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

Y. Wang¹, T. Madi¹, S. Majumdar¹, Y. Jarraya²,
A. Alimohammadifar¹, M. Pourzandi², L. Wang¹ and M. Debbabi¹

¹ Concordia University, Canada, ² Ericsson Security Research, Canada
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
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 word 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

1. Size of virtual networks: 150M+ VM pairs*

2. Diverse and distributed network functions
   (L3/4 functions including virtual routing, NATing, firewalling)

3. Large data from heterogeneous sources

4. Quickly invalidating verification results

* OpenStack user survey, 2016. Available at: https://www.openstack.org
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

Focuses 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
TenantGuard: Architecture

Input:
- Routing Rules
- Virtual Network Topology
- Distributed Firewall Rules

TenantGuard:
- Virtual Network Model
- Routing Tries
- X-fast Binary Tries
- Ingress/Egress Radix Tries

Step 1:
Private IP Prefix-Level Verification

Step 2:
Public IP Prefix-Level Verification

Step 3:
VM-Level All-Pair Verification

Policy Verification:
- Security Policies
- Compliance V&V
- Audit Report

Data Collection & Preparation

Compliance Verification & Reporting
TenantGuard: Architecture

Input
- Routing Rules
- Virtual Network Topology
- Distributed Firewall Rules

TenantGuard
- Virtual Network Model
- Routing Tries
- X-fast Binary Tries
- Ingress/Egress Radix Tries

Step 1
Private IP Prefix-Level Verification

Step 2
Public IP Prefix-Level Verification

Step 3
VM-Level All-Pair Verification

Policy Verification
- Security Policies
- Compliance V&V
- Audit Report

Data Collection & Preparation

Compliance Verification & Reporting
Key Ideas

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

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

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

**External Network**
- 1.10.0.0/22
- 1.9.0.0/22
- 1.8.0.0/22

**Tenant Beta**
- SNB1
- SNB2
- SNB3

**Tenant Alpha**
- SNA1
- SNA2

**Router**
- RA1
- RB1
- RB2

**Subnet**
- SNA1
- SNA2
- SNB1
- SNB2
- SNB3

**VM**
Hierarchical Virtual Network Model

Routing Rules
NAT Rules
Host Routes
Security Groups
Baseline Approach

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

Step one
Check isolation between subnets within the same tenant environment

Step two
Check isolation between different tenant environments

Step three
Check VM-isolation only for subnets found to be reachable
Top-Down Verification

**Step one**
Check isolation between subnets within the same tenant environment

**Step two**
Check isolation between different tenant environments

**Step three**
Check VM-isolation only for subnets found to be reachable
### Step one
Check isolation between subnets within the same tenant environment

### Step two
Check isolation between different tenant environments

### Step three
Check VM-isolation only for subnets found to be reachable

**Tenant alpha**

**Tenant beta**

Subnets not reachable
Top-Down Verification

**Step one**
Check isolation between subnets within the same tenant environment

**Step two**
Check isolation between different tenant environments

**Step three**
Check VM-isolation only for subnets found to be reachable
Efficient Data Structure
Capturing Routing Rules

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

<table>
<thead>
<tr>
<th>Rule</th>
<th>Prefix</th>
<th>Next-Hop</th>
</tr>
</thead>
<tbody>
<tr>
<td>r0</td>
<td>10.0.1.0/24</td>
<td>IF_A12</td>
</tr>
<tr>
<td>r1</td>
<td>1.10.0.0/22</td>
<td>RG_A1</td>
</tr>
<tr>
<td>r2</td>
<td>1.10.0.0/24</td>
<td>IF_A22</td>
</tr>
<tr>
<td>r3</td>
<td>1.10.0.0/28</td>
<td>IF_A31</td>
</tr>
</tbody>
</table>
Efficient Data Structure
Storing Intermediary Results

• Storing results of matching routing rules against IP ranges

• Searching is $O(\log L)$, here $L$ is max. 32
Efficient Data Structure
Storing Intermediary Results

• Storing results of matching routing rules against IP ranges

• Searching is $O(\log L)$, here $L$ is max. 32
Efficient Data Structure
Storing Intermediary Results

• Storing results of matching routing rules against IP ranges
• Searching is $O(\log L)$, here $L$ is max. 32
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

Tenant alpha

Tenant beta

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

NEW!
Incremental Verification
Adding a Security Group
Application to OpenStack

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

TenantGuard performs 82% faster than the baseline.

Data collection and processing time vary from 1.5 to 2 seconds.

Further Performance Improvement

Reachability between 168 millions VM pairs in 13 seconds

Relationship between cluster size and speedup gain
Identifying Performance Factors

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

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

Project webpage: arc.ensc.concordia.ca
Corresponding author: Suryadipta Majumdar (su_majum@ensc.concordia.ca)

Thank you
Backups
Experimental Settings

• Test Environment
  – Two series of datasets
    • SNET (represents small to medium networks)
    • LNET (represents large networks)
  – NoD (NSDI’15) 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