TenantGuard: Scalable Runtime Verification of Cloud-Wide VM-Level Network Isolation

Han Song

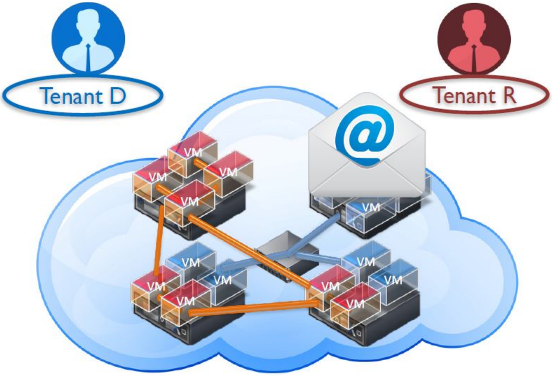
SJTU

May 24, 2017

Outline

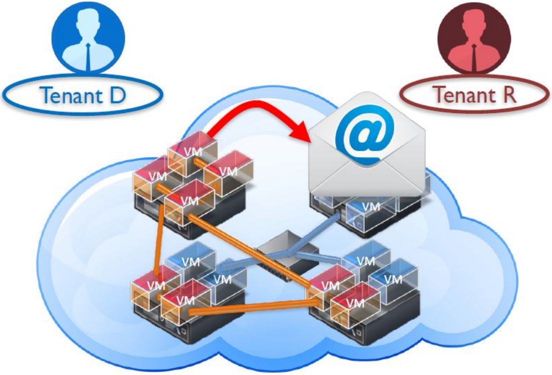
Isolation Breaches

One of the Biggest Security Concerns in Cloud



Isolation Breaches

One of the Biggest Security Concerns in Cloud

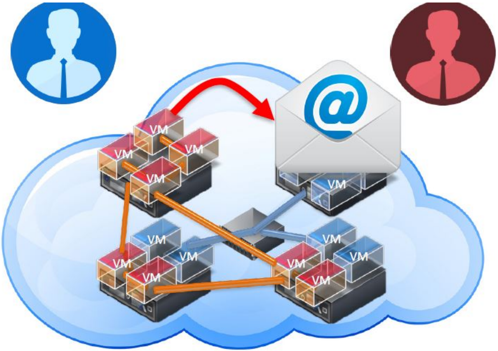


Something went wrong and D is hacked!

Isolation Breaches

One of the Biggest Security Concerns in Cloud

OpenStack real world vulnerabilities



[OSSA 2014-008]

Any tenant is able to create a port on another tenant's router!

Reported: 22.10.2013
Fixed: 27.03.2014

[OSSA 2015-021]

Security group rules are not effective on instances immediately!

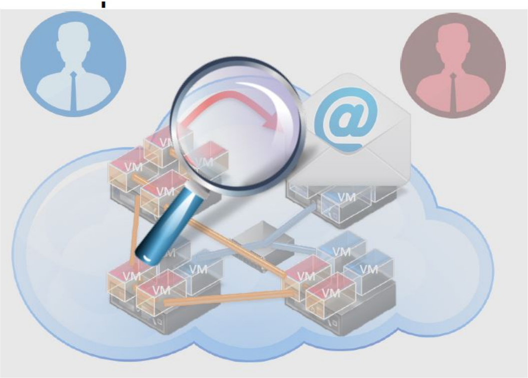
Reported: 02.09.2015
Fixed: 11.09.2015

More on: https://www.cvedetails.com/vulnerability-list/vendor_id-11727/Openstack.html

Isolation Breaches

One of the Biggest Security Concerns in Cloud

One possible solution is: network isolation verification



[OSSA 2014-008]

Any tenant is able to create a port on another tenant's router!

Reported: 22.10.2013
Fixed: 27.03.2014

[OSSA 2015-021]

Security group rules are not effective on instances immediately!

Reported: 02.09.2015
Fixed: 11.09.2015

Network Isolation Verification

Challenges

- Size of virtual networks: 150M+ VM pairs
- Diverse and distributed network functions
- Large data from heterogeneous sources
- Quickly invalidating verification results

Network Isolation Verification

Challenges

- Size of virtual networks: 150M+ VM pairs
- Diverse and distributed network functions
- Large data from heterogeneous sources
- Quickly invalidating verification results

Network Isolation Verification

Challenges

- Size of virtual networks: 150M+ VM pairs
- Diverse and distributed network functions
- Large data from heterogeneous sources
- Quickly invalidating verification results

Network Isolation Verification

Challenges

- Size of virtual networks: 150M+ VM pairs
- Diverse and distributed network functions
- Large data from heterogeneous sources
- Quickly invalidating verification results

Existing Approaches

- Designed for physical networks
  - Not suitable for VM-level pair-wise reachability
- Focus on small to medium virtual infrastructure
  - Not designed for millions of VM pairs
- Can support VM-level reachability
  - Taking minutes to hours for over 100 million pairs

Existing Approaches

- Designed for physical networks
  - Not suitable for VM-level pair-wise reachability
- Focus on small to medium virtual infrastructure
  - Not designed for millions of VM pairs
- Can support VM-level reachability
  - Taking minutes to hours for over 100 million pairs

Existing Approaches

- Designed for physical networks
  - Not suitable for VM-level pair-wise reachability
- Focus on small to medium virtual infrastructure
  - Not designed for millions of VM pairs
- Can support VM-level reachability
  - Taking minutes to hours for over 100 million pairs

Existing Approaches

- Designed for physical networks
  - Not suitable for VM-level pair-wise reachability
- Focus on small to medium virtual infrastructure
  - Not designed for millions of VM pairs
- Can support VM-level reachability
  - Taking minutes to hours for over 100 million pairs

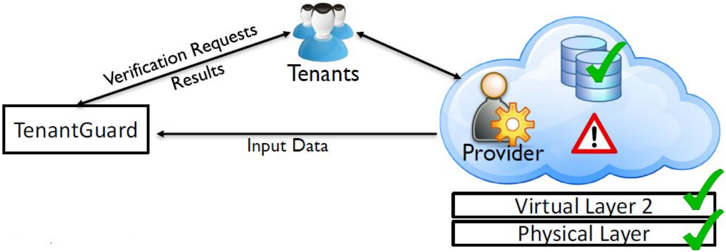
Existing Approaches

- Designed for physical networks
  - Not suitable for VM-level pair-wise reachability
- Focus on small to medium virtual infrastructure
  - Not designed for millions of VM pairs
- Can support VM-level reachability
  - Taking minutes to hours for over 100 million pairs

Existing Approaches

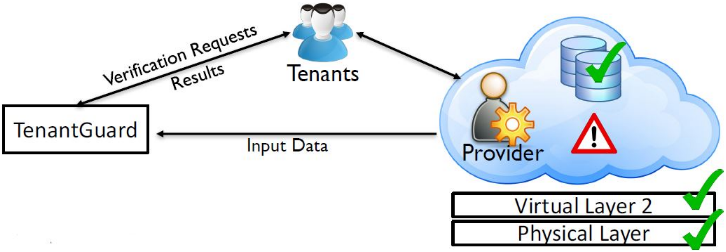
- Designed for physical networks
  - Not suitable for VM-level pair-wise reachability
- Focus on small to medium virtual infrastructure
  - Not designed for millions of VM pairs
- Can support VM-level reachability
  - Taking minutes to hours for over 100 million pairs

Assumptions



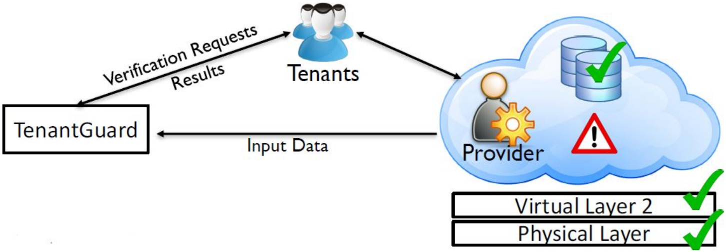
- Focused on:
  - Verifying security properties specified by cloud tenants
  - Not detecting any specific attack
- Relies on:
  - The correctness of input data
  - Existing solutions at other layers
  - No sensitive information in the verification results

Assumptions



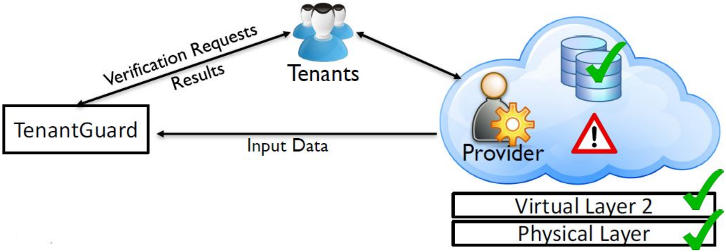
- Focused on:
  - Verifying security properties specified by cloud tenants
  - Not detecting any specific attack
- Relies on:
  - The correctness of input data
  - Existing solutions at other layers
  - No sensitive information in the verification results

Assumptions



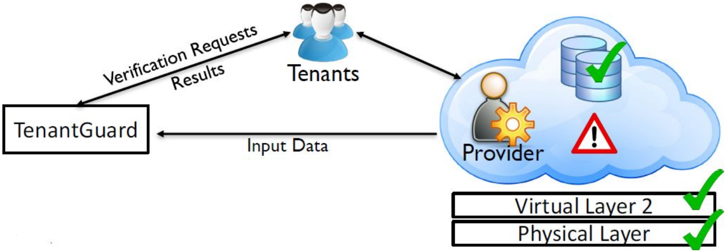
- Focused on:
  - Verifying security properties specified by cloud tenants
  - Not detecting any specific attack
- Relies on:
  - The correctness of input data
  - Existing solutions at other layers
  - No sensitive information in the verification results

Assumptions



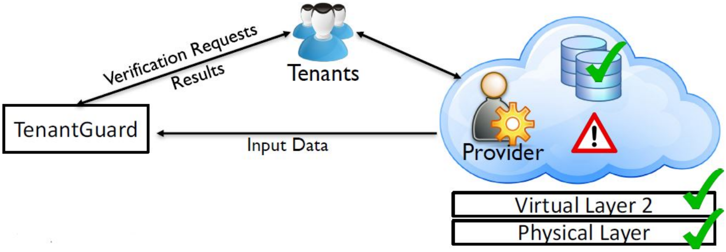
- Focused on:
  - Verifying security properties specified by cloud tenants
  - Not detecting any specific attack
- Relies on:
  - The correctness of input data
  - Existing solutions at other layers
  - No sensitive information in the verification results

Assumptions



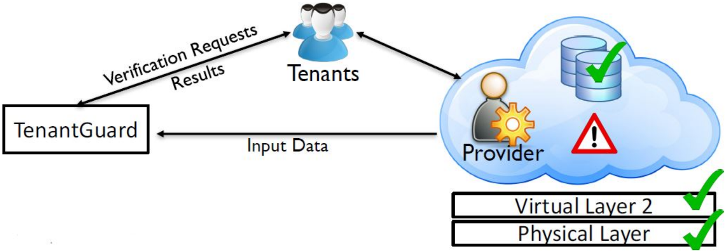
- Focused on:
  - Verifying security properties specified by cloud tenants
  - Not detecting any specific attack
- Relies on:
  - The correctness of input data
  - Existing solutions at other layers
  - No sensitive information in the verification results

Assumptions



- Focused on:
  - Verifying security properties specified by cloud tenants
  - Not detecting any specific attack
- Relies on:
  - The correctness of input data
  - Existing solutions at other layers
  - No sensitive information in the verification results

Assumptions



- Focused on:
  - Verifying security properties specified by cloud tenants
  - Not detecting any specific attack
- Relies on:
  - The correctness of input data
  - Existing solutions at other layers
  - No sensitive information in the verification results

Highlights

TenantGuard, a VM-level network isolation verification system

- Pairwise reachability for over 25K VMs in 13s
- Built on OpenStack, a popular cloud management platform
- Based on a hierarchical model for virtual networks
- Leveraging efficient data structures, incremental verification and parallel computation

Highlights

TenantGuard, a VM-level network isolation verification system

- Pairwise reachability for over 25K VMs in 13s
- Built on OpenStack, a popular cloud management platform
- Based on a hierarchical model for virtual networks
- Leveraging efficient data structures, incremental verification and parallel computation

Highlights

TenantGuard, a VM-level network isolation verification system

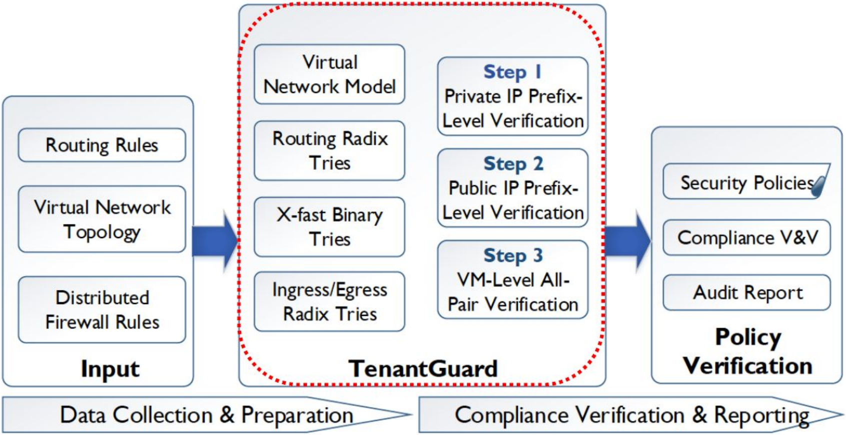
- Pairwise reachability for over 25K VMs in 13s
- Built on OpenStack, a popular cloud management platform
- Based on a hierarchical model for virtual networks
- Leveraging efficient data structures, incremental verification and parallel computation

Highlights

TenantGuard, a VM-level network isolation verification system

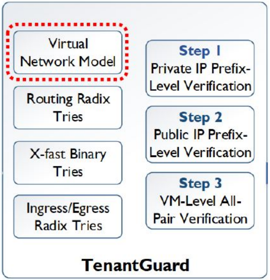
- Pairwise reachability for over 25K VMs in 13s
- Built on OpenStack, a popular cloud management platform
- Based on a hierarchical model for virtual networks
- Leveraging efficient data structures, incremental verification and parallel computation

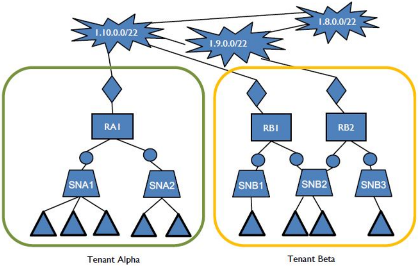
TenantGuard: Architecture



Key Ideas

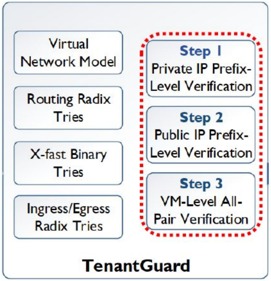
- Hierarchical virtual network model (Router, subnet, VM)
- Top-down verification approach (from prefix-level to IP-level)
- Efficient data structures (Radix Trie and X-fast Binary Trie)

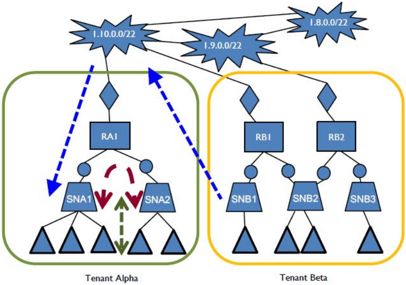




Key Ideas

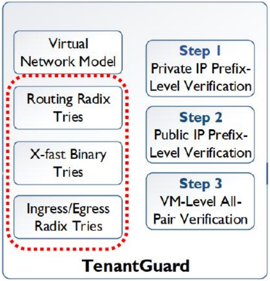
- Hierarchical virtual network model (Router, subnet, VM)
- Top-down verification approach (from prefix-level to IP-level)
- Efficient data structures (Radix Trie and X-fast Binary Trie)

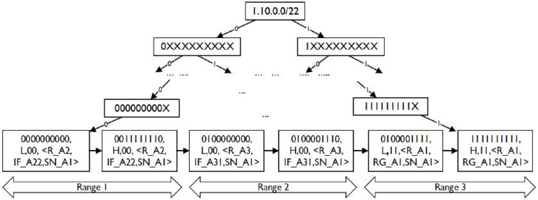




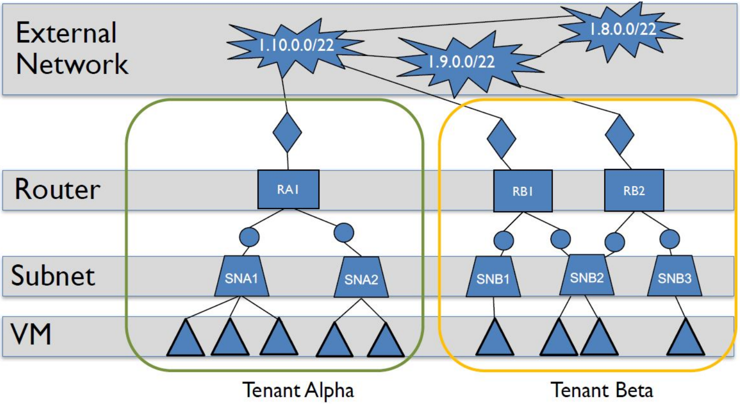
Key Ideas

- Hierarchical virtual network model (Router, subnet, VM)
- Top-down verification approach (from prefix-level to IP-level)
- Efficient data structures (Radix Trie and X-fast Binary Trie)

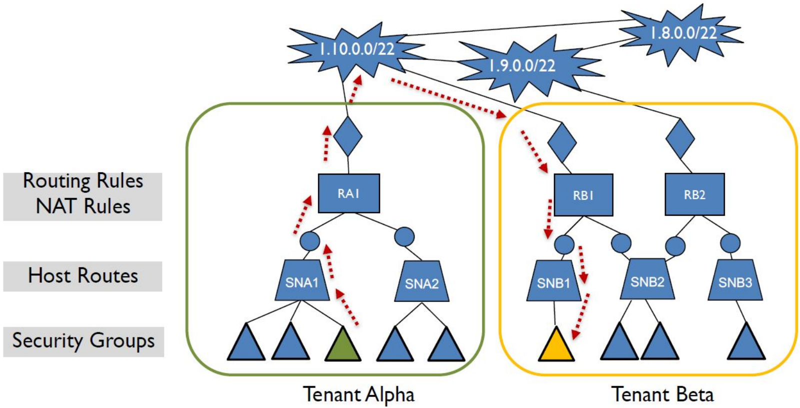




Hierarchical Virtual Network Model

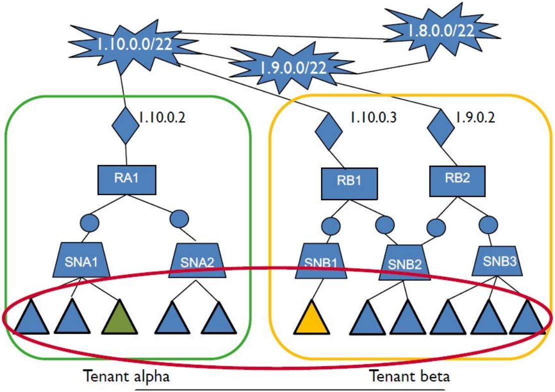


Hierarchical Virtual Network Model



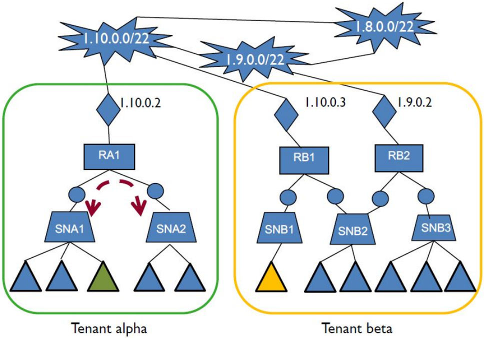
Baseline Approach

Verifying every possible VM pair (e.g., over 150 million pairs!!)



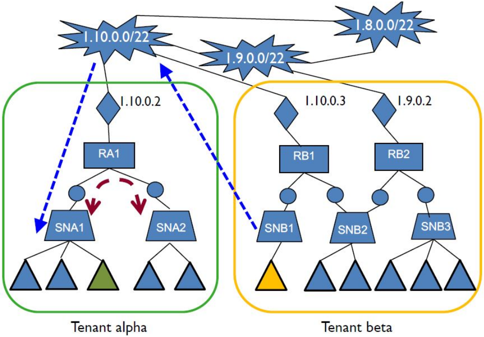
Top-Down Verification

- Step 1: Check isolation between subnets within the same tenant environment
- Step 2:Check isolation between different tenant environments
- Step 3: Check VM-isolation only for subnets found to be reachable



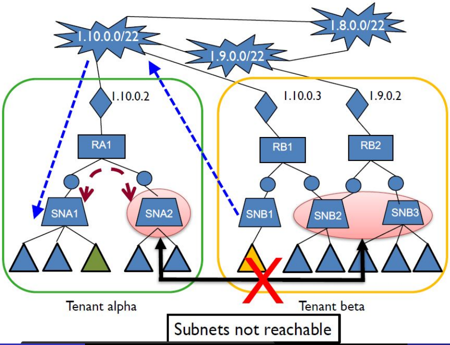
Top-Down Verification

- Step 1: Check isolation between subnets within the same tenant environment
- Step 2:Check isolation between different tenant environments
- Step 3: Check VM-isolation only for subnets found to be reachable



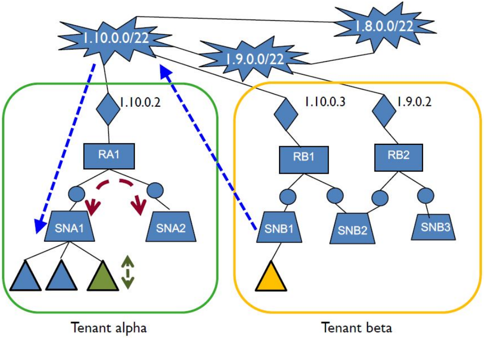
Top-Down Verification

- Step 1: Check isolation between subnets within the same tenant environment
- Step 2:Check isolation between different tenant environments
- Step 3: Check VM-isolation only for subnets found to be reachable



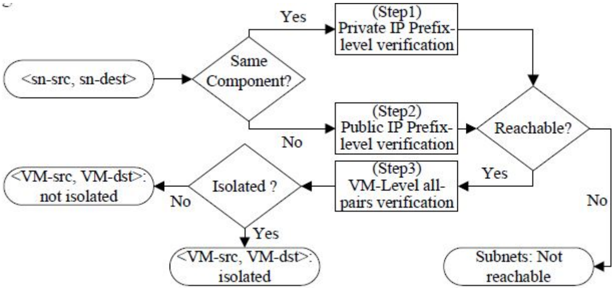
Top-Down Verification

- Step 1: Check isolation between subnets within the same tenant environment
- Step 2:Check isolation between different tenant environments
- Step 3: Check VM-isolation only for subnets found to be reachable



Top-Down Verification

- Step 1: Check isolation between subnets within the same tenant environment
- Step 2:Check isolation between different tenant environments
- Step 3: Check VM-isolation only for subnets found to be reachable



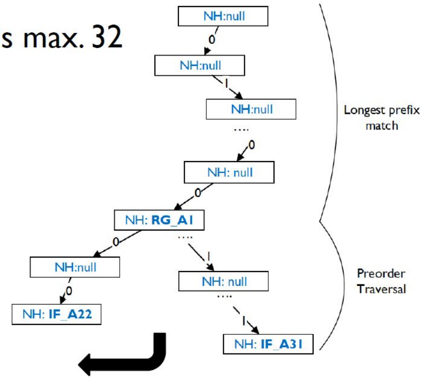
Efficient Data Structure

Radix Trie: Capturing Routing Rules

Matching rule is O(L), here L is max. 32

Rules in Router R_A1

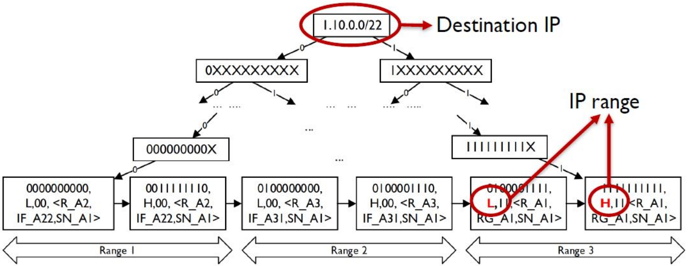
Rule	Prefix	Next-Hop
r0	10.0.1.0/24	IF_A12
r1	1.10.0.0/22	**RG_A1**
r2	1.10.0.0/24	**IF_A22**
r3	1.10.0.0/28	**IF_A31**



Efficient Data Structure

BTries: Storing Intermediary Results

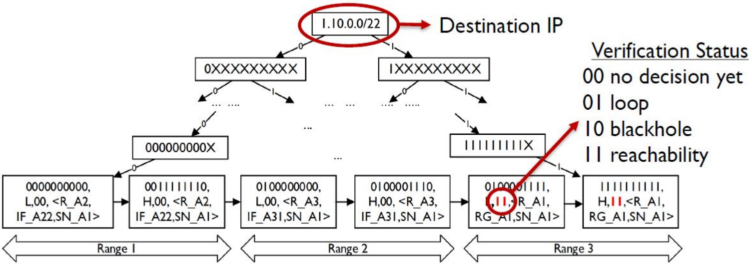
- Storing results of matching routing rules against IP ranges
- Searching is O(logL), here L is max. 32



Efficient Data Structure

BTries: Storing Intermediary Results

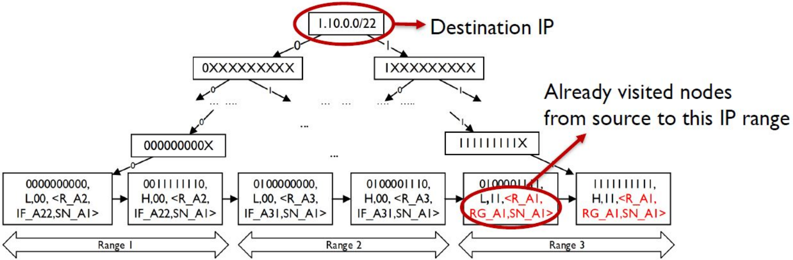
- Storing results of matching routing rules against IP ranges
- Searching is O(logL), here L is max. 32



Efficient Data Structure

BTries: Storing Intermediary Results

- Storing results of matching routing rules against IP ranges
- Searching is O(logL), here L is max. 32



Prefix-to-prefix Algorithm

Algorithm 1 *prefix-to-prefix*$(btrie)$

Input/Output: $btrie$
counter=0
for each range $[L, H]$ in $btrie$.*leafs* with $RLB = 00$ **do**
    $router = get(HR, r\_id)$
    $dst = getroot(btrie)$
    if $searchTries(dst, router) = false$ **then**
        $TempBTrie = Match(RadixTrie(router), dst)$
    else
        $TempBTrie = getBTrie(dst, router)$
    $Copy(btrie, TempBTrie, [L, H])$
    counter = counter + 1
if $counter \neq 0$ **then**
    prefix-to-prefix$(btrie)$

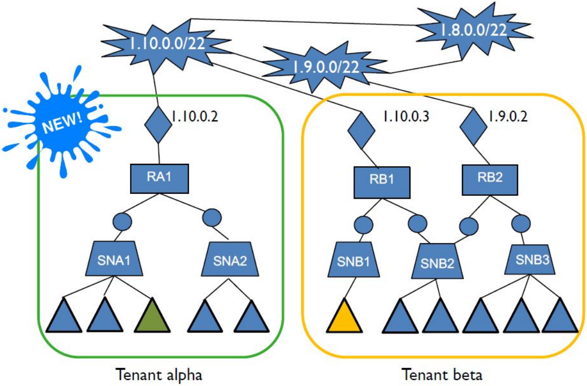
VM-to-VM Algorithm

Algorithm 2 *VM-to-VM*(VM_{src}, VM_{dest})

```
Triepub = getBTrie(VM_dst.publicIP.CIDR, VM_src. subnet_id)
Triepriv = getBTrie(VM_dst.privateIP.CIDR, router_id)
routable = Route-Lookup(Triepub, Triepriv)
if routable = true then
    VerifySecGroups(VM_src, VM_dest)
```

Incremental Verification

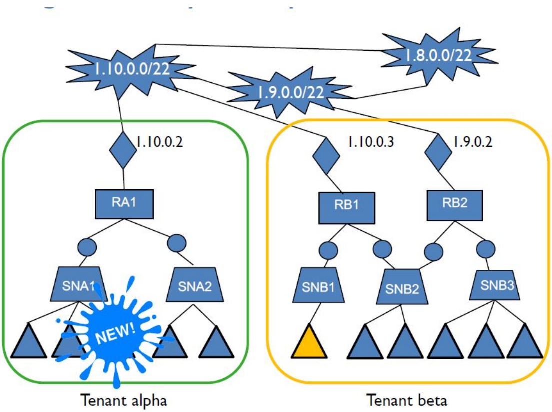
- Graph update
- Radix trie creation/deletion
- Radix trie update
- X-fast trie creation/deletion
- X-fast trie update
- VM-level isolation verification
- Security group verification



Incremental Verification

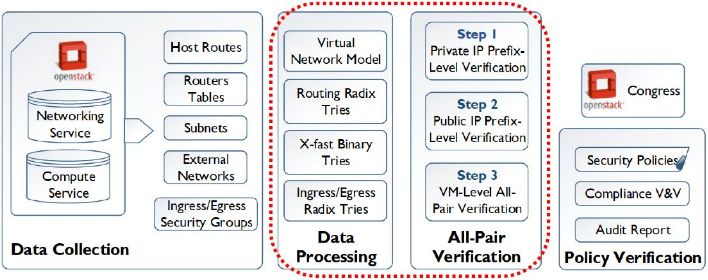
Adding a Security Group

- Graph update
- Radix trie creation/deletion
- Radix trie update
- X-fast trie creation/deletion
- X-fast trie update
- VM-level isolation verification
- Security group verification



Application to OpenStack

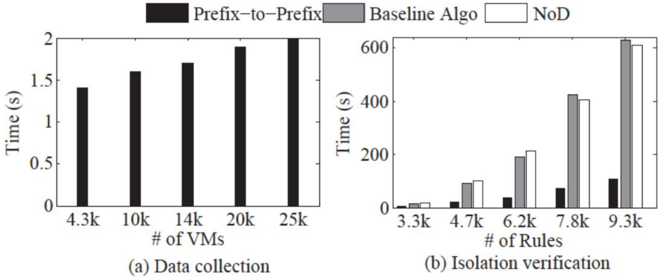
- OpenStack Kilo with one controller and 80 compute nodes
- Parallelization of reachability verification with Apache Ignite
- Integration to OpenStack Congress



Experimental Settings

- Test Environment
  - Two series of datasets
    - SNET (represents small to medium networks)
    - LNET (represents large networks)
  - NoD (NSDI15) and a baseline algorithm
- Real Cloud
  - Ericsson research cloud
  - Mainly to evaluate the real world applicability of TenantGuard
  - Only observed a minor incompatibility issue due to version mismatch

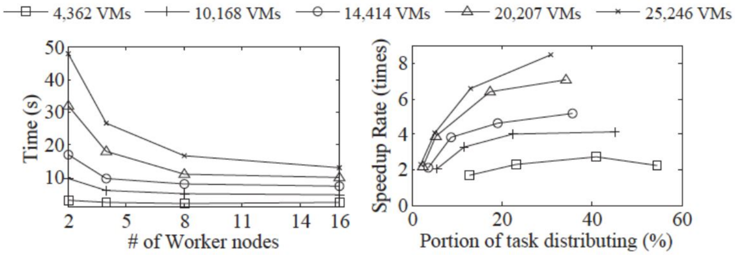
Performance Evaluation



Data collection and processing time vary from 1.5 to 2 seconds

TenantGuard performs 82% faster than the baseline

Further Performance Improvement



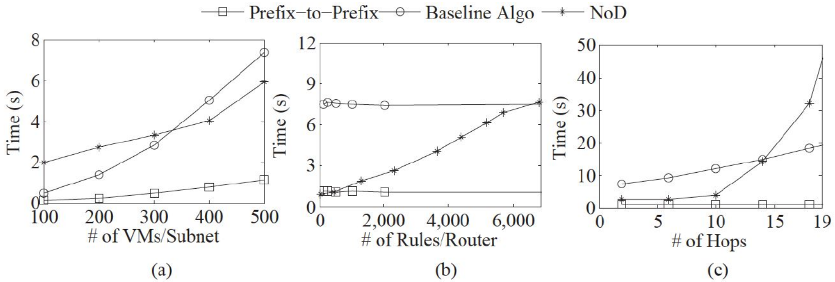
(a) Parallel Mode

(b) Speedup Analysis

Reachability between 168 millions VM pairs in 13 seconds

Relationship between cluster size and speedup gain

Identifying Performance Factors



Number of VMs and hops have less effects due to the reduced complexity and design

Number of routing rules has almost no effect due to the use of Radix and X-fast tries

Conclusion

- Future Work
  - Integrating existing tools at other layers (physical, L2)
  - Ensuring integrity of input data
  - Addressing privacy issues from the verification results
- Summary
  - TenantGuard, a VM-level network isolation verification system
  - Integrated our approach to OpenStack
  - Reachability for over 150 million VM pairs in 13 seconds

Thank You!
Q & A?