A가 인터넷에 로그온하면 라우터는 사설 주소에 포트 번호를 붙여서 유일한 주소를 할당한다. 그 다음에 컴퓨터 B 가 인터넷에 로그온하면 컴퓨터 A에게 할당한 똑 같은 사설주소에 다른 포트 번호를 붙여서 유일한 주소를 만든다.

<도커의 배경>

가상화 : 컴퓨터 안에 독립적인 컴퓨터를 만드는 것
가상화를 하는 이유 : **물리적인 리소스를 여러 사용자 또는 환경에 배포해서 제한된 리소스를 최대한 활용하기 위함**
Ex) 3개의 물리 서버가 있는데 사용량이 크지 않는 경우, 가상화를 이용해 **하나의 서버에 두 개의 서버를 독립적으로 분리한다.**

추가적으로, 가상화는 단일 하드웨어 시스템에서 여러 운영 체제(Windows, Linux)가 동시에 실행될 수 있도록 한다. 컨테이너는 동일한 운영 체제 커널을 공유하고, 시스템의 나머지 부분으로부터 애플리케이션 프로세스를 격리한다.

1) **호스트형 서버 가상화** : 하드웨어(x86 Architecture) 위에 호스트 OS를 설치하고 가상화 SW를 이용해 게스트 OS를 작동시키는 기술

2) **하이퍼 바이저형 가상화**
하드웨어 상에 가상화를 전문적으로 수행하는 SW인 하이퍼바이저를 올라가는 방식. 이 하이퍼바이저가 하드웨어와 가상 환경을 제어한다. 호스트OS가 없어져서 이전꺼 보단 부담이 덜하지만, 각 VM마다 게스트 OS가 들어가기 때문에 가상 환경 시작에 걸리는 오버헤드가 크다.

3) **컨테이너**
오버헤드를 최소화 하기 위한 방법이다. 호스트 OS 상에 독립적인 공간을 만들고 별도의 서버인 것처럼 사용한다. 각 컨테이너는 같은 호스트 OS를 공유하기 때문에 오버헤드가 적고 고속으로 동작한다

4) 도커 : 애플리케이션 실행에 필요한 환경을 이미지로 만들고 해당 이미지를 활용해 다양한 환경에서 실행 환경을 구축하기 위한 오픈소스 플랫폼이다. (내부에서 컨테이너 사용)

5) 쿠버네티스

클러스터링 : **실제 애플리케이션은 여러 컨테이너에 걸쳐 있고 이러한 컨테이너는 여러 서버에 배포되어 있다.** 이렇게 여러 대의 서버나 하드웨어를 모아서 **한 대 처럼 보이게** 하는 기술이다.
이런 멀티호스트 환경에서 컨테이너를 클러스터링하기 위한 툴을 컨테이너 오케스트레이션 툴이라고 한다. (컨테이너 시작 및 정지, 호스트 간 네트워크 연결, 스토리지 관리, 컨테이너를 어느 호스트에서 가동 시킬 지와 같은 스케쥴링 기능을 제공한다.)

쿠버네티스는 이런 컨테이너 오케스트레이션 툴의 표준이다.

-가상 머신을 구성하기 위해 우리는 VMWare, Virtualbox, Hyper-v 기술을 사용했으나 최근에는 Docker로 대표되는 LXC(Linux Container)와 같은 컨테이너형 가상화로 옮겨지고 있다.

기존의 가상 머신
-가상 머신 자체는 완전한 컴퓨터다. OS를 자체적으로 가지고 있기 때문에 OS를 가상머신 이미지에 포함해야 하고 배포이미지의 용량의 커진다는 단점이 있다.
-가상화 기술의 개발이 활발해 지면서 기존의 하이퍼바이저와 다른 베어메탈 하이퍼바이저가 공개되면서 성능 문제는 많이 해결됨.

컨테이너 기술
-리눅스에서는 리눅스 컨테이너라고 불리는 곳에 프로세스를 격리시킨다.
-컨테이너 가상화는 사용자 공간을 여러 개로 나누어 각각의 사용자 프로세스에서 보이는 자원을 제한하여 CPU나 메모리는 프로세스가 필요한 만큼만 추가적으로 사용하고 성능적으로 손실이 없는 것이 특징이다.
-기존의 가상화 기술에서는 가상 머신이 실제 물리 하드웨어를 에뮬레이트하기 때문에 OS가 필요하다
-LXC는 모든 프로세스는 호스트 OS에서 바로 시작하며, 일반적인 프로세스와 다른점은 그 과정의 일부를 그룹화하고 다른 그룹과 그룹에 속하지 않는 프로세스에서 단절된 공간으로 동작
-LXC는 이러한 방식의 차이점 때문에 가상화 오버헤드가 발생하지 않고 가상 머신의 부팅 및 종료라는 개념이 존재하지 않는다. OS가 사용하는 자원을 분리하여 여러 환경을 만들 수 있고, 가상 환경의 시작과 종료를 빠르게 실행할 수 있다.

컨테이너의 장점
-LXC는 다른 가상화 기술머신보다 시작과 종료가 빠르다
(**컨테이너를 생성한다는 것은 OS 입장에서 단순히 프로세스를 시작하는 것이기 때문**)
-낮은 오버헤드 : 가상화를 위한 하드웨어 에뮬레이트 단계가 없다.
-높은 집적도 : 컨테이너는 커널이 직접 프로세스를 조작하여 분리된 공간을 구성하기 때문에 PC상에서 동작하는 OS는 하나이다.
-컨테이너는 가상머신과 달리 init을 먼저 시작하거나 각종 데몬들을 실행할 필요가 없다.
-컨테이너는 목적에 맞는 프로세스만 존재하는 환경을 만들 수 있다.
(ex. 웹 서버용 컨테이너 : Apache Httpd 프로세스만 존재하는 컨테이너를 만들 수 있다.)

컨테이너 단점
-Host OS에 종속적 : Linux 컨테이너는 OS에서 Linux Kernel이 관리하기 때문에 Linux 이외의 다른 OS에서는 동작하지 않으며, 컨테이너 환경에서도 다른 OS를 설치할 수 없다.

도커

-앞에서 말한 컨테이너 기술을 사용한다. 프로세스를 격리시켜 사용하는 방식이기 때문에 가볍고 빠른 동작이 가능하다.
-하나의 서버의 여러 개의 컨테이너를 실행하면 서로 영향을 미치지 않고 독립적으로 실행됨
-실행중인 컨테이너에 접속하여 명령어를 입력할 수 있고, apt-get 이나 yum으로 패키지를 설치할 수 있다

Dokcer image
-이미지는 컨테이너 실행에 필요한 파일과 설정값등을 포함하고 있는 것이다.
-컨테이너를 이미지를 실행한 상태라고 볼 수 있다.

<

<AWS>

CSP (Cloud Service Provider) 가 제공하는 서비스들을 선택하는 패러다임으로 넘어왔다.
Public cloud enterprise (ex. 아마존) 의 데이터센터가 기존의 데이터센터를 대체할 것이다.

AWS 서비스 물리 구성 현황
-글로벌하게 서비스를 제공하는 Region이 존재하며 하나의 Region에는 다수의 AZ으로, 독립된 데이터센터가 존재. (Region 별로 특징이 다르다)
-Edge 센터 (CDN 서비스 제공 – 응답 속도 측면에서 – 사용자가 가까운곳에 캐쉬기능 장비 구축)
-DX 센터 : 기업의 전용망 서비스까지 아마존 데이터센터와 연동해서 POP을 구성한다.

-기존 데이터센터에서 구매, 설치, 구성해야 했던 대다수의 서비스가 제공된다.
-인프라 영역부터 Application service, 보안 , Data 분석 까지 다 제공한다.

AWS Network (VPC 내에서)
: VPC에 대한 기능, Restriction 및 Network performance
-VPC IP Addressing , ipv6 …

1) Design and implement AWS Network
Edge location : CDN 서비스를 위함. 통신 사업자들의 rect를 임차해서 AWS 서비스를 구축
주로 고객들의 서버자원 들이나 서비스 자원들은 Region에 있지만 Region 안에 Edge location과 연동이 되어 네트워크 인프라가 구성되어 있음.
-Region 안에 AZ(독립된 곳)과 연동되어 있고 각 국가의 통신사업자들과 연동하기 위한 Edge location과 전용망 연동을 위한 DX location이 있다.

-VPC 내의 호스트들은 DHCP SERVER에 브로드캐스트 같은 형태로 자동으로 게이트웨이, DNS 등을 할당받는다.
-DHCP option set 의
-아마존에서 제공하는 Default AWS provided DNS 대신에 Route 53에 private hosted zone 이라는 사설 도메인을 만들고나서 provided dns 에 등록이 된다. (이렇게 하면 아마존 서비스들을 사용하는데 어려움이 있다.) , 해당 private host zone에 있는 instance들은 default provided dns가 아니라 사설 도메인을 lookup 하게 된다.

-AWS가 가지고 있는 공인IP 대역이 아니면 인터넷과 통신 할 수 없다.
-Reserved IP : ex) 10.0.0.0/24
0 : Network address
1 : 아마존이 가지고 있는 Default gateway
2 : 아마존이 내부에 서브넷에 제공하기 위한 DNS 서비스를 위해 예약

3 : 나중의 서비스를 위해
253 : ROUTE 53의 사설 HOST ZOEN DNS
255 : 브로드캐스트

-VPC의 IP를 Private subnet, public subnet으로 나뉜다 (private subnet은 외부 인터넷과 통신이 안되게 만들어 놨다 – 보안적인 측면)
-> 서브넷 별로 라우팅 테이블 (해당 서브넷이 어디로 갈 수 있는지가 정의되어 있음)이 있는데 라우팅 테이블의 default gateway로 인터넷에 나가는 IGW gateway로 잡을 수가 있냐 의 문제다 (실제로는, IGW로 인터넷에 나갈 때 NAT 될 수 있느냐)
-아마존이 제공하는 서비스들과 직접 통신이 가능한데 이것은 end point 라고 하는데, end point를 통해서 외부로 나갈 수도 있다.
-VPN을 위한 VGW로도 외부로 나갈 수가 있다.

VPC 내 Private ip : instance가 STOP되거나 rebooting 되어도 ip가 바뀌지 않는다
(private ip는 instance가 생성이 될 때, DHCP로 자동으로 설정된 값)
Public ip : instance가 stop 되거나 rebooting , terminate되면 ip가 바뀐다
(IGW가 NAT 시키는 것)
Elastic ip : instance에 elastic ip를 인스턴스에 매핑을 시킬 수도 있다 (IGW 말고)

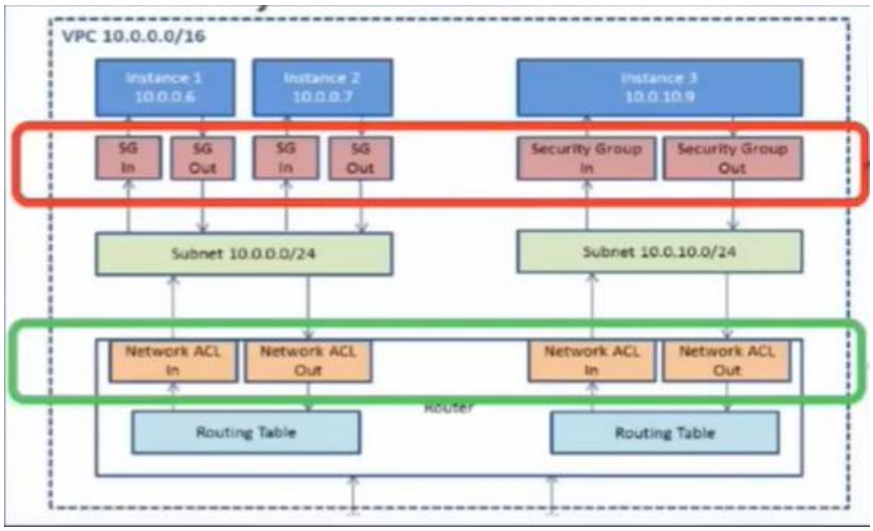
-ipv6는 IGW가 필요없고 직접적으로 통신한다.

-ENI(Elastic network interface) : 네트워크 관리존을 분리하기 위해

<DHCP option set>
-IP풀을 주고 DHCP 서버와 통신하여 여유있는 IP를 자동으로 할당, 도메인 정보, DNS 정보를 호스트가 받게 한다.
-VPC 내에서는 기본적인 DHCP 옵션셋의 default 옵션셋이 있고, 아마존이 제공하는 provided DNS가 자동으로 세팅되어 있다. AWS에서는 도메인 기반으로 서비스가 연동된다. 내가 VPC 내에서 private host zone이라는 사설 도메인을 새로 만들고 이 도메인가지고 연동을 하겠다 하면, 아마존이 제공하는 provided d n s 에 등록이 된다. (아마존이 제공하는 서비스를 이용하는데 어려움이 있다)
-NTP 서버(Network Time Protocol) : network상에 연결된 장비와 장비 간에 시간 정보를 동기화 하기 위한 프로토콜
-아마존이 제공하는 provided d n s 서버는 d n s 의 전체 아이피 대역중에 네트워크 IP의 + 2번이 디폴트로 d n s 역할을 한다. 사용자가 VPC 내에 사설 도메인을 만들면 개를 통해서만 lookup이 가능해 진다. (가급적이면 사설 도메인은 안 만드는게 나음)

<VPC NEWTWORK SECURITY>
VPC 내에서의 보안이라 하면 access 보안이다.

(VPC의 instance에 접근을 할 수 있는지 없는지)



외부에서 VPC내로 통신을 하게 되면 Network ACL in -> SG in, 내부에서 밖으로 나가는 것은 SIG out->Network ACL out으로 나가게 된다

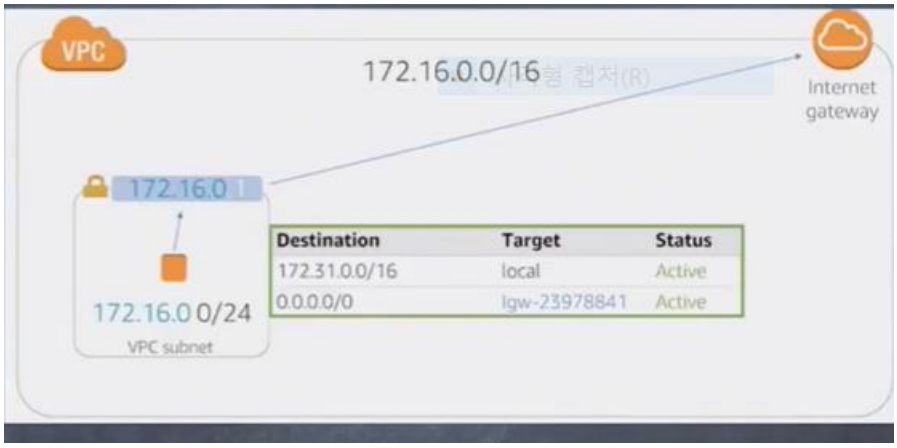
1) **Security group(SG)** : 방화벽 이랑 비슷함
-stateful : 통신의 상태를 가지고 있는 테이블 (allow가 되어 있냐)
Ex) 외부에서 TCP connection을 위해 syn을 보내면 SG의 access 가능하도록 allow가 되어 있으면 통신할 수 있다. Ack 패킷은 syn에 대한 응답패킷이므로, 지나간 정보를 간직하고 있는 state table을 참조해서 바로 보낼 수 있다.
-Instance 바로 앞 단에서 설정이 된다.
-allow할 것만 지정을 한다.

2) **Network ACL(NACL)** : packet by packet 정책에 따라
-stateless : in과 out에 대해 별도의 패킷으로 동작한다 (SG같은 경우 응답패킷은 자동으로 열어줬는데 NACL은 이마저도 허용을 시켜줘야 한다)
-서브넷 별로 동작된다.
-기본 세팅은 모두 allow이지만, 어떤 것을 deny할지를 결정한다.

VPC Flow log : 허용이 잘 되고 있는지 통신이 잘 안되었을 때 이것을 통해서 알 수 있다.
Netflow랑 비슷.

<Routing table>
-AWS VPC 내에는 묵시적인 라우터가 있다. 라우터 1개가 VPC network을 local로 가지고 있다.
-subnet 별로 할당이 된다 (객체로 관리)
(서브넷에 있는 instance가 해당 서브넷의 라우팅 테이블에 의해 경로가 결정된다)

-해당 서브넷의 디폴트 게이트웨이 (1번)이 쓰인다.



VPC를 통해서 나갈 수 있는 경로는 무수히 많다.
-라우팅 테이블에 prefix 우선순위가 있다.
1) VPC Local Route (우선순위 최강좌)
2) Longest prefix match first
-라우터가 ip패킷을 포워딩(라우팅)할 때, 포워딩 테이블에서 해당 항목을 찾는 규칙
-ip패킷의 목적지 ip주소가 라우팅 테이블에 있는 수많은 목적지 ip 주소 중 일치하는 부분이 가장 긴 곳으로 포워딩(라우팅)하는 규칙
3) Static
4) Dynamic (BGP)
DX BGP -> VPN STATIC -> VPN BGP

NEXT HOP
-IGW, NAT GW, VGW, GW endpoint, ENI, peering connection, TGW
-NAT GW도 IGW를 통해서 인터넷에 연결이 됨
-Eni : 아마존이 제공하는 GATEWAY가 아니고 또다른 인스턴스를 NEXT HOP으로 설정(EX.방화벽)
-Peering connection : 다른 VPC로 전달될 수 있는데, 이 때 연결된 peering connection가 Next hop 가능

*인바운드 트래픽 : 외부에서 가상서버 내부로 데이터가 유입될때의 트래픽
아웃바운드 트래픽 : 내부에서 외부로 데이터가 나갈 때

IGW : 인터넷으로 나가는 관문 역할(아마존이 관리하는 게이트웨이)
(public ip를 가진 instance들이 이 곳에서 NAT가 일어남)
실질적으로, 인터넷에서 사용되는 public ip는 여기서 변환이 일어남

NAT GW : 아웃바운드의 보안적인 측면에서 많이 사용.

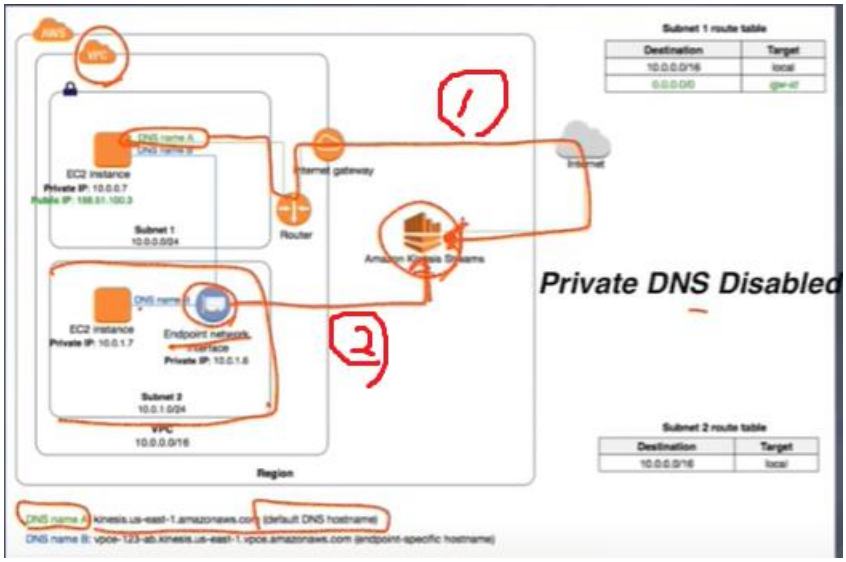
(NAT GW가 사용하는 서브넷은 별도로 만들 수도 있음)

-NAT GW를 바라보는 서브넷도 많고, 해당 서브넷내에 인스턴스가 많고 외부의 특정 IP 와 통신을 자주하면 포트가 65500개로 정해져 있는데 실제로는 55000개 정도 된다.
(PAT 라서 문제가 생길 수가 있다) -> NAT GW를 분리해야 함
-NAT GW의 아웃바운드는 IGW로 매핑이 되어있다. (인터넷 나가는 용도로만)
(NAT GW를 통해서 VPC PEERING, END POINT, VPN 로 가는건 안된다는 의미임)

<VPC에서 통신하는 방법 중 하나인 END POINT>
-VPC 자원들이 AWS 의 다른 서비스들과 통신을 할 때, 대부분 AWS 서비스들이 PUBLIC한 영역에서 도메인 기반으로 서비스를 많이 한다. 그래서, vpc 내에 있는 인스턴스들이 서비스를 받을려면 IGW를 통해 나가서 AWS 도메인 기반의 서비스를 이용해야 한다. 그러나, PRIVATE 망들끼리 별도로 통신할 수 있게 하는게 END POINT 이다.
(즉, public으로 나가지 않고 private vpc 들 끼리 통신할 수 있게 하는 것임)

1) **INTERFACE** 형태로 END POINT : ENI 방식으로 연동이 된다.
AMAZON KINESIS
네트워크 인터페이스가 vpc subnet에 하나가 할당이 되고, 이 아이피를 통해서 외부에 있는 aws 서비스, 타 사업자들의 서비스를 이용하는 루트가 생긴다.

-내부에 있는 아마존 서비스들과도 직접 통신을 하지만, 주로 AWS 내의 다른 VPC 한테 서비스를 제공하는 경우에 많이 사용한다.



1번은 인터넷으로 경로가 유지가 되고, 2번은 서브넷 내에 Endpoint network를 통해서 바로 KINESIS 와 연동이 가능하다
-Interface endpoint 같은 경우는 도메인이 따로 하나가 생성됨 (ENDPOINT -specific dns

hostname)

-endpoint policies가 없다. 단순히, 랜 카드가 하나 연결되서 인터페이스가 연동된다는 컨셉임
-만약에 1번에서 밑에 만들어 놓은 end point를 통해서 바로 KINESIS와 연동을 하게 하려면 amazon **route 53** dns private hosted zone의 서비스를 가지고 만든 도메인이라고 하면, private dns을 enable 시키면 기존에 서비스하는 default public dns name으로 가지고도 바로 접속이 가능하다.

대신에 security group은 정할 수 있음

2) **GATEWAY** 형태로 END POINT
AMAZON S3
AMAZON DYNAMO DB ->

-multiple AZ 존을 지원한다. (vpc내의 라우터 같은 개념이라고 보면 댐)
(Interface 형태의 End point 같은 경우는 AZ 별로 별도로 구축해야 함)
-End point 정책을 가지고 있다.

-AWS Direct Connect나 VPN으로 바로 접속은 안 됨

*묵시적 게이트웨이 (1번) 라우팅 테이블은 서브넷 별로 할당된다.
*NAT 라우터는 아웃바운드 처리량에 따라서 비용처리가 된다

<VPC peering> : VPC 간에 통신할 수 있는 방법
AWS가 VPC 간에 통신할 수 있게끔 별도의 라우팅 테이블로 적용이 되게 하는 방법이다
(Peering connection이라는 형태로 라우팅 테이블에 상대방 VPC IP 대역에 대한 정보가 해당 VPC에 PERRING CONNECTION으로 보내라 라고 하는 형태로 적용된다)
1) Peering request
2) Accept & active
-Region이 달라도 가능하다.
-서로 신뢰관계가 있을 때 보통 vpc peering을 많이 쓴다
그게 아니라면 end point (특정 필요한 대상만 내가 통신을 시키겠다) OR VPN
-peering을 맺고 있는 당사자 끼리만 통신을 할 수 있다 (거쳐서 X)
-특정 VPC 내의 특정 서브넷, 혹은 특정 인스턴스 하나만을 위해 VPC peering을 설정할 수 있다.

(라우팅 테이블만 잘 설정하면 됨)

<ENI (Elastic network interface) > - 가상의 네트워크 랜 카드다.

Network Interfaces

- 기본적으로 인스턴스마다 VPC 대역내의 사설 IP 가 할당된 network interface 가 있음.
- 사용자가 ENI(Elastic Network Interface)를 추가할 수 있음.
- 기본 사설 IP 외에 추가적인 사설 IP 를 가질수 있음.
- eth0 으로 이용하는 network interface 에 한개의 자동할당된 공인 IP 를 이용할 수 있음.
- 사설 IP 하나당 Elastic IP 를 가질수 있음.
- 하나이상의 security group 을 가질수 있음.
- MAC 주소가 있음.
- ENI 를 다른 인스턴스에 재할당할 수 있음. 이때 ENI 에 할당된 설정들이 ENI 를 따라감.

Network Interfaces: ENI(Elastic Network Interface)를 생성하고 EC2 인스턴스에 장착합니다.

- EC2 인스턴스에 ENI 를 여러 개 장착할 수 있고, 장착과 제거를 유동적으로 할 수 있습니다.

EC2 인스턴스가 속하지 않은 다른 VPC 와 통신을 하고 싶을 때 ENI 를 사용합니다.

- 공개망과 사설망을 함께 연결하고 싶을 때
- EC2 인스턴스에서 대역폭이 다른 네트워크를 2 개 가지고 싶을 때

<Network performance>

기본적인 가상화 환경에서 호스트의 Network performance를 보장하는 방법

1) VM에 virtual NIC가 설정이 되고, 중간에 Virtual swtich같은 것이 있는데 이것을 통해서 physical NIC를 통해서 외부로 나가게 되는데, Virtual switch를 포함한 I/O를 다른 VM들과 공유하는 형태이다. 이 I/O를 보장해 주는 방식

2) VM들이 밑단에 있는 하드웨어 상으로 CPU, 메모리들을 공유하면서 네트워크 패킷처리를 하게 된다. 따라서, 패킷처리를 많이 하는 특정 VM이 같은 하드웨어 상에 떠있으면 다른 VM들의 성능에 영향을 받게 된다.

네트워크 프로세싱을 위한 별도의 CPU의 코어를 할당해서 네트워크 프로세싱만을 위해 사용하는 용도가 INTEL의 DPDK(CPU)이다. (네트워크 처리를 위한 CPU 분리)

-가상화 환경(VM) 에서 네트워크 성능이 인스턴스 타입에 따라서 따르다.

-network performance를 maximize 하기 위해서는 Enahnced networking 의 여부를 확인 해야 하고 아마존이 제공하는 ENA(Elastic network adapter)를 사용했을 때는 max 25G까지 나온다 (표가 따로 있음)
-인스턴스들은 EBS(Elastic block store)라는 AWS내의 일종의 하드디스크를 사용한다.
따라서, EBS에 대해서 optimized된 throughput을 보장하는 기능들을 제공한다.
-placement group :
1) 클러스터 배치 그룹

클러스터 배치 그룹은 단일 가용 영역 내에 있는 인스턴스의 논리적 그룹입니다. 클러스터 배치 그룹은 동일한 리전의 피어링된 VPC 에 걸쳐 적용될 수 있습니다. 동일한 클러스터 배치 그룹의 인스턴스는 TCP/IP 트래픽에 더 높은 흐름당 처리량 제한을 제공하며 네트워크의 동일한 높은 양방향 대역폭 세그먼트에 배치됩니다.

다음 이미지는 클러스터 배치 그룹에 배치되는 인스턴스를 보여줍니다.

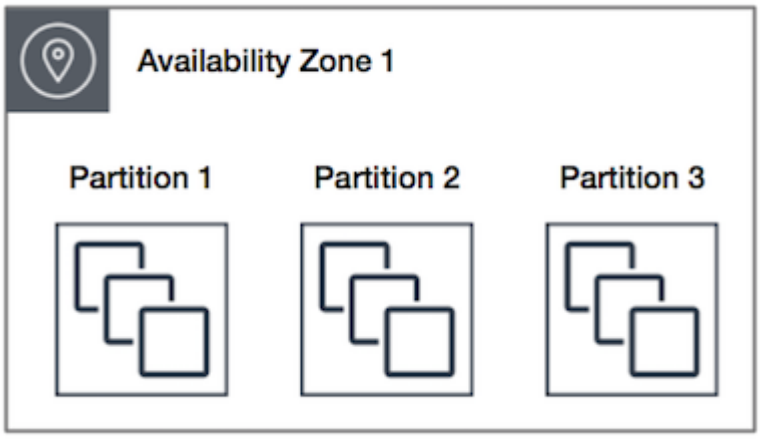
클러스터 배치 그룹은 짧은 네트워크 지연 시간, 높은 네트워크 처리량 또는 둘 다의 이점을 활용할 수 있는 애플리케이션에 권장됩니다. 또한 대부분의 네트워크 트래픽이 그룹 내 인스턴스 간에 전송되는 경우에도 권장됩니다.

2) 파티션 배치 그룹

- *파티션* – 인스턴스를 논리적 파티션에 분산해, 한 파티션에 있는 인스턴스 그룹이 다른 파티션의 인스턴스 그룹과 기본 하드웨어를 공유하지 않게 합니다. 이 전략은 일반적으로 Hadoop, Cassandra, Kafka 등 대규모의 분산 및 복제된 워크로드에 필요합니다.
- 각 랙은 자체 네트워크 및 전원이 있습니다. 배치 그룹 내 두 파티션이 동일한 랙을 공유하지 않으므로 애플리케이션 내 하드웨어 장애의 영향을 격리시킬 수 있습니다.

    - 다음 이미지는 단일 가용 영역에 있는 파티션 배치 그룹을 시각적으로 간단하게 표현한 것입니다. 여기서는 세 개의 파티션인 **파티션 1**, **파티션**

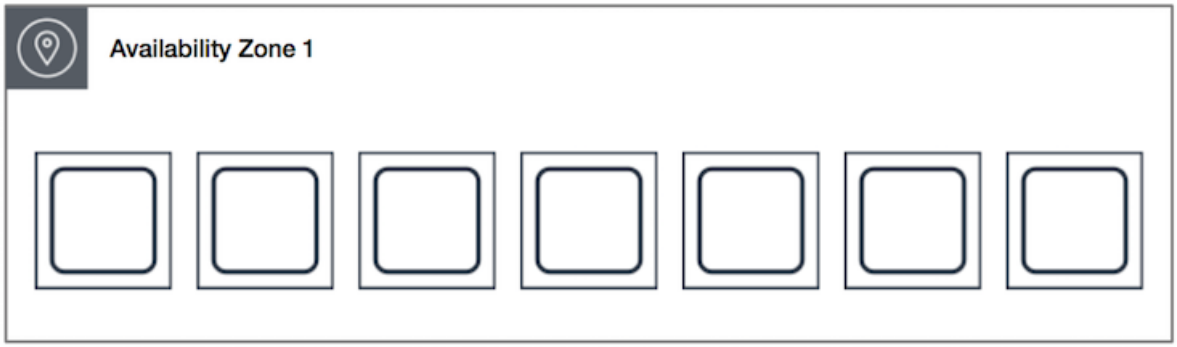
2 및 **파티션 3**이 있는 파티션 배치 그룹에 배치된 인스턴스를 보여줍니다. 각 파티션은 여러 인스턴스로 구성됩니다. 각 파티션에 있는 인스턴스는 다른 파티션에 있는 인스턴스와 랙을 공유하지 않기 때문에 단일 하드웨어 장애의 영향을 관련 파티션으로만 국한할 수 있습니다.



3) 분산형 배치 그룹: 다른 하드웨어로 분산해서 상호의존성을 아예 없앤다

분산형 배치 그룹은 각각 고유한 랙에 배치된 인스턴스 그룹이며 랙마다 자체 네트워크 및 전원이 있습니다.

다음 이미지는 분산형 배치 그룹에 배치되는 단일 가용 영역에 있는 인스턴스 7 개를 보여줍니다. 7 개의 인스턴스는 7 개의 서로 다른 랙에 배치됩니다.



<placement group>
-인스턴스를 배치를 할 때 3가지 유형으로 AWS에서 지원한다
1) 클러스터 형 : 동일 렉 상에 인스턴스들을 모아서 배치 ->RTT 최소화
2) 파티션 형 : VPC PEERING이 되어 있는 VPC 간에도 동일한 PLACEMENT GROUP을 적용해서 확장할 수 있다.

<Jumbo frame>
-PATH MTU DISCOVERY : 경로 상에 최대 MTU SIZE가 얼마인지 알아보는 것?
AWS VPC 내부에서는 MTU가 최대한 9000바이트다. 일반적인 네트워크에서는 1500임
(그러나 VPC 내부와 aws 안에 dx 온-프레미스 환경 (physical한 네트워크 장비가 있음) 과 VPC 간의 통신 MTU가 항상 9000바이프로 보장되지는 않는다)
-온 프레미스 환경에서의 한 라우터의 MTU가 1500이면 MTU가 1500으로 될 수 밖에 없음
(온 -프레미스는 MTU를 9000 바이트를 보장하지 않기 때문에)