Cyber Testing Tools and Methodologies: Presentation at ITEA

*Our Experience on the Virtual Ad hoc Network (**VAN**) Testbed*

C. Jason Chiang, Alex Poylisher, Yitzchak Gottlieb and Constantin Serban

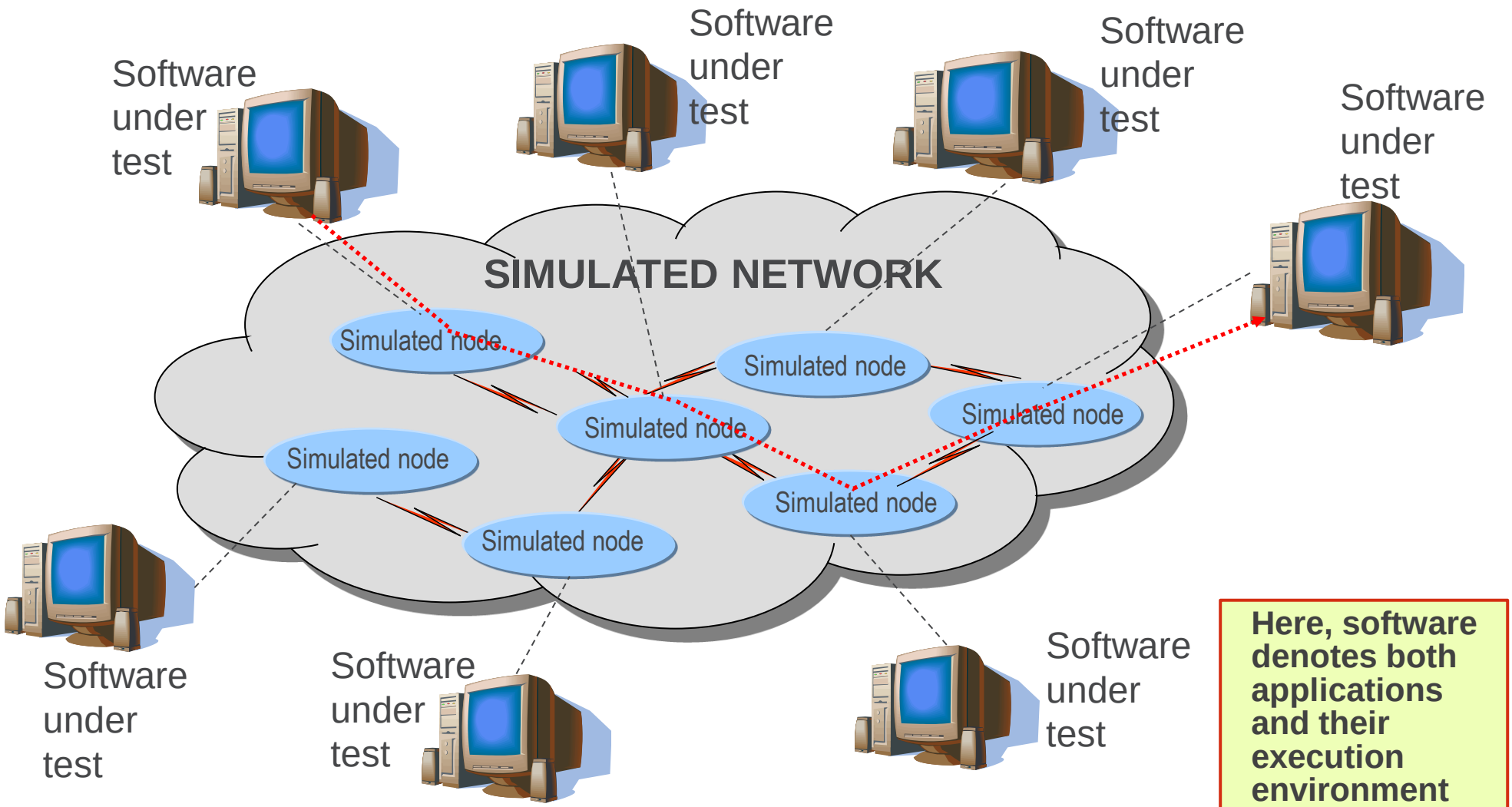
November 13, 2013

In a nutshell, what is VAN Testbed?

- VAN Testbed is
  - A versatile, multi-featured toolset for supporting network-oriented testing and evaluation
  - Especially suited for performing mobile wireless tactical network experiments
  - Composed of commodity hardware, network simulators, Xen hypervisor and VAN software
  - Designed for testing the correctness of and evaluating the performance of actual applications and protocol implementations

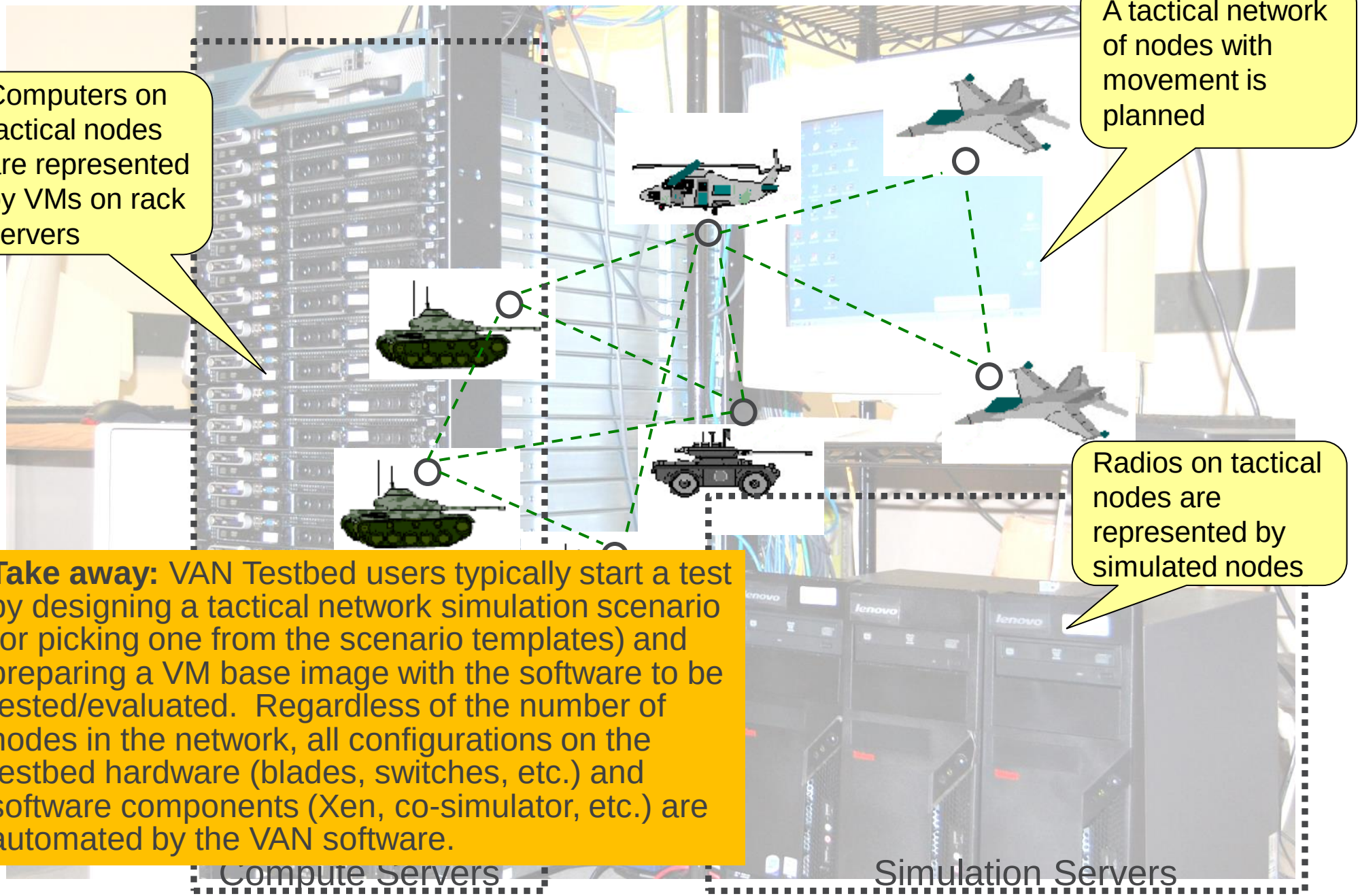
How does VAN Testbed work?

Software-in-the-loop simulation + seamless packet forwarding + host virtualization + automated VAN Testbed configuration, test execution and data collection



Here, software denotes both applications and their execution environment

How to plan tests on VAN Testbed?



Take away: VAN Testbed users typically start a test by designing a tactical network simulation scenario (or picking one from the scenario templates) and preparing a VM base image with the software to be tested/evaluated. Regardless of the number of nodes in the network, all configurations on the testbed hardware (blades, switches, etc.) and software components (Xen, co-simulator, etc.) are automated by the VAN software.

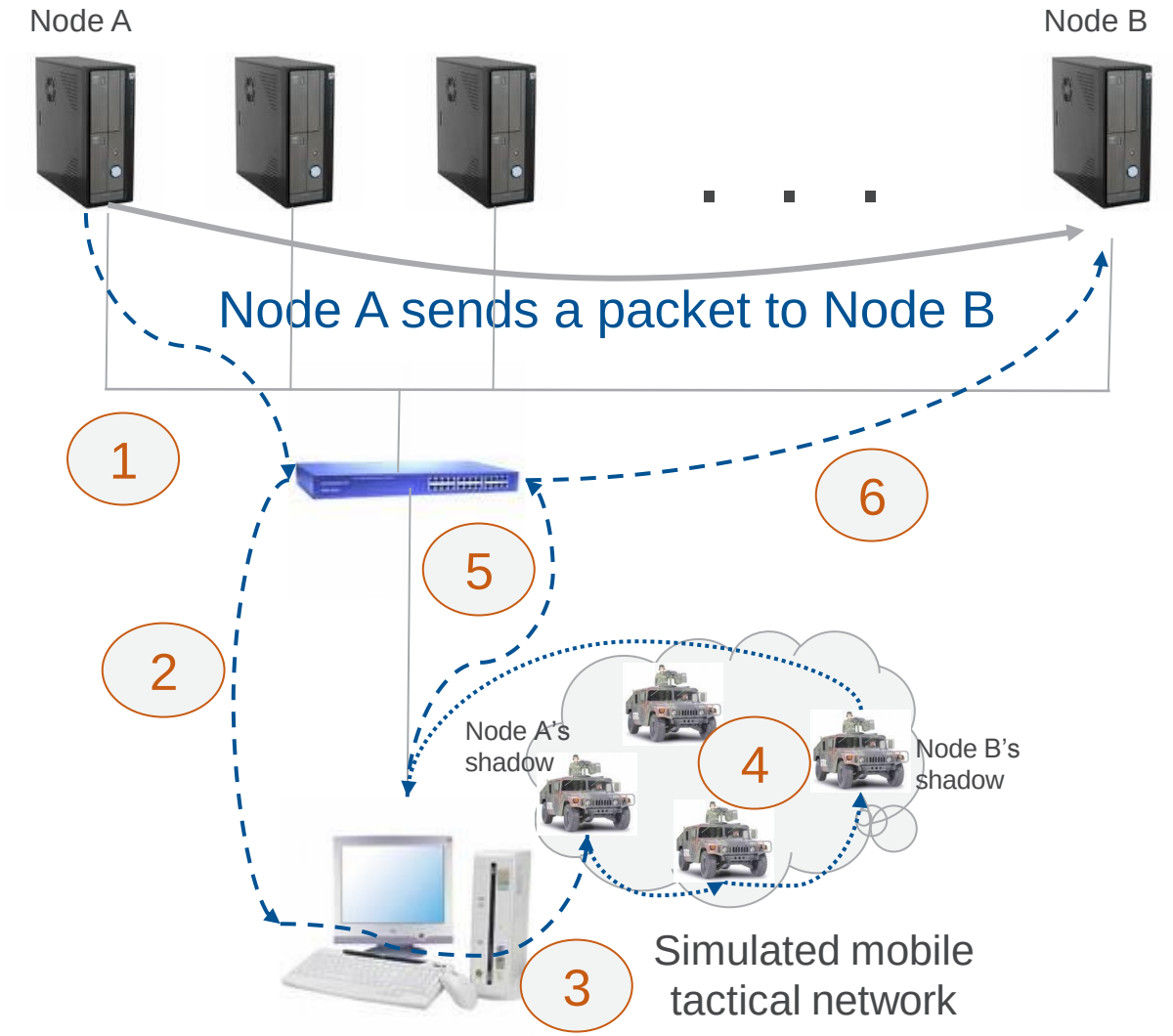
How do packets traverse through VAN?

The challenge:

- Most simulation-based hybrid network emulation solutions require manual modification to applications and/or custom-configure their underlying operating systems node by node

VAN Solution:

- VAN's seamless packet forwarding capability links nodes with their corresponding "shadow nodes" in the simulated network. It require neither code change nor manual system configuration



Take away: VAN Testbed is easy to use and supports testing/evaluation of any software on Windows/Linux without requiring access to source code, while other known solutions are much harder to set up and may not support testing/evaluating software binary.

Three major innovations

- Seamless packet forwarding
  - VAN enables setup of a virtual network on top of the testbed physical network without manual configuration of the network elements
  - Allow any software on its native OS to send packets over a simulated tactical network no different than over a real one
- Distributed time synchronization
  - By enforcing a nonlinear virtual (simulation) time across the testbed such that slower-than-real-time simulation will not affect test fidelity
  - Enable large-scale tests and evaluation, especially at high-fidelity
    - Already proven by a 274-node demo at APG to OSD and will be used by a CERDEC cyber program in the TRL-6 exit demo on a target network with over 500 nodes
- Testbed Operating Support Systems (TOSS)
  - Streamline and automate tedious, error-prone testbed operations by introducing a distributed testbed operating system with many management and user-support functions

Any difference between VAN and OPNET?

Feature	OPNET	VAN
Simulation model support	Implements a discrete event simulator, not compatible with other DES systems	Does not implement DES; works in conjunction with several COTS DESs like OPNET, QualNet, and ns-2
Support for hardware-in-the-loop	"System-in-the-Loop" (SITL) supports physically connecting hardware to an OPNET simulation. The physical part of the combined network needs to be configured for each scenario manually.	VAN allows the use of 'proxy' to seamlessly connect hardware to the VAN testbed. This layer of abstraction through proxy greatly simplify the configuration process. Not to mention that devices use specific interface to connect (e.g., USB or bluetooth) can be bridged to VAN via proxy as well
Scalability for hardware-in-the-loop	Only supports small-scale scenarios where simulation runs in real time	VAN's *TimeSync* feature provides *precise synchronization of time* across the hardware and the simulation, providing support for complex scenarios with *variable slowdown factors*, where some parts of the simulation may run slower than real time and other parts may run faster.
Support for software-in-the-loop	Supported via SITL or ESYS (co-simulator) API that requires *modifying the software under test* to use the API.	*VAN allows execution of unmodified software-under-test*, running in its native environment on virtual machines (VMs) which are automatically set up and configured

Some useful features

At a glance

- Support for broadcast, unicast and multicast
- Disk storage management solution
- Pause and resume of test scenarios on the fly
- Automated data collection and analysis support
- Hybrid emulated testbed setup
- 2D/3D GUI supporting on-the-fly and replay
- Regression test support
- Test life cycle management
- Software regression test support
- Compatible with multiple network simulators to take advantage of the best radio models available
- Flexible testbed resource sharing
- Efficient base image update support
- On-the-fly configuration and scenario adjustments
- Bots for performing interactive tests
- Virtual GPS feeds
- On-demand node movement and test scenario adjustment
- Testing with physical devices
- Modeling of red-black network separation
- Streamed testbed operations
- Support for large-scale (hundreds of nodes) tests and evaluation
- . . .

Cyber Testing Challenge:
Maintaining fidelity as scaling up tests

- Scale up tests while not sacrificing fidelity is a major challenge
  - Running actual software with real system configurations are critical to cyber testing fidelity
  - Resource availability and resource allocations are two key issues that need to be addressed
  - Every testbed has its limitation
    - When the testbed resources used to support emulation are lower than the resources in the network environment being emulated
- Possible solutions
  - By running full-fledged virtual machines
  - Through parallel simulation
  - Through time synchronization

Cyber Testing Challenge:
Root cause analysis for failure in the tests

- Data analysis support
  - Support online data collection and SQL query on the collected data
- Setting breakpoints, step execution, taking and saving snapshots
  - Techniques similar to those for debugging software
- Regression test
  - Rerun tests in exactly the same sequence of events as discrete event simulation can function in a deterministic manner
  - Automatically rerun tests after applying updates to the software and the environment

Cyber Testing Challenge:
Seamless use of cloud and virtualization technologies

- Virtualization enables flexible creation of virtual networks
  - It is important to know the abstractions being introduced, the deviation from the reality and its potential impact on test fidelity
    - E.g., a Linux VM can be configured as a router to handle packet forwarding; however, its throughput will be much lower than a commercial product and the Ethernet adaptor in use could be an order of magnitude slower
  - If communication effect is a concern, running time sync on the testbed is a possible solution
    - E.g., by slowing down the time perception on each VM by 10 times, the Ethernet adaptor used by the VM will appear to the VM 10 times faster
- To run cyber testing on the private cloud, it is critical to ensure proper isolation and to automate testbed operations
  - If resources cannot be shared, it's not a cloud!

Cyber Testing Challenge:
Testbed management for increased efficiency

- Testbed management is about test resource allocation and test automation
  - The concept of testbed management to testbed is similar to operating system to computer
    - Tests on testbed are like processes on computer
  - ACS has been developing a Testbed Operation Support System (TOSS) for the VAN Testbed
    - The key word is automation, super important for large-scale tests!
    - Major operations include new deployment, execution, pause & resume, save, withdrawl and redeployment
    - VM image management, disk space management and runtime disk access management are critical to performance
  - Many of TOSS' functionalities are common to many testbeds and can be easily customized to satisfy specific needs

Cyber Testing Challenge:
Automated attack launching and after-attack status examination

- Attack launching is an essential part in cyber testing
  - The process can be either manual or automated, depending on the types of attacks
  - Denial of service attacks are difficult to be simulated at high fidelity, as the testbed may not have the same amount of resources as the real network
- Pause and resume is useful for checking the testbed nodes' status after simulated cyber attacks
  - If each node on the testbed is represented as a VM, it is possible to perform VM memory introspection from the privileged domain
  - By checking the attack signatures, one can assess the effectiveness of the defense and the severity of the attacks at different time points after the attacks

Cyber Testing Challenge:
Post-mortem forensics analysis

- There are two possible targets for performing the post-mortem forensics analysis
  - Logs recorded throughout the tests
  - VM images at the end of the tests
- The content of the logs can include
  - IP/Mac packet logs
    - Useful for studying attack propagation
    - SQL queries can be used to speed up the investigation
  - Various system logs
  - Memory snapshots taken at the selected time points when pause and resume function was executed

Cyber Testing Challenge:
Big data analytics

- Assuming a cyber test includes 100 nodes and each node is allocated 4GB memory for execution
  - If 10 memory snapshots were taken during a test, it requires 10*100*4GB = 4TB hard drive to keep all the memory maps
    - This is already a performance stretch for some SQL engines and analytics software
    - If such tests need to be executed 100 times for different configuration settings, we would need to handle 400TB data
- Big data analytics engine and techniques are here to help
  - Tools in the Hadoop family and Map-Reduce paradigm are useful
    - Query time goes down from hours to minutes
    - The downside is configuration is complex, taking time to tune up the performance and the learning curve is steep
    - Machine learning algorithm support by Berkeley SPARK can expedite the process of data analysis
      - E.g., to identify (unknown) features of the VM images after cyber attacks

Summary

- VAN Testbed technologies are ready for transition
  - A number of programs have already benefited from using VAN Testbed to test their software
- VAN Testbed is versatile and highly customizable
  - It already supports a number of special testing requirements (e.g., used Nett Warrior Android handheld devices in the testbed and allowed JBC-P on the VMs to communicate with JBC-P on the Nett Warriors)