# EC2 Synchronization

## Networking

AWS presents a natural containment hierarchy for many of the resources making up a VPC network.

┌───────────────────────────────────────────┐  
│Region │  
│ ┌──────────────────────────────────┐ │  
│ │VPC │ │  
│─ ┼ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─│─ ─ ─│  
│ │ ┌──────────────────────┐ │ │  
│ │ │Subnet │ │ │  
│ │ │ ┌─────────┐.─. .┴. │ .┴────. .───────.  
│ │ │ │Instance (SG )────(ACL)──┐ │┌──( IGW )────────▶Internet )  
│ │ │ └─────────┘`─' `┬' │ ││ `┬────' `───────'  
│AZ│ └──────────────────────┘ .─────. ││ │  
├ ─│─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─( RTR )┼┘─ ─ ┤  
├ ─│─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ `─────' ┼ ─ ─ ┤  
│ │ ┌──────────────────────┐ │ │ │  
│ │ │Subnet │ │ │ │  
│ │ │ ┌─────────┐.─. .┴. │ │ │  
│ │ │ │Instance (SG )────(ACL)──┘ │ │  
│ │ │ └─────────┘`─' `┬' │ │  
│AZ│ └──────────────────────┘ │ │  
│─ ┼ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─ ─│─ ─ ─│  
│ └──────────────────────────────────┘ │  
└───────────────────────────────────────────┘

Here is the graph built for JupiterOne:

.─────.  
 ( EC2 )  
 `─────'  
 │  
 │  
 ┌──────────┬──────────┬─────┴────┬──────────┬──────────┐  
 │ │ │ │ │ │  
 HAS HAS HAS HAS HAS HAS  
 .───▼───. .───▼───. .───▼───. .───▼───. .───▼───. .───▼───.  
( VPC )( Subnet )(Instance )( IGW )( ACL )( SG )  
 `───────' `───────' `───────' `───────' `───────' `───────'  
  
  
 .─────.  
 ┌───────────┬── VPC ──┬───────────┐  
 │ │ `─────' │ │  
 │ │ │ │  
 │ │ │ │  
 CONTAINS CONTAINS HAS HAS  
 .───▼───. .───▼───. .───▼───. .───▼───.  
 ( Subnet ) ( IGW ) ( ACL ) ( SG )  
 `───┬───' `───────' `───────' `───────'  
 │  
 │  
 HAS  
 .───▼───.  
 (Instance )  
 `───────'  
  
 .───────.  
 ┌─ALLOWS─▶ Network )  
 .───────. .───────.────────┘ `───────'  
( Subnet ◀───PROTECTS─── ACL )  
 `───────' `───────'◀───────┐ .───────.  
 └─DENIES── Network )  
 `───────'  
  
 .───────.  
 ┌─ALLOWS─▶ Network )  
 .───────. .───────.────────┘ `───────'  
(Instance ◀───PROTECTS─── SG )  
 `───────' `───────'◀───────┐ .───────.  
 └─ALLOWS── Network )  
 `───────'

There are also relationships across the hierarchy between resources.

Routing tables specify the allowed routes for outbound traffic leaving the subnet. Security groups are stateful host firewalls, network ACLs are stateless subnet firewalls. A host can reach a network only when a route exists and packets are allowed by a security group and a network ACL when the network is outside the host subnet.

### AWS API

The AWS API provides containment relationship information in the contained resource. For example, to determine the subnet of an instance, describe-instances includes with each instance the SubnetId.

Vpcs [  
 {  
 VpcId  
 IsDefault  
 CidrBlock  
  
 CidrBlockAssociations: [  
 { CidrBlock }  
 ]  
 }  
]

Subnets [  
 {  
 VpcId  
 SubnetId  
 CidrBlock  
 }  
]

Instances [  
 {  
 VpcId  
 SubnetId  
  
 NetworkInterfaces.Groups [  
 { GroupId }  
 ]  
  
 SecurityGroups [  
 { GroupId }  
 ]  
 }  
]

SecurityGroups can be used. Information about the interfaces is not used to build the graph. Instances that do not define any security groups will be assumed to belong to the default security group of the VPC.

RouteTables [  
 {  
 RouteTableId  
 VpcId  
  
 Associations [  
 {  
 SubnetId  
 Main: true | false  
 }  
 ]  
  
 Routes [  
 {  
 GatewayId: local | igw-xxx | vgw-xxx  
 DestinationCidrBlock  
 State: active | blackhole  
 }  
 ]  
 }  
]

A route table with an association having Main: true signifies that the route table is the main table for the containing VPC. There will be no SubnetId in this case. Determining which subnets use the main route table requires processing all route tables in the VPC to discover which subnets have no entry in any Associations.

NetworkAcls [  
 {  
 VpcId  
 IsDefault  
  
 Associations [  
 { SubnetId }  
 ]  
  
 Entries [  
 {  
 CidrBlock  
 Ipv6CidrBlock  
 RuleNumber  
 Protocol: '-1' | 'tcp' | 'udp' | 'icmp' | '<protocol number>'  
 Egress: true | false  
 RuleAction: allow | deny  
 }  
 ]  
 }  
]

Associations are used to create PROTECTS relationships to subnets.

SecurityGroups [  
 {  
 VpcId  
  
 IpPermissions | IpPermissionsEgress [  
 {  
 IpProtocol: '-1' | 'tcp' | 'udp' | 'icmp' | '<protocol number>'  
 FromPort  
 ToPort  
 IpRanges [  
 { CidrIp }  
 ]  
 Ipv6Ranges [  
 { CidrIpv6 }  
 ]  
 }  
 ]  
 }  
]

Child resources indicate their relationship to a parent. Resources are therefore generally processed by creating entities top down, with relationships from parent to child being created after the child entity.

Other relationships are built later by processing the graph of entities and using additional information as necessary.

### Analyzing Network Access Controls

A number of questions about AWS EC2 network resources and their relationships can be answered by the graph today. More complex analysis is not currently implemented by JupiterOne.

Consider a very simple question: *What networks can be reached by this host?* This graph, built by processing security group rules and relationships, would at first appear to provide the answer:

.───────.  
 ┌─ALLOWS─▶ Network )  
 .───────. .───────.────────┘ `───────'  
(Instance ◀───PROTECTS─── SG )  
 `───────' `───────'◀───────┐ .───────.  
 └─ALLOWS── Network )  
 `───────'

The J1QL query would be:

FIND Firewall AS f  
THAT PROTECTS Host WITH cidr='55.1.0.4' AS h  
THAT ALLOWS Network AS n  
RETURN h.displayName AS Host, n.displayName AS Network, n.cidr AS CIDR

What happens when the instance is associated with multiple security groups and one of them denies the same network allowed by another group? What if the network ACL attached to the subnet of the instance blocks the traffic, or perhaps the network ACL of the destination network does not allow it?

See that the question as posed cannot be meaningfully answered! This does not make our graph useless. This question provides meaningful information: *What networks are allowed by security groups associated with this host?*

### Networks as Global Resources

In an ideal world, every subnetwork of the Internet could be uniquely identified by the network CIDR. However, there are private subnetwork ranges for a reason: there aren’t enough IPV4 addresses to go around.

AWS subnets are unique - they have an ARN - though two or more subnets may have the same private CIDRs. This is acceptable when the subnets are not in any way connected, otherwise it is likely a misconfiguration.

When processing EC2 resources in different regions or VPCs which reference hosts or networks in other regions or VPCs, the relationships should point to the single representation of the referenced resource.

Networks outside the ingested AWS environments are be created by the integration directly but through mapper operations instead.