A Plan for Malware Containment in the DETER Testbed

Ron Ostrenga and Stephen Schwab
SPARTA, Inc.
Robert Braden
USC Information Sciences Institute

Abstract

The DETER testbed provides a shared Internet-accessible environment where security researchers can safely run experiments and companies can test their security products. Experimentation with malware in DETER has so far been limited to simulated worms, which only simulate the spreading action without actually infecting any computer systems. This paper outlines a set of architectural and procedural changes that should allow safe experimentation with a class of moderately risky, real malware in the DETER testbed.

1 Introduction

The DETER testbed laboratory [1,9] which runs a tailored version of Utah's Emulab software [2], provides a shared Internet-accessible environment where security researchers can run experiments and companies can test their security products, without threatening the Internet. DETER was initially funded in 2003 by the National Science Foundation (NSF) and the U.S. Department of Homeland Security Advanced Research Projects Agency (HSARPA); it is currently funded by HSARPA. This paper assumes some general knowledge of the design of Emulab.

As a security testbed, DETER is designed to provide *containment* and *isolation* for all experiments. For example, there is no direct IP path from a DETER experimental node to the Internet. A wide range of cyber security-related experiments have been performed on DETER, including DDoS attacks, worm propagation, and BGP attacks.

However, DETER malware experimentation to date has been primarily focused on simulated worms that did not actually exploit operating system vulnerabilities; instead, they simulated the spreading action without actually infecting systems. Real malware experiments have required the testbed to be disconnected from the Internet, and testbed operators to be extensively involved in executing an experiment. This paper outlines a set of extensions to the DETER testbed architecture, policies, and procedures to enforce strengthened containment to enable controlled experimentation with real malware that is moderately risky, and furthermore, to enable experimenters from remote sites to interact with and run these experiments without excessive support from the local testbed operations staff. We use the term *malware containment* for the stronger containment discussed in this paper.

The Anti-Virus (AV) research community, including individual security vulnerability researchers as well as billion dollar corporations, has built up a modus operandi over many years of tracking malware. The results are a set of practices that enable sample sharing, safe handling, participant vetting, and control of information dissemination within the community [3]. One of the overall goals in creating a malware containment capability for DETER is to support collaboration and interactions between DETER users, academic researchers, and the existing AV research community. To facilitate these on-going interactions, we intentionally adopt a malware containment strategy that aims to be substantially compatible with the practices of the AV research community, as described below.

1.1 "Moderately Risky" Malware

Specifically, we wish to provide safe containment for real malware that infects widely deployed general purpose operating systems, such as Windows, Linux and FreeBSD, and that is currently detectable by anti-virus (AV) software or by intrusion detection systems (IDS). Our goals include supporting experiments with worms, viruses, botnets, email mass-mailers, adware, and spyware that chronically infect systems attached to the Internet.

We can classify malware as follows, loosely in order of increasing risk. We acknowledge that these classes are tentative, not necessarily objectively defined, and that experts may disagree as to the class and relative risk of specific malware samples:

- Class 1: Malware that is known not to mutate from generation to generation and that is recognized by AV scanners or IDSs with current signature files.
- Class 2: Malware that is polymorphic, i.e., may mutate into several distinct forms, but each presents a known signature from a decrypted body.
- Class 3A: Malware that is metamorphic and does not present known signatures in all its forms.
- Class 3B: Malware that tampers with the "hardware", i.e., infects BIOS or nonvolatile storage (RAM/EEPROMs/Flash) other than the primary file system stored on hard drives.

- Class 4: Unknown malware, including newly discovered wild samples.

The selection of numerical classes 1-4 is meant to loosely correspond to bio-safety levels [4] for infectious agents. Innocuous agents belong to class 1; lethal viruses such as Marburg and Ebola belong to class 4.

We are proposing to fully support only classes 1 and 2, i.e., our approach will depend upon the ability to recognize the malware using commercial AV scanners with current signature files. With some limitations and additional restrictions, the same techniques may be applicable to some members of class 3, on a case-by-case basis. We expect that class 4 will continue to require the complete system isolation ("clean room") techniques that have been used for all classes in the past.

1.2 Requirements

Current best practices within the AV research community for handling malware involve isolating the machines and networks where malware will be loaded and studied. Only trained and trusted individuals are permitted to work with malware in these controlled environments. Most organizations accomplish this by physically locating the machines in a secured room without external network connectivity, and imposing a restriction that no media entering the room be allowed to leave. Certain exceptions are made for sharing or exchanging malware samples between collaborating organizations, with the malware traditionally being transported via media such as floppy disks, not over the Internet.

The primary requirement for DETER's malware containment mechanism is to provide a similar level of assurance by isolating the environment in which malware can run, while permitting *remote* experimenter's to control or interact with the environment *over the Internet*. Like the AV research community, we assume evil intent in the malware but not in the experimenters running the malware (although human errors must be accommodated); any security can be thwarted by compromised personnel.

This top-level requirement is driven partly by technical need, but mainly to promote a strategy of long-term interoperability and sample sharing with the AV research community. A key tenet of our plan will be to design to fit with current best practices, adopting the same policies, procedures, and technical mechanisms where feasible. For example, we will initially require that malware be imported to DETER on shippable media -- optical disk (CD-R/DVD-R) or an external hard drive -- not over the Internet. While a design employing FTP of encrypted archives might provide a similar degree of protection, the AV research community has generally adopted a best practice of not transmitting malware, in any form, over the Internet, to prevent accidental release. Clearly, this community believes that using physical media is less likely to result in accidental release when compared to the risks of sending data archives, even encrypted ones, over the Internet. (More recently, some members of the AV research community have shifted to using password-protected archive formats to contain malware samples while being transferred over the Internet. Once we have gained experience with safe handling and transfer of malware, DETER may also introduce similar password-protected archives modeled on these best practices.)

On the other hand, safe containment of malware experiments in DETER requires a set of policies and operational procedures as well as specific technical mechanisms to ensure containment. We wish to break with the AV research community practice of complete isolation, because we must meet a DETER objective of allowing researchers to access the testbed without physically traveling to the testbed cluster sites. Our plan controls how and where malware may be introduced within the testbed, and recognizes the following requirements.

- **Malware Transport:** As described earlier, any malware must be transported to or from the testbed on physical media, clearly marked. Ideally, documentation of the malware contained on the optical disk or external hard drive will accompany it. It may also be encrypted.
- **Isolation:** Live malware must only be introduced into experiments that are isolated, by means described below, from the DETER control plane and the rest of the Internet.
- **Sterilization:** All live malware must be removed from DETER experimental nodes before isolation is broken and the nodes are made available for use by other experiments.
- **Cleaning:** Experimental results, including logfiles or other data from a malware experiment, must be AV scanned and cleaned before they are released from the DETER containment facility to a general DETER user account. This is the principle reason that our plan supports only classes 1 and 2, i.e., malware with recognized AV signatures. In the absence of a known signature, it is very difficult to automate the release of experiment results while assuring containment.
- **Control:** Media containing malware must be secured when not in use. The DETER clusters are physically secured, and malware loaded on and controlled by DETER meets this standard of physical security.
- **Defense in Depth:** We should avoid depending upon any single mechanism for containment.

Additionally, we note that all DETER experiment applications are reviewed by an executive committee, and security risks posed by each proposed experiment are assessed prior to granting approval. Malware experiments are closely scrutinized, and operations staff will be tasked to ensure that new malware experimenters are aware of safe malware handling procedures and to ensure compliance with DETER malware policies and procedures.

In our experience with the DETER testbed, the only occasions of loss of isolation/containment were caused by experimental modifications to the testbed control plane software, e.g., adding new features. During the time that one or more malware experiments is running, the operational staff must ensure stability of the testbed control software.

2 High Level Design

2.1 Overview

The primary features of an approach to malware containment, which will meet the above requirements, are as follows:

1. An enhanced isolation mechanism, using firewalls and VLANS, that will prevent any unauthorized data, including possible exploits or self-propagating code, from leaking out to affect other experiments, the DETER control plane, or the Internet.
2. Mechanisms and procedures for safely introducing malware into a contained malware experiment.
3. Mechanisms to allow the experimenter to remotely control and monitor the experimental nodes. Of a contained malware experiment. Specifically, the experiment can access a contained malware experiment only via a carefully-limited Virtual Network Console (VNC) [5] session. This is discussed more fully below.
4. Procedures to safely AV scan, clean, and extract results such as logs and traces from the experiment. This may additionally include quarantining malware samples, using encrypted archives for example, to provide persistent storage between runs of the same malware experiment.
5. A mechanism to zero-wipe all components involved in the experiment upon termination. An exception will be made for components that are dedicated exclusively to supporting malware experimentation, which will alternately be isolated when not in use. (By removing power, or physically disconnecting network cables, for example.)

When the malware is stored at the testbed site and introduced into an experiment or when data is extracted from the experiment, it is necessary to provide some technical means of rendering the malware harmless. We consider two possible mechanisms for accomplishing this: (1) encrypting the files that may contain the malware, or (2) storing the malware on an external hard disk that can be powered down independently of the host system to which it is attached. We need more experience to understand the tradeoffs between these two mechanisms, and we currently plan to use one or both in each contained experiment, while evaluating their long-term costs and risks.

We plan to integrate the malware containment mechanisms into Emulab's control software, taking advantage of Emulab's recursive self-hosting feature known as "Emulab-in-Emulab" ("EinE") [6], in conjunction with augmenting the Emulab per-experiment firewall configured and deployed automatically at the boundary between the DETER control plane and the experiment itself. This approach will provide a layer of defense to protect the "outer" control plane of the DETER testbed, while imposing modest limits on the experimenter.

2.2 Phases of a Malware Experiment

A malware experiment will be conducted in phases, described below and illustrated in Figure 1. The phases are controlled by state kept outside the contained experiment[1].

The malware containment control state prevents violation of the containment policies. However, manually following the sequence of steps shown in the figure and described in detail below will be complex, especially for novice users. To aid users, reduce errors, and reduce support calls to testbed operations, we plan to incorporate malware containment controller agents into DETER's Security Experimenter's Environment (SEER) [7]. SEER provides an experiment control node within an experimental. This node is controlled by the user through the SEER GUI.

In a non-malware experiment, the SEER GUI (a Java executable) can execute on the user's desktop, and contact nodes in the experiment via a proxy running on users.deterlab.net (the outer users host on the left side of figure 1.) In a malware experiment, the SEER GUI is executed on a node within the experiment, preferably the control node, and may display its frame on a virtual VNC server display. The DETER experiment's desktop will connect through a path including VNC port forwarding on users.deterlab.net and a VNC application proxy on the per-experiment firewall. All the flexibility and automation of SEER will be available to malware experiments because the project and per-experimenter file systems are cloned from the outer users to the inner users machine when the experiment is created, along with a cloned set of databases

[1] In the "outer" DETER control plane

and other state from the outer boss to the inner boss machine.

We expect to extend SEER with node agents and convenient PERL scripts for invoking the malware containment machinery in accordance with the rules described here. However, steps that require the user to interact with the outer DETER control plane will not be available from the SEER GUI *after* the containment firewall is activated. Instead, an experimenter would use a separate web browser connected to the outer boss to control those aspects of the experiment. The malware containment lifecycle includes six phases (see figure 2) as described in detail in the remainder of this subsection.

2.2.1 Preliminary Configuration

Phase S0: To run an experiment with real malware, the experimenter's ns2 script must specify a firewalled EinE configuration to the DETER control plane. It must also designate a Malware Archive (M/A) node. The DETER control plane is configured to make the M/A node a special class of DETER node that cannot be allocated to an experiment unless it is specified as a part of a firewalled EinE type experiment. Testbed operations staff will load the physical media containing the (possibly encrypted) malware into this node in advance of the first run of the malware containment experiment. Subsequent runs of the experiment may use the malware previously loaded, without requiring manual intervention.

An experimenter creates a firewalled Emulab-in-Emulab ("firewalled EinE") experiment within their project, swaps it in, configures it, and tests it without malware. Using Emulab software, the topology may be specified directly by an ns2 script or may be created using the GUI tool located on the create-experiment page. During this phase, the experiment does not need malware containment.

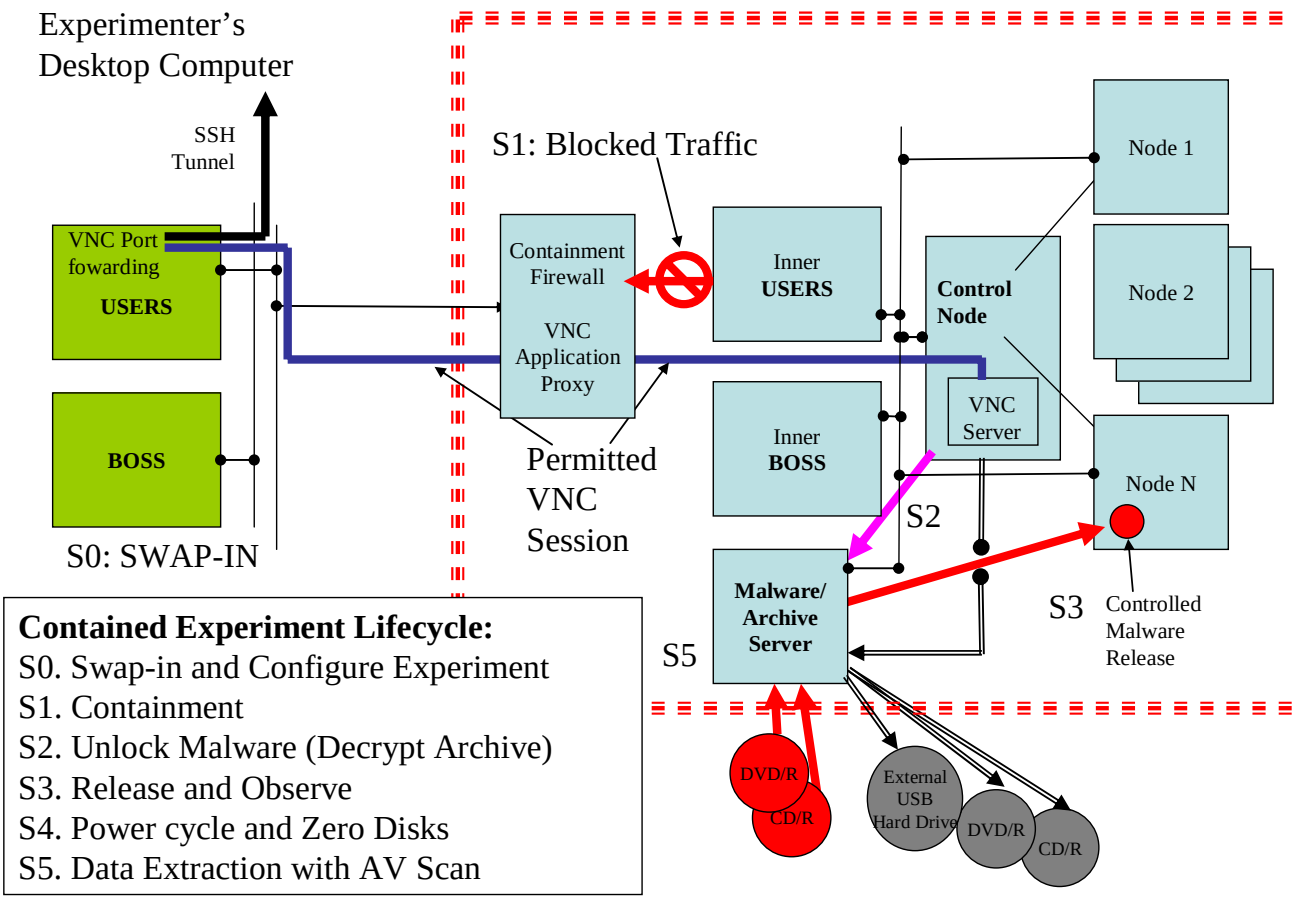

Figure 1. The DETER Architecture for Malware Containment.

2.2.2 Containment Establishment Phase

In Phase S1, the user requests that malware containment be established for the experiment. It is created by allocating a special malware archive node that causes the Emulab swapin process to configure and instantiate a per-experiment firewall ("Containment Firewall" in Figure 1) to severely restrict traffic into and out of the experiment. The only IP traffic permitted across the firewall is a carefully controlled VNC (Virtual Network Console) [5] session, which is used by the experimenter to interact with the contained experiment. Of course, the malware being tested must be known not to be able to infect the firewall.

2.2.3 Malware Introduction Phase

Phase S2: After the firewall is in place to provide malware containment, the DETER control plane powers up the external disk drive on the Malware Archive (M/A) node, containing the malware, or signals the M/A node to unlock (decrypt) the malware for the experiment. Note that only DETER's (outer) Boss server has direct access to the power controller for the M/A node and its external hard drive, and hence determines when the malware is provided within the experiment.

2.2.4 Experiment Execution Phase

Phase S3: The user controls and monitors the experiment over the VNC session. Within the containment boundary (shown by the dashed red line in Figure 1), the "inner Users" and "inner Boss" nodes implement the normal Emulab control plane functions for the set of nodes allocated to the experiment, and isolated from all other experiments. For example, the Firewalled EinE mechanism allows the contained experiment to have NSF access to a local copy of the project's file space.[2] However, this data (and all other data in the experimental run) will be wiped at experiment termination unless steps are taken to preserve it.

The experiment nodes can create files (e.g., log or trace files) to be made available after experiment termination by encrypting them and writing them on an optical CD/R or an external hard drive (XHD). Such a disk can later be removed by the operations staff and sent offsite to the experimenter's home location for decryption and analysis. It is also possible to have files persist across multiple invocations of the experiment by storing them on an external hard drive associated with this experiment. In this case, the DETER control plane associates the external hard drive with this malware experiment, and ensures that power up of the hard drive only happens during containment.

2.2.5 Data Extraction Phase

Phase S4: The experimenter terminates the experiment by requesting "swapout" from the outer DETER control plane. All experiment nodes (except the M/A node) are power cycled and their disks zeroed before the experiment is swapped out and nodes reused. It is still useful to perform a swap out operation on a zeroed experiment, as the nodes will be re-imaged with their initial pristine operating system images upon a future swap in. The M/A node will not be zeroed, but rather powered down at this point.

2.2.6 Experiment Termination Phase

Phase S5: The M/A node will be powered up and rebooted with a contained experiment extraction image, to clean it up and retrieve results *after* a malware containment experiment. The same malware containment mechanisms will be reused, but the "experiment" will consist solely of the Malware Archive node and its attached external hard drive. Once contained, the Malware Archive node will perform an AV scan and clean specific files or archives, moving them off the external hard drive and onto the Malware Archive's own hard disk. Upon completion, the external hard drive will be powered down again, and the firewall will permit the transfer of the specific clean files from the Malware Archive node to the experimenter's file system in the outer Users node.

Additional degrees of protection can be tailored for specific malware experiments, such as using more than one external hard drive or introducing additional reboot and zero disk cycles, on a case-by-case basis.

[2] Emulab stages the project files to the "inner" User node, copying from the "outer" Users node.

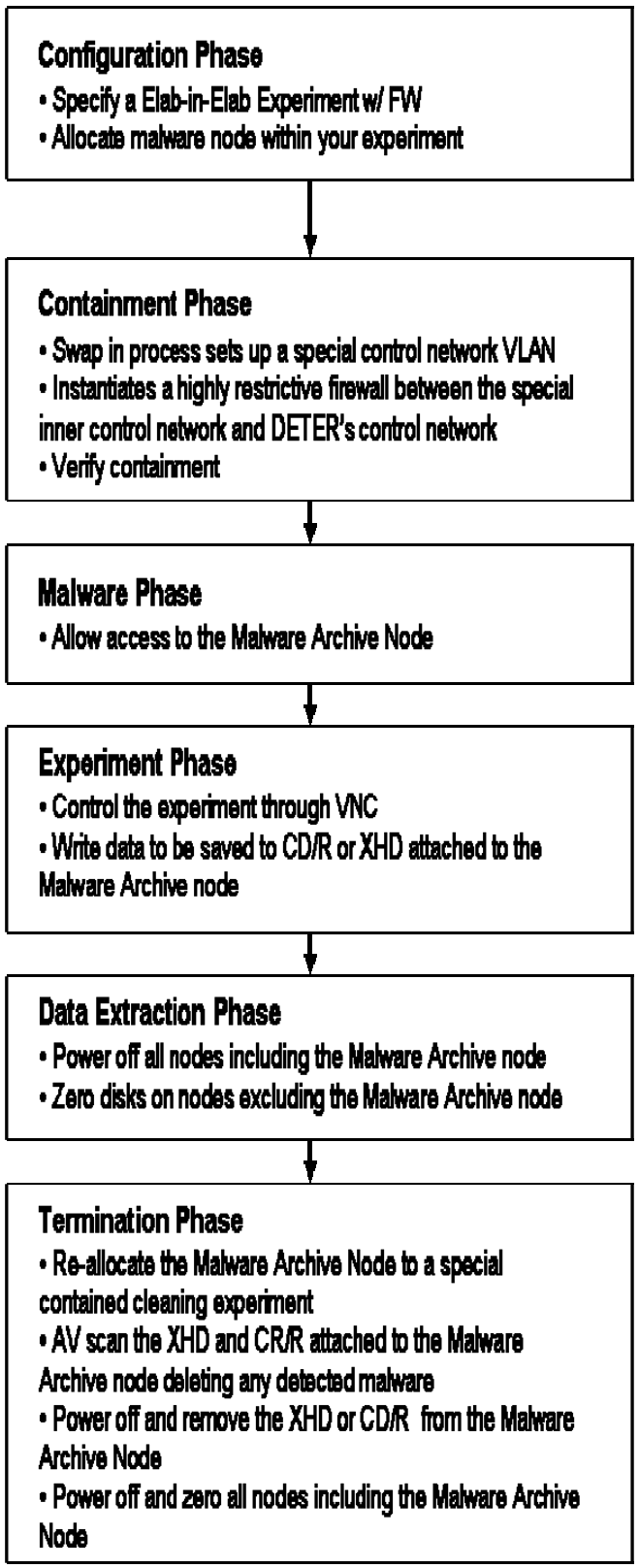

Figure 2 Phases of the Experiment

2.3 Discussion

Normally a DETER experiment can be swapped out and later swapped back in to continue. However, since a malware containment experiment is zeroed upon swap out and has no access to the normal NFS file system used to record intermediate experimental progress, an alternate mechanism, the dedicated USB- or Firewire-attached external hard drive (XHD), is proposed. The external drive will be designated in the DETER database as being dedicated to a single malware project. It should also be marked as a device that is "dirty", i.e., possibly infected.

Minimally, these drives will need to be zeroed, AV-scanned and cleaned before they can be reused by a different experimenter. During the experiment swap-out process, the external hard drive will be powered-off prior to the containment firewall being disabled. If necessary, it can be disconnected from the machine to which it was attached and put in a place for safe keeping until the experimenter requests that the drive be re-attached to another Malware Archive node within a firewalled EinE experiment for subsequent experimentation. The drive can also be sent to the experimenter's home site for further analysis. While these steps may not be relevant to all experiments, they are options that are available, and easy to perform when persistent storage is restricted to external hard drives, as opposed to the internal hard drives within experiment nodes.

3 Implementation Details

3.1 Malware Procedures

Malware for a particular project will be kept as an encrypted archive file (encrypted tar files or zip files are appropriate formats.) This file can be sent to the testbed operation staff on removable media, a CD/R or DVD/R disk, or on an external hard drive (XHD).

- Before introduction into the experiment, the operation staff, working on an isolated system, will decrypt the archive file and scan the contents with an AV scanner. The scanner should detect the malware; if it does not, the experiment fails the assumptions given above and it must not go forward. The scanner will be configured not to delete the malware from the archive for this test.

- The operations staff will load the disk containing the encrypted malware archive file into a drive on an experimental node that is dedicated to the project running the malware experiment. This node, called a Malware/Archive (M/A) Server, must be allocated to the malware experiment and

must be within the containment provided by the per-experiment firewall.

- The decryption key for the malware archive will be maintained in the primary testbed control database (i.e., on the outer Boss.) The key will be copied and made available to the experimenter on the inner Users machine, by being stored in a predetermined location in the project's directory. After the firewalled EinE experiment swaps in and containment is established and verified, the experimenter will be granted access to the Malware Archive server. Using the key, files from the encrypted archive may be decrypted and released, allowing subsequent behavior to be observed on one or more nodes in the experiment.

3.2 Running the Experiment

Malware experiments must be constructed to use a specific type of experiment that is known as a firewalled Emulab-in-Emulab experiment. This type of experiment creates copies of the *users* machine and the *boss* machine and isolates them from the real DETER by means of a firewall and vlan separation.

- A standard feature of a firewalled Emulab-in Emulab experiment is the panic button. In case of an unexpected malware leakage this button can be pressed via the external DETER web interface. This will disconnect the malware experiment from the rest of the DETER testbed.

- The firewall for containment will be highly restrictive, allowing only VNC traffic to pass on the control "network" (VLAN) between the experiment and the main ("outer") DETER control plane. "Inner" experimental nodes will have no control network interface connection to the "outer" common control network VLAN for all of DETER. It will also guarantee that all experimental network traffic is limited to VLANs with connections to other inner experimental nodes, or to the single firewall node.

- Specifically, the firewall will implement an application-layer proxy that inspects VNC protocol traffic in both directions. Only screen bitblts (pixel output) and cursor updates will be delivered in the outbound direction, and the outbound stream will be carefully monitored to prevent buffer overflow types of attacks. Input events (keyboard, mouse movement, mouse clicks) will similarly be checked in the inbound direction.

The use of the VNC virtual network console to allow safe experimenter interaction with a contained experiment is a central feature of our design.

- The user will use the VNC session to invoke the malware-extract command, which will cause a malware agent inside the firewall to decrypt and extract the malware binaries or other files from the malware archives and place them in the designated directories.

- AV scanners will run in the main DETER control plane, scanning the testbed control nodes and the firewall during the malware experiment. An intrusion detection system (IDS) box will also be scanning the control network interfaces for malware.

3.3 Ending an Experiment

Normally, a DETER experiment can be "swapped out" and saved to be later "swapped in" again. This is not possible for a malware experiment; the last step before containment is removed will be to zero-wipe all experiment nodes and power off all internal hard drives. Therefore a malware experiment cannot save its state between experimental runs. After all experiment nodes, including the firewall, are zero-wiped, the experiment will swap out. An AV scan will be done on the nodes that were allocated to the experiment and if they are clean then they will be returned the DETER available nodes pool.

3.4 Extracting Data and Results

It will be possible to explicitly save logs and data from one run to the next of the same malware experiment, by writing the data on an external hard drive (XHD) or by burning a CD/R. The XHD will be powered down individually before containment is removed, and powered up only after containment is re-established. It may also be encrypted, using a key that will be printed to the screen via VNC and must be copied out by hand.

- As an additional precaution, the XHD will be scanned with an AV scanner to remove any copies of the self-propagating malware, before it is powered down. Once the XHD drive is certified to be clean of malware, it may be powered-on in a non-contained experiment to retrieve data and files.

- Alternatively, the testbed OS image used for AV scanning may be used to transfer an explicit list of

files and archives from the XHD to an accessible file system on users or the experimenter's home network. The selection between these two options is a per project policy choice, to be made on a case by case basis when the experiment is approved or its secure requirements change.

4 Future Steps

The next step for the DETER testbed is to finalize the implementation and deployment options within the framework outlined in this paper, and begin coding up the primitive functions as modifications to existing Emulab software. In parallel with this development effort, a red team effort will be conducted to assess the overall residual risks, and to help improve the assurance of the malware containment software prior to roll-out as a production feature of DETER.

5 Acknowledgements

This material is based upon work supported by the Department of Homeland Security, and Space and Naval Warfare Systems Center, San Diego, under Contract No. N66001-07-C-2001. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the Department of Homeland Security for the Space and Naval Warfare Systems Center, San Diego.

6 References

[1] T. Benzel, R. Braden, D. Kim, C. Neuman, A. Joseph, K. Sklower, R. Ostrenga, and S. Schwab. Experience with DETER: a testbed for security research. In *Proceedings of the 2nd IEEE Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities (TridentCom 2006)*, Barcelona, SPAIN, March 2006.

[2] B. White, J. Lepreau, L. Stoller, R. Ricci, S. Guruprasad, M. Newbold, M. Hibler, C. Barb and A. Joglekar. An integrated experimental environment for distributed systems and networks. In *Proceedings of the Fifth Symposium on Operating Systems Design and Implementation (ODSI)*, pp 255-270, Boston, MA, Dec. 2007.

[3] R. Hicks, Déjà vu all over again, Virus Bulletin, Jan 1, 2007, http: //www.virusbtn.com/virusbulletin/archive/2007/01/ vb200701-comment .

[4] J. Y. Richmond and R. W. McKinney (editors), Biosafety in microbiological and biomedical laboratories, 4th ed, ISBN 0-7881-8513-6. http: //www.cdc.gov /od /ohs /biosfty/ bmbl4 /bmbl4toc.htm .

[5] T. Richardson. The RFB Protocol, RealVNC Ltd., Version 3.8, June 18, 2007. http: //www.realvnc.com/docs /rfbproto.pdf .

[6] J. Lepreau. Emulab: recent work, ongoing work. DETER Community Meeting, January 31, 2006. http: //www.cs.utah. edu/flux/testbed-docs/emulab-dev-jan06.pdf .

[7] S. Schwab, B. Wilson, C. Ko, and A. Hussain. SEER: a security experimentation environment. In *Proceedings of the DETER Community Workshop on Cyber Security Experimentation and Test 2007*, USENIX, Boston, MA, August, 2007.

[8] W. A. Arbaugh, J. D. Farber, and J. M. Smith. A secure and reliable bootstrap architecture. In *Proceedings of the 1997 IEEE Symposium on Security and Privacy*, May 4-7, 1997, IEEE Computer Society, Washington, D.C.

[9] T. Benzel, R. Braden, D. Kim, C. Neuman, A. Joseph, K. Sklower, R. Ostrenga, and S. Schwab. Design, deployment, and use of the DETER testbed. In *Proceedings of the DETER Community Workshop on Cyber Security Experimentation and Test 2007*, USENIX, Boston, MA, August, 2007.