# CyberVAN: A Cyber Security Virtual Assured Network Testbed

Ritu Chadha, Thomas Bowen, Cho-Yu J. Chiang, Yitzchak M. Gottlieb, Alex Poylisher, Angello Sapello, Constantin Serban, Shridatt Sugrim, Gary Walther
Applied Communication Sciences
150 Mt. Airy Road, Basking Ridge, NJ 07920
{rchadha, tbowen, jchiang, ygottlieb, apoylisher, asapello, cserban, ssugrim, gwalther}@appcomsci.com

Lisa M. Marvel
E. Allison Newcomb
Network Science Division, CISD
U.S. Army Research Laboratory
Aberdeen Proving Ground, MD 21005
{lisa.m.marvel.civ, elizabeth.a.newcomb9.civ}@mail.mil

Jonathan Santos
Cyber Security Division, S&TCD
U.S. Army CERDEC
Aberdeen Proving Ground, MD 21005
jonathan.m.santos.civ@mail.mil

***Abstract*—In this paper we describe CyberVAN, a Cyber Security Virtual Assured Network testbed. CyberVAN enables speedy and flexible setup of high-fidelity cyber security scenarios to evaluate the effectiveness of both novel existing cyber technologies. CyberVAN provides many features needed by cyber security researchers, developers and practitioners alike, and can be used for both verification and validation purposes. We provide an overview of CyberVAN's functionality and a blueprint of the envisioned roadmap. Currently CyberVAN is available to ARL Cyber Security CRA (Collaborative Research Alliance) members. It is being used to evaluate CRA-developed cyber defense technologies and assess their applicability to the military strategic and tactical network environments.**

***Keywords—CyberVAN, testbed, cyber experimentation, security, evaluation.***

## I. Introduction

Cyber security is a critical concern nowadays. The Internet has become an indispensable part of business and of people's lives; however, it has also opened doors for criminals to commit cyber crimes and for adversaries to practice espionage, exfiltrate confidential data, and damage cyber physical systems. This can be primarily attributed to the lack of attention paid to security when computer systems were designed and built, and to the lack of theories, techniques, and foundations for developing secure computer systems.

To this end, the U.S. Army Research Laboratory (ARL) established a Cyber Security Collaborative Research Alliance (CRA) [1]. The alliance includes ARL, U.S. Army Communications Electronics Research, Development and Engineering Center (CERDEC), and academia and industry researchers, and its goal is to develop a science of cyber security in the context of Army networks. The overall objective of this CRA is to develop a fundamental understanding of cyber phenomena, including aspects of human attackers, cyber defenders, and end users, so that fundamental laws, theories, and theoretically grounded and empirically validated models can be applied to a broad range of Army domains, applications, and environments.

In this paper we describe CyberVAN, a Cyber security Virtual Assured Testbed designed and developed by Applied Communication Sciences (ACS) for validating the ARL Cyber Security CRA research. CyberVAN is built on top of the Virtual Ad hoc Network (VAN) testbed [2][3][4][5][6][7][8], which was also designed by ACS for testing the correctness and evaluating the performance of software applications and communication protocols in tactical military networks. CyberVAN incorporates key capabilities from VAN, including technologies and support for (i) speedy creation of high-fidelity strategic and tactical network scenarios by using a mix of physical machines/devices, virtual machines, physical networks and simulated networks; (ii) scenario deployment, control and management by using a GUI or issuing commands on a console; (iii) special features for supporting large-scale, high-fidelity experimentation; (iv) most commonly used networking protocols, including unicast and multicast protocols for IPv4, and IPv6; and (v) utilities that facilitate the experiment process. Given the above technology foundations, we have added capabilities to CyberVAN to support the specific needs of cyber security experimentation, namely, the creation of realistic cyber experimentation environments, including the setup of cyber attacks, cyber defense, and typical activities performed by human users. We provide an overview of the above capabilities in this paper.

Although CyberVAN was designed to support the ARL Cyber Security CRA basic research, its utility goes beyond evaluating proof of concepts. As CyberVAN provides a realistic cyber security evaluation facility in a closed laboratory environment, it can be used to evaluate many different types of cyber security techniques and technologies, such as those related to the detection of botnet command and control mechanisms and malware propagation approaches. CyberVAN provides many features useful for cyber security researchers, developers and practitioners alike, and can be used for both verification and validation purposes.

The rest of this paper is organized as follows. Section II provides background information about the core technologies that enable CyberVAN operations. Section III provides an overview of the salient features of CyberVAN. Section IV

The work reported in this document/presentation was partially performed in connection with contract number W911NF-14-D-0006 with the U.S. Army Research Laboratory. The views and conclusions contained in this document are those of the authors and should not be interpreted as presenting the official policies or position, either expressed or implied, of the U.S. Army Research Laboratory, or the U.S. Government unless so designated by other authorized documents. Citation of manufacturer's or trade names does not constitute an official endorsement or approval of the use thereof. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

describes the user interface of CyberVAN, and how users conduct experiments on CyberVAN. Section V describes use cases to illustrate some utilities of CyberVAN. Section VI discusses our envisioned roadmap. Section VII discusses related work, and Section VIII concludes the paper.

## II. Background

### A. Architecture

The CyberVAN testbed was designed to enable the setup of a testing and evaluation environment by leveraging network simulation and virtualization technologies. CyberVAN can be used to instantiate an instance of high-fidelity mobile wireless tactical networks, strategic networks, or a combination of both. CyberVAN consists of three seamlessly integrated enabling technologies: *software-in-the-loop network simulation*, *transparent packet forwarding*, and *host virtualization*. Software-in-the-loop network simulation allows real network traffic to be forwarded through a simulated network, transparent packet forwarding enables use of any physical machines, devices and virtual machines in an evaluation scenario without requiring manual IP configuration, and host virtualization allows the use of a wide range of virtualization technologies from light-weight containers to full virtual machines. CyberVAN also provides a novel time synchronization technology called TimeSync [7] that allows entities in large-scale experimentation to observe a common virtual time. Figure 1 shows the high-level architecture of CyberVAN.

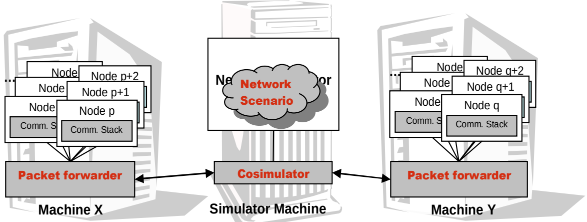
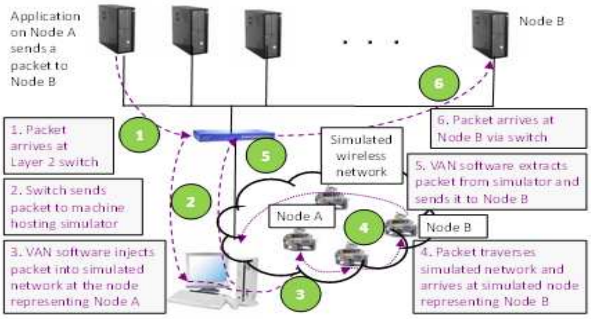


**Figure 1: High-Level Architecture of CyberVAN**

### B. Transparent IP-layer Packet Forwarding

As mentioned above, transparent redirection of IP-layer traffic through a simulated network is critical to CyberVAN. Application software and human participants involved in experiments should not be able to distinguish a virtual network from a real one from the communication viewpoint. The virtual network faithfully simulates packet forwarding activities. An example of this process is shown in Figure 2.



Application on Node A sends a packet to Node B
Node B
1. Packet arrives at Layer 2 switch
2. Switch sends packet to machine hosting simulator
3. VAN software injects packet into simulated network at the node representing Node A
4. Packet traverses simulated network and arrives at simulated node representing Node B
5. VAN software extracts packet from simulator and sends it to Node B
6. Packet arrives at Node B via switch
Simulated wireless network
Node A
Node B
1
2
3
4
5
6

**Figure 2: Transparent Packet Forwarding on CyberVAN**

## III. Key Features of CyberVAN

### A. Scenario Creation

CyberVAN can use simulation models developed for OPNET, QualNet, ns2 or ns3 to set up a simulated network. As a result, a large number of simulation models can be leveraged to quickly create CyberVAN evaluation scenarios. Typical evaluations on CyberVAN make use of existing scenarios or slightly adapted ones, while high-fidelity scenarios are created as needed. For example, to evaluate whether a specific cyber defense works for today's tactical networks, high-fidelity simulation models developed for the DoD can be used in the evaluation scenario to model a lower-echelon network at the highest fidelity possible in a laboratory environment.

Typically, to create a CyberVAN scenario, we (i) design the topology of the evaluation network and identify network elements and hosts; (ii) specify the operating systems running on the hosts and the required fidelity of their representations (using light-weight containers, virtual machines, or physical devices, etc.); (iii) specify software running on the hosts, such as DNS server, web server, database server, etc.; (iv) specify the locations and mechanisms of the cyber attacks and cyber defense software to be evaluated; (v) describe the execution flow of the experiment in terms of both deterministic actions and stochastic actions; and (vi) create a scenario definition file in XML using the data identified in (i) – (v). As CyberVAN already has a growing list of scenarios in its archive, the scenario needed for a particular experiment can typically be adapted from an existing one, and thus a scenario can be created in a short period of time.

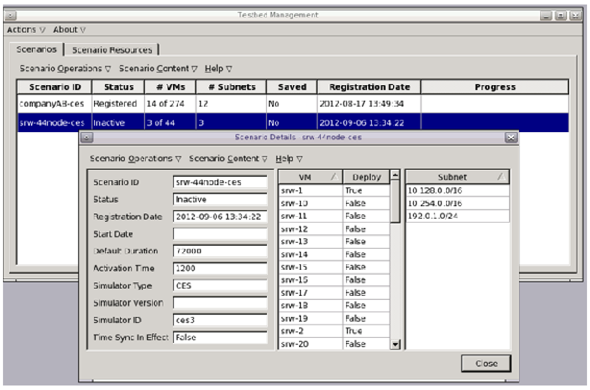
Although typical CyberVAN evaluation scenarios require only IP-layer transparent packet forwarding support between virtual machines and containers, CyberVAN also supports Link-layer transparent packet forwarding. This capability was developed to support the use of actual Link-layer devices if they are relevant to the evaluations. For example, one of the key ARL CRA research areas is Cyber Agility, and SDN-enabled moving target defense mechanisms are being studied [10]. As a result, we developed a link-layer transparent packet forwarding capability to enable insertion of SDN switches in the form of software (e.g., Open vSwitch) or in the form of hardware into evaluation scenarios.

To use physical machines or physical devices in a CyberVAN scenario, there are two options: the first option is to modify the IP configurations on the devices, if permissible; the other option is called "IP bridging", which introduces a CyberVAN proxy between the devices and the network simulator to transparently forward the packets between them. We have used the second approach to include Nett Warrior Android phones in the evaluation scenarios [6], as IP configurations could not be modified on those devices.

### B. Scenario Deployment, Control and Management

After creating a network simulation model and a scenario definition file for the considered evaluation scenario, experiments can start after the scenario has been deployed on CyberVAN. CyberVAN takes the scenario definition file as input and automates the scenario deployment process so that users do not have to be concerned about available resources, network connectivity, physical network provisioning and configurations, as well as network simulation. CyberVAN was designed to support concurrent experiments; multiple scenarios can be run simultaneously by different users without affecting others. Although CyberVAN allows expert users to run commands from a console to control the deployment of scenarios, novice users can use a scenario management GUI to control the deployment of scenarios, monitor, and manage the progress of their experiments.

The CyberVAN scenario deployment process identifies available resources needed for the scenario, allocates the resources to the scenario, boots VMs/containers on the assigned resources, and configures transparent packet forwarding network elements including hypervisors, layer-2 switches and simulation machines. Users can use the scenario management GUI to manage the life cycle of their experiments, such as deploy scenarios, start and stop scenarios, withdraw scenarios, and redeploy previously withdrawn scenarios. A snapshot of this GUI is shown in Figure 3.



| Scenario ID | Status | # VMs | # Subnets | Saved | Registration Date | Progress |
|---|---|---|---|---|---|---|
| companyAB-ces | Registered | 14 of 274 | 12 | No | 2012-08-17 13:49:34 | |
| srw-44node-ces | Inactive | 3 of 44 | 3 | No | 2012-09-06 13:34:22 | |

| | |
|---|---|
| Scenario ID | srw-44node-ces |
| Status | Inactive |
| Registration Date | 2012-09-06 13:34:22 |
| Start Date | |
| Default Duration | 72000 |
| Activation Time | 1200 |
| Simulator Type | CES |
| Simulator Version | |
| Simulator ID | ces3 |
| Time Sync In Effect | False |

| VM | Deploy |
|---|---|
| srw-1 | True |
| srw-10 | False |
| srw-11 | False |
| srw-12 | False |
| srw-13 | False |
| srw-14 | False |
| srw-15 | False |
| srw-16 | False |
| srw-17 | False |
| srw-18 | False |
| srw-19 | False |
| srw-2 | True |
| srw-20 | False |

| Subnet |
|---|
| 10.128.0.0/16 |
| 10.254.0.0/16 |
| 192.0.1.0/24 |

**Figure 3: A snapshot of the CyberVAN Scenario Management GUI**

### C. Large-Scale Experimentation

For experiments that do not require direct participation of human users, CyberVAN's time synchronization technology is a key utility for supporting high-fidelity experimentation and close examination of cyber behaviors during experiment runtime. For complex, large-scale simulations, discrete event simulation time advances slower than real time (typically in a non-uniform way), thus distorting packet propagation characteristics. For example, in a scenario where the simulation time advances constantly two times slower than real time, the packet propagation latency perceived by applications running on virtual machines will be twice as much as the expected value because of the slower simulation. To address the time divergence problem, we developed *TimeSync*, our time synchronization technology that uses the discrete event simulation time to control and synchronize time advance on virtual machines for large-scale experimentation [7]. The key idea in *TimeSync* is to make the time flow inside a VM closely track simulation time by dynamically controlling the Xen-VM time interface. Ideally, this should happen continuously, at infinitely small time granularity. In practice, however, extracting time information from the simulator and control of VM time can only be done at finite granularity. Our solution is to track the value of simulation time in small discrete steps (e.g., every 3ms of simulation time), along with an approximation of its average rate of change between consecutive steps. To achieve this, TimeSync uses two mechanisms: (i) simulator-side introspection, which extracts time information as the simulation is running, and (ii) dynamic time interrupt control in Xen, to apply this information dynamically to the VMs via the hypervisor-VM interface. The time information consists of the value of simulation time at a given instant and its projected rate of progress relative to real time. Readers who are interested in the details of the solution are referred to [7]. Using the TimeSync concept, one can model a much larger network at high fidelity than would be possible when all components run in real time. As an example, we worked with CERDEC in the past to conduct cyber experiments at the scale of 548 nodes running SRW, NCW, 802.16 (LAW) and Ethernet. The simulation of the 548-node network ran with a high-fidelity military radio waveform simulation model on QualNet was ~1.5 to ~1.8 times slower than real time due to the large network scale. We used TimeSync to slow down execution of the VMs representing the 548 nodes in the experiment in order to match their advancement of time with that of the QualNet simulator.

### D. Support for IPv4 and IPv6 protocols

CyberVAN supports the use of commonly used IPv4 unicast protocols such as TCP, UDP, ICMP, etc. CyberVAN also supports IPv4 multicast protocols such as SMF and IGMP. CyberVAN scenarios can also use IPv6 protocols and features, such as neighbor discovery. Many IPv4-based scenarios have been created for CyberVAN and a few IPv6 scenarios are currently available. In addition, for cyber security applications that need to modify network configurations on-the-fly, CyberVAN also supports experiments with such "network-aware" applications [8].

### E. Utilities for Experimentation Support

CyberVAN provides a number of utilities for experimentation support, including scenario visualization GUI, experimentation event framework, automated network data collection, SQL query interface, and user traffic generation. The scenario visualization GUI was built on top of the NASA WorldWind API [9]. It is especially useful to visualize cyber phenomena in scenarios with mobile tactical networks. The experimentation event framework is a technology enabler for the visualization GUI. The GUI subscribes to the real-time scenario information published to the event framework in order to receive and display status changes. The event framework is also useful for scripting experiments, such as launching certain applications on certain hosts at the beginning of an experiment, and launching cyber attacks after certain events have occurred. Since all packets in CyberVAN scenarios must transit through the simulated network, CyberVAN provides a facility to collect the entire set of generated network packets and store them in a database for offline analysis. Users may use their own preferred packet analysis tools to inspect the packets, or they can run SQL queries on the traffic flow records stored in a database. Finally, CyberVAN provides two types of traffic generation capabilities. The first is to program background traffic generation sources on the source nodes to send traffic to the sink nodes by specifying traffic generation parameters; the second is to use Skaion's "*ConsoleUser*" tool that mimics human users' interactions with everyday network-based applications, which generates realistic traffic accordingly. For example, *ConsoleUser* can be scripted to read and send email, browse web, download documents by ftp, etc. One can modify the "Markov Brains" used by ConsoleUser instances to mimic the behaviors of different human users. The resulting traffic patterns generated by ConsoleUser approximate those generated by real human users.

### F. *Realistic Cyber experimentation*

A realistic experimentation environment is a critical requirement for a cyber testbed. This includes incorporating realistic network topologies, services (e.g., web and DNS services) and traffic (or workload), so that researchers can measure how experimental defensive mechanisms will behave in target environments. Realism is also key for *predictability*, because any lack of realism in the experimentation process will lead to results that do not reflect those predicted by researchers.

ACS has built a collection of sophisticated malware samples that cover broad classes of malware (e.g., worms, trojans, bots, RAT/APT, crimeware toolkits, etc), and span the spectrum of sophistication from relatively straightforward, older samples such as the Zeus Russian crimeware toolkit to more recent samples thought to be equivalent with the capabilities of nation-state threats.

We have developed utilities for simulating different types of cyber attacks, including malicious traffic generation, which includes both background malicious activity and foreground attacks. The background malicious activity is based on driving tools such as Metasploit and Nmap according to some observation models, and the foreground attacks can use malware collections as well as attack models with engineered vulnerabilities inserted into target applications. The purpose of this malicious traffic generation is to create a stream of background malicious traffic, composed of scans and attacks that mimic the "background noise" seen on any Internet-connected system, ensuring that the foreground scenario of interest is not the only malicious traffic seen by a system under test. This is an important aspect of realistic experimentation, since it is a key step in determining how a system under test handles false positives, or whether it can distinguish between successful and failed attacks. The tools implement apparently undirected, failed scans and attacks in a configurable manner, where the numbers of targeted hosts and ports per scan, as well as the ports targeted during a scan, can be informed by analysis of network data captured from real user environments.

## IV. CYBERVAN WEB PORTAL

### A. *CyberVAN Web Portal*

CyberVAN provides testbed access to authorized users via a Web portal. The Web portal provides various types of information, including user documentation, tutorials, and case studies. The Web portal also integrates a web-based remote desktop utility, thus allowing users to access resources on CyberVAN through their choice of browser (e.g., IE, FireFox, and Chrome) without having to install any remote desktop and learn the configuration settings of the remote desktop software (e.g., port number to use). We describe this capability below.

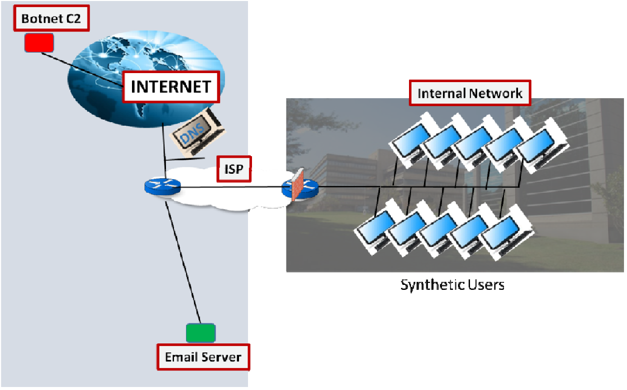
### B. *Scenario Access through Guacamole*

Guacamole [15] is a browser-based utility that allows users to access remote machines by running a remote desktop application using a Web interface. By using Guacamole, CyberVAN simplifies the procedure that users need to follow in order to run their experiments. By default, CyberVAN offers users a Linux Fluxbox desktop through Guacamole. Users can launch terminal windows, manage their scenarios by using the testbed management GUI, or run their applications. CyberVAN also provides facilities for users to upload the files (e.g., data analysis scripts or experiment essential data) from their own machines to CyberVAN, and also download files (e.g., various experiment logs on machines) from CyberVAN to their own machines for offline analysis.

## V. CASE STUDIES

### A. *Emulated Cyber Attack Data Trace Collection*

One of the key challenges facing cyber security researchers is the lack of cyber attack data for analysis, for verifying whether developed new techniques will work against attacks, and for generating reproducible results. As an initial effort to support the cyber security research community, a number of attack scenarios have been created on CyberVAN, experiments have been conducted for each of these scenarios, and substantial data have been collected. The details of this effort are described in [11]. Below we provide a brief description of one use case.



**Network Layout**

**Figure 4: Scenario used for data collection**

Figure 4 shows the notional topology of a scenario used for dataset collection. The internal network consists of workstations, server nodes, and a firewall node. The workstations have outgoing Internet access and are used by synthetic users to perform email, ftp, and web browsing tasks that access both internal and external servers. The internal server nodes provide internal web sites and internal name service. The internal server nodes have outgoing Internet access and limited incoming Internet access: the internal web servers allow external access for web services, and the internal name server has bi-directional connection to the public name server. A firewall node is located between the internal and public networks. The public network consists of a set of attacker nodes, and a synthetic Internet provided by our partner

Skaion, which is a single node that simulates thousands of public Internet nodes, including web sites, ftp sites, name servers and email servers. Note that all the elements are in a closed scenario environment on CyberVAN.

All of the datasets consists of an attack-free period of at least one hour, followed by a period during which an attacker, using an attack node in the public Internet, launches an attack against the internal nodes for the purpose of establishing a botnet that infects the majority of the internal nodes. The botnet used is synthetic; however, it contains typical real world botnet features such as using stealth and migration to hide the location of the bot master and rendezvous points, using a process-hiding rootkit for persistence on victims, and supporting stealthy communication between bot master and victims for executing commands and reporting status.

The datasets were collected in the context of 4 scenarios, and the scenarios differ in the methods used by the attacker to obtain initial entry into the internal network and the characteristics of the botnet once it is implanted. The entry methods all require social engineering using methods that range from tricking a naïve user into executing an attachment in a spoofed email purporting to be from a colleague, to tricking a user into clicking a link on a frequently used internal web site that the attacker has corrupted. The varied botnet characteristics include how the bots discover and communicate with the bot master. Initial attack entry and subsequent propagation within the internal network exploit both real-world and synthetic vulnerabilities. The attacker is presumed to have some prior knowledge of the internal network, such as the email addresses of internal users, the IP address ranges of the internal network, and a rough idea of the types of internal web services offered. The attacker expands this knowledge using nmap and sqlmap utilities.

The datasets that we collected contain data that are gathered from the internal nodes and contain the following information:

- Network packets, as captured by tcpdump,
- File writes, as captured by sysdig,
- Pre and post tarballs of node state, as shown in the /proc file system, and
- Pre and post tarballs of logfiles, from /var/log.

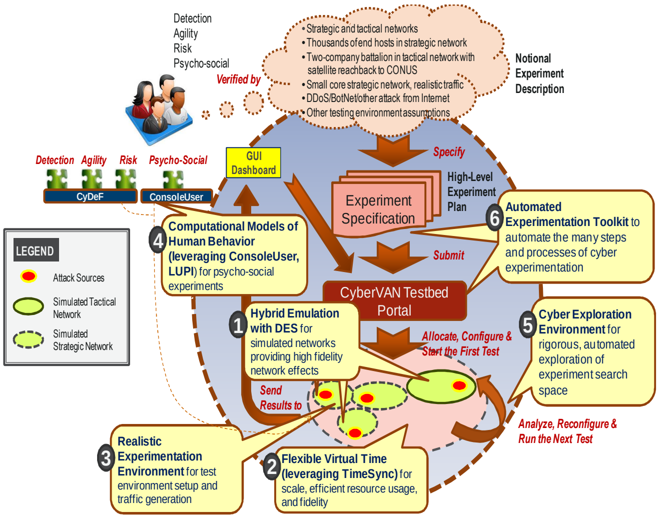
Note that this set of observations is a starting point in this ongoing effort. The value of this use case is that attack scenarios are repeatable so that if researchers need to collect additional data, collectable through either standard utilities or newly developed tools, the scenario can be modified to collect the data and the scenarios can be run again to produce new datasets.

### B. SDN-enabled Cyber Deception

CyberVAN was also used in the development of ACyDS, an adaptive cyber deception system [10]. ACyDS provides a unique, *deceptive* virtual network view to each host in an enterprise network. That is, a host's view of its network, including subnet topology and IP address assignments of reachable hosts and servers, does not reflect actual network configurations and is different from the view of any other host in the network. For example, each host will access the network's DNS using a different IP address—the address assigned to the DNS in the network view that that host observes. ACyDS generates views with the desired properties dynamically, changing every host's network view on-the-fly. ACyDS enforces dynamic network view changes to invalidate cyber intelligence collected by an attacker from prior reconnaissance activities, as subnet topology and IP address assignments are changed in every view update. In a nutshell, ACyDS's deception approach (i) makes it very difficult to perform reconnaissance on a network, (ii) prevents collusion across multiple compromised hosts, and (iii) increases the likelihood and confidence of detecting the presence of intruders. ACyDS leverages SDN technologies, including OpenFlow switches and controllers to enable seamless reconfiguration of the above network views. Our implementation uses Open vSwitch and POX to seamlessly handle the typical types of IP network traffic and management applications—e.g., ARP, UDP, TCP, ping and traceroute—by dynamically modifying IP header fields at the SDN switch per installed flow rules that implement the network views. A prototype of ACyDS has been developed and evaluated using CyberVAN. The concept was successfully verified to be a viable approach through testing UDP, TCP and ICMP traffic flows between nodes with completely different network views. The necessary network address translations are handled by the SDN Controller, SDN Switch, and Deception Server; the latter was introduced to handle specific deception actions that are too complex or considered not suitable for the SDN switch to carry out.

## VI. Development Roadmap

ACS was selected by ARL in 2014 as the Applied Research and Experimentation Partner (AREP) for the Cyber Security CRA. Since then, ACS has been developing technologies to enhance the capabilities of the CyberVAN testbed and support experimentation efforts. The current and envisioned capabilities of CyberVAN are shown in Figure 5. Below we enumerate the key functions, and their benefits, that we plan to design or enhance in CyberVAN. They are numbered based on the labels shown in Figure 5.



Detection
Agility
Risk
Psycho-social
Verified by
• Strategic and tactical networks
• Thousands of end hosts in strategic network
• Two-company battalion in tactical network with satellite reachback to CONUS
• Small core strategic network, realistic traffic
• DDoS/BotNet/other attack from Internet
• Other testing environment assumptions
Notional Experiment Description
Detection
Agility
Risk
Psycho-Social
CyDeF
ConsoleUser
GUI Dashboard
Specify
Experiment Specification
High-Level Experiment Plan
Submit
CyberVAN Testbed Portal
Allocate, Configure & Start the First Test
Send Results to
Analyze, Reconfigure & Run the Next Test
LEGEND
Attack Sources
Simulated Tactical Network
Simulated Strategic Network
1 Hybrid Emulation with DES for simulated networks providing high fidelity network effects
2 Flexible Virtual Time (leveraging TimeSync) for scale, efficient resource usage, and fidelity
3 Realistic Experimentation Environment for test environment setup and traffic generation
4 Computational Models of Human Behavior (leveraging ConsoleUser, LUPI) for psycho-social experiments
5 Cyber Exploration Environment for rigorous, automated exploration of experiment search space
6 Automated Experimentation Toolkit to automate the many steps and processes of cyber experimentation

**Figure 5: Current and Envisioned Capabilities of CyberVAN**

(1) We plan to introduce parallel simulation to reduce the slowdown factor for running large-scale cyber security experiments (e.g., DDoS experiments). Using multiple simulators can significantly reduce the total amount of time needed to complete one round of a large-scale experiment.

(2) Our TimeSync technology was developed for Linux VMs on Xen. We plan to port this technology to a generic hardware virtualization container such as QEMU [20] so that a wide range of OS virtual machines can be supported. Furthermore, this enhancement will also provide CyberVAN the flexibility to use virtualization platforms other than Xen, such as Linux Kernel-based Virtual Machine (KVM) [21] and VMWare [22].

(3) We are enhancing the capabilities of realistic experimentation and expanding our portfolio of cyber attacks to enable a wider range of cyber experiments. Thus, CyberVAN can support a larger variety of cyber experiments.

(4) We plan to incorporate the ARL CRA research on human factors into CyberVAN, including models for users, defenders and attackers. For example, currently attacker behaviors are always pre-scripted. This is different than a real-life cyber attack scenario where the attacker will first analyze the situation and then plan the next move. To this end, we have performed preliminary investigations on developing a "cognitive attacker" that will respond to different situations based on the results of analysis. This capability will allow a much more thorough evaluation of cyber defense technologies, as the emulated attacker would behave like a human being.

(5) We plan to develop a cyber exploration environment that supports rigorous and automated exploration of cyber experimentation search space. Since a cyber experiment typically includes multiple parameters each with a wide value range, an automated exploration mechanism using different combinations of parameter values to intelligently explore the search space can greatly accelerate the validation of the technologies under test, establish confidence in the tested technologies, and identify weakness in the technologies.

(6) We plan to enhance CyberVAN's automated cyber experimentation toolkit to further support the experimentation process and thus increase resource usage and research productivity.

## VII. Related Work

Most of the testbeds developed to support cyber security experimentation have very strict access policies. For example, the National Cyber Range that was initially launched by DARPA is not openly accessible and comprises many classified elements. Aside from CyberVAN, the most well-known testbed dedicated to cyber security experimentation is the DeterLab [16] hosted and maintained by USC ISI. The DeterLab was developed on top of NSF GENI's [19] Emulab [18], which provides facilities to set up an emulated network topology using physical network elements and hosts. As a result, the DeterLab's emulation can only support fixed topology networks such as enterprise networks and strategic networks, and does not support mobile and wireless networks with any level of fidelity. PlanetLab [17] is another NSF-funded effort, which provides a generic testing and experimentation facility. Many of the key concepts discussed in this paper has similar counterparts in PlanetLab. However, PlanetLab's focus is to allow the construction of a laboratory environment leveraging resources scattered all over the world and therefore does not provide any control over the network link latency. Also, PlanetLab was designed to support Internet-based experimentation, and does not provide mobile wireless network experimentation support.

## VIII. Conclusions

This paper describes CyberVAN, a testbed designed for supporting cyber security experimentation. We described the existing functionality of CyberVAN, provided descriptions of test cases to illustrate some of its utilities, and also discussed the planned roadmap for CyberVAN.

## References

[1] http://www.arl.army.mil/www/default.cfm?page=1417, retrieved on 4/19/16.

[2] A. Poylisher, C. Serban, J. Lee, T. Lu, C. J. Chiang, "Virtual Ad hoc Network Testbeds for high fidelity testing of tactical network applications", IEEE MILCOM, 2009.

[3] A. Poylisher, T. Lu, C. Serban, J. Lee, R. Chadha, C Jason Chiang, "Realistic modeling of tactical networks with multi-level security in VAN testbeds", Proceedings of IEEE MILCOM 2010.

[4] Alex Poylisher, Constantin Serban, John Lee, Ted Lu, Ritu Chadha, Cho-Yu J. Chiang, "A Virtual Ad Hoc Network Testbed", International Journal on Communication Networks and Distributed Systems, 2009.

[5] Alexander Poylisher, Florin Sultan, Yitzchak M. Gottlieb, Constantin Serban, John Lee, Ritu Chadha, Cho-Yu Jason Chiang, Keith Whittaker, James Nguyen, and Chris Scilla, "TimeSync: Virtual Time for Scalable, High-FidelityHybrid Network Emulation", MILCOM 2012.

[6] C. Serban, A. Poylisher, A. Sapello, Y. Gottlieb, C. J. Chiang, R. Chadha, "Testing android devices for tactical networks: A hybrid emulation testbed approach", IEEE MILCOM 2015.

[7] F. Sultan, A. Poylisher, C. Serban, C. J. Chiang, R. Chadha, "TimeSync: Enabling Scalable, High-Fidelity Hybrid Network Emulation", the 15th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWIM), Cyprus, 2012

[8] Constantin Serban, Alex Poylisher and Cho-Yu J. Chiang, "A Virtual Ad hoc Network Testbed for Network-Aware Applications", Proceedings of 12th IEEE/IFIP Network Operations and Management Symposium (NOMS 2010).

[9] http://worldwind.arc.nasa.gov/, retrieved on 4/19/16.

[10] ACyDS, an adaptive cyber deception system. Submitted to MILCOM 2016 for publication consideration.

[11] Four Labeled Datasets to Enable Reproducable Cyber Research. Submitted to MILCOM 2016 for publication consideration.

[12] OpenFlow. http://https://www.opennetworking.org/sdn-resources/openflow, retrieved on 4/9/16.

[13] POX. https://github.com/noxrepo/pox, retrieved on 4/19/16.

[14] Open vSwitch. http://openvswitch.org/, retrieved on 4/19/16.

[15] Guacamole. http://guac-dev.org/, retrieved on 4/19/16.

[16] DeterLab. https://www.deterlab.net/, retrieved on 4/19/16.

[17] PlanetLab. https://www.planet-lab.org/, retrieved on 4/19/16.

[18] Emulab, http://www.emulab.net/, retrieved on 4/19/16.

[19] NSF GENI, http://www.geni.net/, retrieved on 4/19/16.

[20] QEMU, http://qemu.org/, retrieved on 4/19/16.

[21] Linux Kernel-based Virtual Machine, http://www.linux-kvm.org/.

[22] VMWare, http://www.vmware.com, retrieved on 4/19/16.