

Lessons Learned in Using Live Red Teams in IA Experiments

David Levin

BBN Technologies
dlevin@bbn.com

Abstract

The DARPA Information Assurance (IA) and Operational Partners in Experimentation (OPX) Programs have conducted over a dozen laboratory-based experiments involving live red teams since April 1999. This paper explores some of the lessons learned from the integrator's perspective that are common among those experiments.

1. Introduction

In 1999, the DARPA Information Assurance (IA) Program began running experiments with live red teams to validate hypotheses posited by its technology performers. Many different approaches were used, such as penetration testing, independent validation and test (IVT), field testing, and laboratory-based scientific experiments. Many of the organizations involved in the DARPA IA/OPX program participated in these experiments: NSA, Sandia, SRI International, AFRL, NAI, Boeing, SCC, Teknowledge, and BBN, to name a few. This paper focuses on the laboratory experiments and does not reflect all of the red teaming performed for the IA DARPA programs during this period.

The DARPA IA Program process for experimentation has evolved to become the Operational Partners in Experimentation (OPX) experimentation process as well. Instead of performing engineering tests or a demonstration, the technologists (the defenders or blue team), mock adversaries (a red team), and a laboratory (or white) team worked together to formulate an experimentation scenario based on a real-world environment. They identified flags for the red team that were representative of real-world adversary goals, and they specified rules of engagement for both blue and red teams that strived for realistically unfettered defensive and offensive play while being constrained enough to allow repeatable results.

All of the experiments produced results, insights, and guidance; some of it is still well accepted today; however, some of the experiments ended with one or more team dissatisfied with the experiment or its results.

This paper analyzes those concerns from a process perspective, compares them to results from other IA and OPX laboratory-based experiments, and concludes with some suggestions for future experiments.

The overall IA/OPX program laboratory-based experimentation process [1] was designed to help provide objective information and insight into the IA program's "Grand Hypotheses":

- Trusted systems can be composed from untrusted components
- Layered Defenses improve system assurance
- Dynamic Defenses improve system assurance.

Figure 1 provides a graphic representation of the overall IA Experimentation Process from idea selection and review to experiment design and execution. This paper focuses on the process used to design and implement an experiment and analyze its results. Discussion of the overall experimentation process was described in [1].

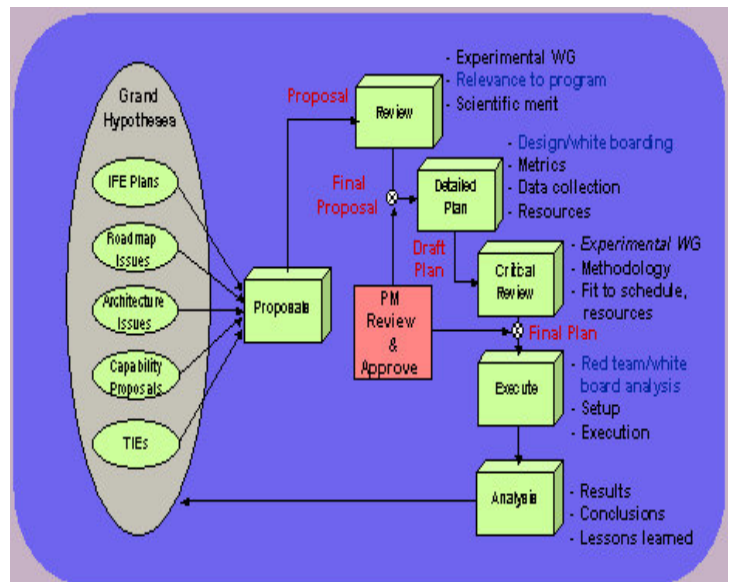


Figure 1. IA Experimentation Process

The goal of the IA/OPX experimentation program was to develop and execute well-posed scientific experiments that are quantitative and repeatable. A successful experiment could help map the problem space and probe for boundaries. It could also test proposed defensive and offensive approaches and assumptions quickly and effectively, and could provide insights and guidance for future efforts. It could collect hard data on an elusive adversary approach or behavior.

2. The design and execution of an IA/OPX laboratory-based experiment

The experimentation process uses a hypothesis-based approach to answer important questions about a program's technologies in an objective and repeatable fashion. The experimentation process consists of the following steps:

Generate hypotheses. The teams meet to select an experiment hypothesis (or hypotheses) that will yield an interesting experiment with results that will provide insight into the mechanism(s) under consideration. For example, one of the hypotheses from the first Hardened Client (HC) experiment was "sensitive data on the HC laptop are protected from unauthorized access by an adversary with physical access."

Develop claims. With a common set of hypotheses, the red and blue teams separate. The blue team develops the *claims* made and approaches supported by the mechanisms that would best serve the hypotheses. For the HC experiment, one of the claims was "Protected volume cannot be mounted by unauthorized users," and the approach was to encrypt the protected volume.

Develop attack trees. In parallel, the red team develops a complete *attack path diagram* (the *attack tree*) that shows all possible paths to refute the hypotheses. The attack path diagram considers all attack means and is not restricted to cyber attacks. For the HC experiment, one path was "Brute force cryptographic attack against the encrypted volume" and another was "Convince the user to unlock (decrypt) the volume and use snooping techniques to acquire the contents."

Hold whiteboard session. Again the teams meet, this time to match *claims* to *attack paths* and eliminate uninteresting claims and paths. In this context, a claim and attack path are uninteresting if they exactly match each other; e.g., the claim of "a strong cipher" and a "long time to crack the cipher" attack. Other claims or attack paths are jointly ruled out of bounds by both red and blue teams. This leaves the "interesting" claims and attack paths. Selection is based primarily on what attacks are expected to yield the most interesting results relative to the hypotheses, with some consideration for what could be accomplished in the available execution time.

The teams also reach agreement as to experiment measures and metrics in the whiteboard session. The most common metric has been Red Team Work Factor (RTWF), which measures time spent by the red team in all aspects of its attack development and execution and compares the measurements from two separate attacks, usually one with defenses and the other without but otherwise identical.

The final step in the whiteboard session is to complete the rules of engagement (ROE) for the experiment. Key aspects of the ROE are *process focused*, i.e., code freeze and setup freeze dates, and *procedural*, i.e., what range of activity is permitted for each team.

Create and populate attack trees. Now the teams separate to prepare for the experiment. The blue team works on its defenses, the red team on its attacks, and the white team on experiment setup.

Hold midcourse correction session. Complex or contentious experiments often require a final review to ensure that attack trees and defenses remain compatible with each other. It is better to find out before the experiment execution that the red team is attacking a known vulnerability or the blue team has changed the definition of a *network attack*.

Establish exemplar of the experiment setup. In experiments involving full disclosure, the *exemplar* provides the red team with hands-on access to the technology. How much and what types of access the red team has is based on the goals of the experiment and agreements reached in the Whiteboard Session. It has ranged from none to complete physical access to the experiment setup including design documents and source code. With access, the red team uses the exemplar to work on testing and debugging its technical attack paths.

Freeze dates are very important as they define the points at which one team is "done" with their experiment-related work so that the next team has a stable base from which to work. Changes to anything after a freeze date, i.e., the experiment setup or the exemplar, are only permitted when all the teams agree. For example, the *code freeze date* signals the end of the blue team work and is usually the last dependency for the white team's work. The *setup freeze date* signals the end of the white team's work and is usually the last milestone before the red team receives an exemplar of the experiment setup. Ideally, setup freeze follows code freeze.

Set up the experiment equipment. In preparation for the experiment, the white team sets up and configures the network per the experiment scenario requirements. This includes adding non-intrusive mechanisms, such as logging, time-stamping, and data set repositories, and supportive mechanisms, such as traffic generators to provide background traffic.

Run the experiment. Experiment execution consists of red team attacks, as specified by the experiment plan. During the experiment execution, the white team resets the necessary logs (and other key system parameters) at the start of each run. Depending on the rules of engagement (ROE) the blue team may or may not be authorized to touch the experiment setup during the execution.

Hold a hot wash. A *hot wash* immediately follows the execution in which the white, red, and blue teams compare notes and come to an agreement on what happened in the experiment. The hot wash serves to review, document, and share preliminary information on the actual experiments run and the initial results. A secondary goal is to ensure that the white, red, and blue teams shared a common perspective on what happened during the experiment. The hot wash allows for correction of any time skew (e.g., between red and blue team perspectives), and missing (or missed) steps can be duly noted and inserted. In addition to the joint briefing, all teams prepare and present their perspective (to ensure that differences are captured). The hot wash is also the genesis of the experiment's lessons learned documents (joint, blue, white, and red). Attendees at the hot wash include those directly involved in designing and executing the experiment (i.e., the three teams), the appropriate DARPA Program Manager, the Program's Principal Investigator, and other invited principals.

Warm wash. After the results are analyzed, possibly requiring weeks or months, a *warm wash* is conducted to present the final results, lessons learned, and future plans. It is open to the entire IA/OPX community. Briefing materials from these meetings, as well as other experiment documentation, can be found on the IA/OPX web site.

This process has been used successfully for many experiments since April 1999. While this process seeks to accommodate the different needs of the white, red, and blue teams, each team preserves its unique perspective throughout the experiment.

3. The IA/OPX experiment teams

As indicated above, the Experiment Team for laboratory-based experimentation consists of three independent teams that work cooperatively to design, execute, and analyze the experiment. While the unifying principle is cooperation, each team has a different motivation and perspective for being involved. The blue and red teams have a strong involvement in the outcome of the experiment as each is focused on a specific outcome from the experiment. The blue team represents the technologies or mechanisms under analysis and usually is involved in the creation of the defensive mechanisms. The red team represents the adversary or attacker.

In a well-designed experiment, these opposing needs are used to focus and drive the experiment. The white team serves as objective observer or referee and is responsible for setting up and maintaining the experiment's equipment.

These differing perspectives are discussed in the following sections.

3.1. Blue team

The blue team is responsible for the defensive mechanisms being used in the experiment. Besides the obvious need to see these mechanisms succeed, the blue team hopes for diversity and completeness in the red team attacks.

The blue team looks at the experiment's hypotheses as the motivation for an exhaustive test of the defensive space. To the blue team, an ideal experiment would be mechanism focused, its attack path diagram would be completely populated, and its execution would exhaustively test all of the paths against the mechanisms.

In this blue-tinged world, no paths would be pruned, even if they were redundant or impractical so that the attack plan would closely resemble a full-scale engineering test of the technology mechanisms since all paths would be tested, regardless of merit. This further implies that the red team would continue to test paths even if sufficient experimental data was already in hand.

3.2. Red team

Like the blue team, the red team has a significant interest in the experiment's outcome; they want to prove that their own effectiveness in thwarting those defensive mechanisms exceeds the capabilities of the mechanisms. It is important to the red team that they successfully defeat the mechanisms as quickly and efficiently as possible.

The charts in Figure 2 [6] show the different phases of red team efforts and a notional sense of the relative time devoted to each.

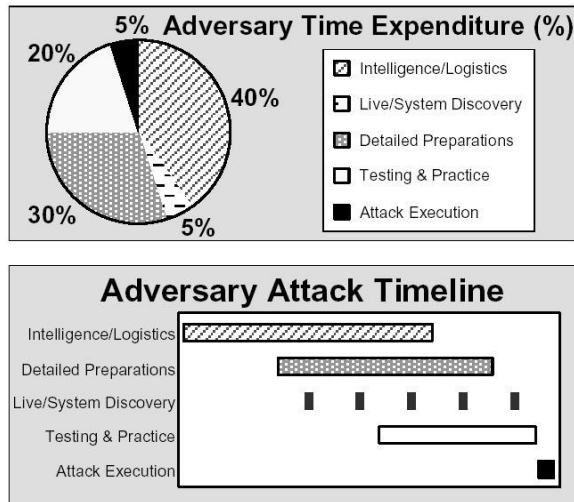


Figure 2. Adversary work distribution

Based on the data available to the authors [6], these percentages are accurate for attacks against live systems and within the experimental context. Note from Figure 2 that the red team effort during the experiment execution is only 5% of the overall effort.

The red team is driven by the challenge to find ways to thwart the defenses. After carefully studying the defenses, the experiment’s hypotheses and goals, and the experiment’s setup (intelligence gathering), a disciplined, process-driven team will build an exhaustive attack path diagram (attack tree). Live system discovery, additional intelligence gathering, and detailed planning will first help in populating the tree and then help prune away improbable and redundant paths.

The red team prides itself on analyzing the blue defenses, thinking outside the box, and finding ways to slip around those defenses (by exploiting omissions, failures, and oversights). A red team will often describe its work as finding the blue team’s abstraction model and dropping below it to mount a successful attack, similar to dropping below the radar for an aerial attack. In this mode of operation, they have no interest in “boring repetitive” work, except when it is part of an overall attack plan, so it is likely that they will only exercise a few paths from the whole tree during the experiment.

Since many of the experiments use red team work factor (RTWF) as a metric, the red team may look for a *silver bullet* attack that will keep its work effort to a minimum, an attack that defeats the defensive mechanisms (achieves all flags) with a single attack path. The red team believes that if the silver bullet attack captures the flags, why discover another attack path? Other means to achieve the same result are superfluous and unnecessary.

Failing to find a silver bullet, high regard is given to *one shot* attacks. These are attacks that are fast and direct,

though often not elegant. For example, when asked to find and modify a file on a Microsoft Windows box, the red team ran a simple DOS batch file to exhaustively search for the file and then used the DOS *echo* command to add the flag to the file. The point here is that the blue team was looking for an attack that conformed to their model, i.e., use Microsoft Word to modify the file, while the red team was emulating an adversary who is likely to be “sneaky and lazy” in achieving his goals, i.e., use the simplest tool for the job.

In the context of this experimentation, a red team’s goal is often to find one unlocked door, exploit it, and move on. There is no need to perform an exhaustive test of the other doors beyond this one. Of course, the charter and guidelines set forth by the experiment team can govern such an outcome.

The red team looks at the experiment’s hypotheses through the attack trees that are devised and analyzed to plumb the defensive depths. Elegance is perceived in terms of the ability to successfully capture the flags, the simple, one step attack being preferred to more complex ones. Attacks on the attack tree are ranked based on their effectiveness and ease, not compliance to any other model or standard of beauty.

Further discussion of the red team can be found in [3], [5], and [6].

3.3. White team

Sometimes the most difficult aspect of the white team’s role is keeping the peace. Development times often grow, leading either to schedule delays or impacts on the other teams that depend on that work. In fact, a recurring problem in all experiments, except those involving mature technologies, is the delay caused by missed deadlines. The white team must continue its own setup work and provide alternatives to the red team when the blue team is late. It also falls to the white team to resolve other conflicts, such as flag definitions, when the red and blue teams are at an impasse.

During experiment execution, the white team performs measurements, interprets ROE, and makes the judgment calls as to the interpretation of system behavior. For example, deciding whether a red team’s partial success in crashing a system is sufficient to grant them that flag or not.

The white team has traditionally had the responsibility of assuring that all experiment documentation was complete and accurate. They generally brief the experiment on behalf of the team.

While more can be said about the specifics of the experimentation process and the teams that implement

them, several example experiments are described to convey these specifics.

4. Experiment highlights

Four example experiments are described, from the first IA experiment on layered protection conducted in 1999 to one of the last experiments involving the hardened client conducted June 2002. Each experiment is briefly described followed by specific notes.

4.1. Layered Defense Experiment

This seminal IA experiment was conceived to investigate the impact of layering defenses, or defense in depth, for improving the protection of computer systems. [4] While the concept of defense in depth was in wide use at the time, there was no evidence to support the claims that adding computer defenses together (also referred to as “composing defenses”) improved the protection of the system. The experiment was designed to “Compare attacker work factors as more defense/prevent layers are added in a client-server database architecture.” The hypothesis of this experiment was “Adding layers has at least a cumulative impact on adversary work factor” as the expectation was that RTWF would increase as defensive layers were added.

The red team was given Confidentiality, Integrity, and Availability flags within an imagery database and had to capture the flags starting from both outside and inside the boundary device (e.g., firewall) that protected the imagery database. The white and red teams measured the work factor expended by the red team while attaining flags.

The experiment results were revealing. The increase in RTWF was not immediately evident in the hot wash since the initial analysis focused primarily on differences in execution times. Further analysis showed that while the red team was consistently able to attain its flags with the defenses in place, these successes came at the cost of increased attack design and testing, not execution, times.

The experiment also demonstrated that “Depth without Breadth is irrelevant,” where Depth means multiple mechanisms against a particular attack class and Breadth refers to multiple mechanisms across multiple attack classes. In other words, multiple defenses protecting one mechanism provide no deterrent or defense against attacks that do not affect that mechanism. [4]

Less obvious, but equally important, was the realization that while defensive layers can be incomplete, they can move attack points to manageable places. Thus, incomplete defenses can be used to channel the attacker into the paths that the defender chooses to defend.

Finally, since each defensive layer has unique dependencies, these dependencies must be “managed” to prevent one from amplifying another.

The results, insights, and disagreements from this experiment led to a series of experiments to further investigate each of the technical lessons learned in this first experiment. [7] During the course of this series of experiments, the different types of red team efforts were analyzed (see [3], [5], and [6]). The disagreements are discussed in Section 5, Lessons learned.

On the process side, this experiment uncovered the importance of the teams openly sharing information, the need for multiple whiteboard sessions for complex experiments, and the need for common definitions and terminology.

4.2. Dynamic Defense Experiment

The Dynamic Defense experiment differs from almost all the others because its success was based on hiding key information from the red team. Since the Layered Defense experiments had suggested the importance of the *intelligence gathering* and *discovery* aspects of the red team effort, this experiment was conceived to measure the impact on red team effort if they were consistently denied those gathering and discovery capabilities. The hypothesis for this experiment was that “Dynamic network reconfiguration effectively degrades the attacker’s ability to map the network, and hence increases attacker work factor and improves system assurance.”

This hypothesis was arrived at through the following chain of logic. [6] Adversaries follow a process. This process is primarily driven by risk reduction. If the adversary’s perceived risk is higher than his threshold, he will not execute the attack. Accordingly, adversaries spend significant time in intelligence gathering and discovery phases and the actual attack phase is short to reduce their risk (as illustrated in Figure 2).

We conducted an experiment using a technology called Dynamic Network Address Translation Reconfiguration (DYNAT) [2] that gives the appearance of the network topology being reconfigured “on the fly” by continually remapping hosts to different IP addresses. Red team efforts to map the network topology were measured. The red team first attempted to discover the static network topology and quickly succeeded. Then the mechanism was added to dynamically change the network topology and, without knowing what had changed, the red team was challenged to rediscover the network topology. The result was that the dynamic remapping increased the red team *discovery* phase effort by a factor of 4 with little increase in blue team effort. More importantly, the red team’s

behavior supported the hidden hypothesis that delaying intelligence gathering delays the attack phase.

While the red team was stymied by this mechanism, one artifact of the experiment may have further confounded the red team. Since the network traffic used in the experiment was unusually homogeneous (similar in session duration and in packet and session size), the red team was unable to analyze the traffic in order to intuitively determine the topology.

While the DYNAT mechanism was effective in confounding the red team in this experiment, the experimenters realized that wide deployment of this mechanism would potentially change adversary behavior to accommodate it. For example, other technologies such as VPN's also hide some routing information and red teams have learned techniques to get around this information hiding. By using a novel technique to hide data, the red team was unaware of the deception and reattempted discovery. Once the technology was explained, the red team quickly considered countermeasures.

Finally, since the decision to withhold information was part of the initial experiment design, cooperation between the teams remained high throughout this experiment.

4.3. Hardened Client Experiment

The Hardened Client (HC) Experiment was the first OPX experiment based on inputs from a transition partner and was conducted in October 2001. [8] In the OPX program, operational commands were approached to find operational organizations willing to partner with DARPA to bring research technologies to the warfighter. The Marines in the Pacific Command (MARFORPAC) were the first OPX partner and they brought an important problem. Marines are regularly deployed to remote locations where their laptops are at increased risk of attack. Experiment hypotheses were defined to capture the Marine's goals to protect information on their laptops and an experiment was designed to test the hypotheses.

The following two hypotheses addressed the Marine's concerns, first "These technologies protect the hardened client by jointly filtering the attack space down to a well-defined residual risk space", and second "Sensitive data on the hardened client is protected from unauthorized access by an adversary with physical access". The first addressed the concerns of using the laptop in a hostile network, and the other addressed data protection if the laptop fell into unauthorized hands.

As shown notionally in Figure 3, the scenario for the experiment was a Marine with his laptop deploying to a remote site for a humanitarian effort and included three

specific potentially hostile locations: 1) use of the HC laptop in the headquarters LAN (insider attack); 2) use of the HC laptop in a hostile local area network (LAN); 3) theft of the HC laptop. In both LAN cases, network attacks were launched locally (from within the LAN) and remotely (outside the LAN perimeter). In the theft case, the adversary had full access to the HC laptop.

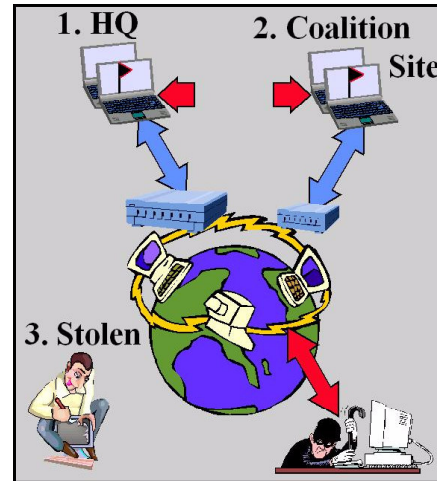


Figure 3. HC Victim and Attacker locations

The flags for the experiment included modification or theft of a sensitive file from a remote location (network attack), modification or theft of a sensitive file from a local location (insider attack), and accessing the sensitive file while in physical possession of the hard drive. As usual, the principal metric was Red Team Work Factor, as measured in the comparison of attack times in protected and unprotected modes.

Several DARPA and commercial technologies were combined to protect Microsoft Windows 2000, Microsoft Outlook, and Microsoft Office running on a laptop.

- An Embedded Firewall (EFW) network interface card (NIC) [9], [10] was used to protect against unauthorized network traffic.
- Safe Email wrappers [11] were used to protect the laptop from malicious code delivered through email.
- PGPDisk (commercial version, formerly open source) was used to encrypt sensitive files.
- East-Tec Eraser 2000, a commercial security solution for protecting data privacy, was used to protect sensitive files, such as the paging surface and temporary files left behind by applications, that cannot be protected in the PGPDisk encrypted volume. East-Tec Eraser 2000 is designed to completely eliminate sensitive data (beyond recovery) from the computer.
- Supporting scripts and software, such as scripts to run East-Tec Eraser 2000 at system shutdown and plug-

ins to enable Safe Email wrappers to protect the Microsoft Exchange mail client.

The results of the experiment showed that the protection mechanisms added work (“raised the bar”) for the red team. The mechanisms successfully blocked several attacks, delayed almost all attacks, and constrained the feasible attack space for the red team because the red team’s attacks were channeled by the mechanisms.

While the mechanisms were effective in increasing RTWF, the red team was able to capture all its flags except reading the dismounted, encrypted disk. Without passphrase, the dismounted volume was unreadable. “Due to the human component of the experiment and the email attack time dependencies, it was not possible to extract statistically complete results from the data, thus the results are largely subjective.”[8]

The red team’s successful attacks uncovered mechanism failures due to bugs and to vulnerabilities, all of which were subsequently corrected. A complete analysis of the experiment results is in [8].

The HC experiment did demonstrate an “improved defensive posture” for the HC laptop and warns, “Once fully integrated and configured by an administrator, these technologies effectively worked to better defend the clients against attack. While this experiment shows the value in protecting the endpoint hosts, care should also always be taken in designing a secure networked system that does not invite the adversary in through more vulnerable attack paths.”[8] So while the HC experiment results supported the hypotheses, the red team and some members of the blue team felt that the experiment had, to some degree, missed its mark.

The red team was frustrated that:

- The experiment ROE were too constrained for a realistic attack scenario, since vulnerabilities that would have made the protections moot were not simulated or were ruled out of bounds. For example, in the experiment’s EFW policy for HQ, both backup and remote system administration were disabled and if either were enabled, simple attacks paths are available.
- A version skew between the red team’s exemplar and the actual experiment setup, due to a missed code freeze, led to the red team developing several exploits that only worked on the exemplar.

The wrappers members of the blue team felt the experiment had done very little to increase their knowledge since the red team had used a *silver bullet* attack, used it repeatedly, and exploited a vulnerability known to the blue team but not shared with the red team.

While the experiment team reached agreements on all of these points, the team proposed a follow on HC experiment to resolve any remaining concerns. In Section 5, “Lessons learned,” this paper will suggest guidelines to avoid these conflicts in future experiments.

4.4. Hardened Client II Experiment

The first HC experiment did not address a key question about wrappers [11] – Can an adversary bypass wrappers or not? We designed a second HC experiment to address that question directly and chose the following hypothesis: “The execution mediation provided by the Wrappers and its non-bypassable mechanism cannot be circumvented, regardless of the mediated process or its offspring, or the effort level of the adversary.”

The experiment focused on testing the following ordered list of sub-hypotheses:

1. *Is it possible to get to the protected OS service without mediation?* (H1) That the new bypass protection mechanism prevents a wrapped process from utilizing the OS kernel-resident services invoked by a mediated interface except by passing through the mediator. For example, the OS service corresponding to a mediated kernel32 API (such as CreateProcess or CreateFile) cannot be successfully invoked by a wrapped process without doing so through the mediated version of the kernel32 API.

2. *Can the mediated process stop non-bypass protection?* (H2) That a wrapped process cannot disable the new bypass protection. This would include exploiting published APIs for loading or unloading kernel modules (including drivers).

3. *Can the mediated process create an unmediated child-process?* (H3) That a wrapped process cannot spawn an unwrapped process or otherwise cause unwrapped code to be executed. This was the major untested hypothesis from the HC1 experiment because the bypass exploit worked.

Only potential attacks that tested one of these three hypotheses were considered for this experiment. Attacks that tested hypotheses H1 and H2 were the most interesting to the blue team, while attacks that tested hypothesis H3 were more interesting to the red team. The combined experiment team agreed that attacks testing hypotheses H1 and H2 had resource priority over H3 attacks.

Technology. Wrappers are a software mechanism designed to allow the execution of untrusted files, e.g., email attachments, while preventing any malicious behavior from being spawned by that untrusted file. Wrappers mediate, or monitor, all behavior from the

spawned process by mediating calls to the supporting services.

When malicious code embedded in an email message or an attachment is executed, the wrappers monitor the behavior and disallow suspect activity. Rule sets are written that specify allowed behavior and deny all else.

When opening an attachment, say a .ppt file, from within the wrapped email application, the email application starts the associated applications, in this case PowerPoint, and the spawned application is automatically wrapped with the context specific wrapper as well. Additional rule sets that apply to the PowerPoint application are used and they also allow pre-defined normal activity and deny all else.

The new Wrappers non-bypass mechanism protects the Wrappers mechanism from any attacks specifically aimed at disabling the mediators.

Metrics. The primary success metric for the red team was their ability to develop a successful attack that refuted hypothesis H1 or H2. A secondary metric of interest was the relative RTWF expended by the red team in crafting all attacks, including the successful attack. Finally, since multiple attacks were considered, the attack with the lowest RTWF would be considered the most effective attack. The intent in choosing these metrics was to produce quantitative results that would provide a measure of the “attack-resistance” provided by Wrappers if the non-bypassable claim was refuted. The total RTWF for each attack included the time it takes to do network discovery, develop and test the exploit, and successfully run the exploit to capture the flag. The red team was responsible for capturing and reporting the other times and level of effort expended for each experiment attack. No other metrics were collected for this experiment.

Rules of Engagement. There was to be no active blue team defense during the experiment on the Wrapped workstation other than the Wrappers code to protect the flag. The red team had full knowledge of all details of the experiment, including source code, defenses, and applications. Exploitation of kernel bugs that allow a user-mode process to modify the kernel were outside the scope of this experiment.

Each red team attack started from within a Wrappers mediated process and focused on capturing a single flag, adding a subkey to a certain registry key that should be prevented by the Mediator (the Wrapper). The red team tested different procedures for capturing this flag.

The experiment was run on a computer running Windows 2000 Professional 5.00.2195 Service Pack 2.

Results. The wrappers and the non-bypass mechanism successfully blocked all attacks obtained from the network

and the blue team’s regression suite, including the *silver bullet* attack from HC 1. The result of the experiment was that the red team was able to bypass the non-bypass mechanism and the blue team was able to demonstrate that the vulnerability was known before the experiment. The red team used a new *silver bullet* attack to refute hypothesis H1 and could have used it to refute H2. No attacks were offered against H3.

RTWF. The successful exploit cost 137 red team hours to develop and test. The cost for the suite of unsuccessful attacks is less than 8 red team hours.

The red team attack diagram identified 22 terminal nodes (attack paths) and though rigorous evaluation of the diagram is incomplete, two were marked out of scope. The red team tried additional attacks from the tree against the exemplar, such as attacks that ignored the Wrappers, and found they were ineffective.

As a result of this experiment, the non-bypassable wrappers team is preparing for an exhaustive test of the wrappers and non-bypassability functionality.

5. Lessons learned

Many successes and challenges have resulted from experimentation in the DARPA IA and OPX Programs. A significant body of experience has been developed that can aid future researchers and experimenters as they design and execute IA experiments. We group the lessons learned into successes to emulate and challenges to avoid.

5.1. Successes to emulate

Share everything. Security by obscurity is not good during IA experimentation. Open communications and cooperation between red and blue teams consistently improves the quality of experiments. There were far more failures from miscommunication and data erroneously held back than from saying too much. As noted in Section 4.2, the Dynamic Defense experiments required secrecy, but this should be the exception.

Write everything down. It is critical that team decisions be written, so they can be referred to later.

Common understandings. Not only is it important for team members to talk to each other, but they must also understand each other. Expectations, such as “run all the tests including the uninteresting ones” or “use more than one attack even if you have a *silver bullet*,” need to be raised and resolved before the experiment execution starts.

Some experiments are complex enough that *multiple whiteboard sessions* are required to ensure common understanding.

What is RTWF? RTWF is still a largely subjective metric. Without precise definition, it can lead to behavior such as *one shot* and *silver bullet* attacks that the blue team does not expect. Consider using credits for attack diversity and multi-kill capability to mitigate this result. Look for scoring incentives that drive the red team to create broader, more populated attack trees.

Human-in-the-loop experiments. Human-in-the-loop experiments are more difficult to design than fully automated ones. As is true of all experiments involving human reasoning, it is important to *order experiment execution* to minimize the impact that red team learning has on the results. In general, this implies ordering attacks so that the more remote attacks are conducted before local ones, in experiments comparing work factor *with* and *without* a mechanism, the *without* case should almost always precede the *with* case.

Set a schedule and keep to it. Without a schedule and adhering to its agreed upon milestones, staff time is wasted and results can be affected negatively by the resultant version skew.

Experimentation is best used for considering hypotheses involving *mature mechanisms* so that a systems, not a mechanism, focus can be used in experiments.

As the tested mechanisms mature, experiments setups should move closer and closer to an *unconstrained system* since real-world adversaries operate in that arena.

5.2. Challenges to avoid

Do not force fit an engineering test into an experiment, as the experimentation process is an expensive avenue for penetrate-and-patch testing. If the blue team wants an exhaustive test suite, find an alternate venue. Using a live red team is too expensive to perform engineering tests, and it is not usually their core competency.

Since both the blue team's objectives and the red team's attack plans can substantially differ from the goals of the experiment, it is very important for the *white team to review* both to avoid wasted effort.

A recurring challenge is the *lack of documentation* and *mismatched expectations*. This is best managed by ensuring all documentation is provided ahead of time (and read by the appropriate teams) and by crafting the flags and scoring to meet both teams' needs. For example, it is counterproductive when one team surfaces a document towards the end of an experiment to discredit the other team (or to save face) when the document could have been used earlier to avoid the conflict.

6. Summary

Overall, the IA/OPX experimentation process has successfully provided a disciplined and effective approach to analyzing cyber technologies, both mature and immature, as they transition from the world of research to the operational world. This process stands as a valuable alternative to non-repeatable, synthetic demonstrations of technology that offers little insight into improvements in information assurance.

It is important to consider the experiment results as well as the lessons learned to ensure that the experimentation process evolves to meet the challenges that have detracted from previous experiments.

7. References

[1] J. Lowry, K. Theriault, "Experimentation in the IA Program", *Proceedings of the DARPA Information Survivability Conference & Exposition 2001 (DISCEX II), Volume II*, IEEE Computer Society, Anaheim, CA, 12-14 June 2001, pp. 134-140.

[2] D. Kewley, R. Fink, J. Lowry, M. Dean, "Dynamic approaches to thwart adversary intelligence gathering", *Proceedings of the DARPA Information Survivability Conference & Exposition 2001 (DISCEX II)*, Volume I, IEEE Computer Society, Anaheim, CA, 12-14 June 2001, pp. 176-185.

[3] B. J. Wood, R. A. Duggan, "Red teaming of advanced information assurance concepts", *Proceedings of the DARPA Information Survivability Conference & Exposition 2000 (DISCEX I)*, Volume II, IEEE Computer Society, Hilton Head, SC, 25-27 January 2000, pp. 112-118.

[4] D. Kewley, J. Lowry, "Observations on the effects of defense in depth on adversary behavior in cyber warfare", *Proceedings of the IEEE SMC Information Assurance Workshop*, West Point, New York, June 2001.

[5] G. Schudel, B. J. Wood, "DARPA Information Assurance Program Red Team Experiments," *2000 IEEE Military Communications Conference Proceedings*, Los Angeles, California, October 2000.

[6] G. Schudel, B. J. Wood, "Adversary work factor as a metric for information assurance", *New Security Paradigm Workshop*, Association for Computing Machinery, Special Interest Group on Security, Audit and Control, Cork Ireland, 18-22 Sept 2000.

[7] D. Kewley, J. Bouchard, "DARPA Information Assurance Program dynamic defense experiment summary", *IEEE Transactions on Systems, Man, and*

Cybernetics-Part A: Systems and Humans, Volume 31 Number 4, July 2001, pp. 331-336.

[8] D. Ryder, D. Levin, J. Lowry, "Defense in Depth: A focus on protecting the endpoint clients from network attack", *Proceedings of the IEEE SMC Information Assurance Workshop*, West Point, New York, June 2002.

[9] T. Markham, C. Payne, "Security at the Network Edge: A Distributed Firewall Architecture", *Proceedings of the DARPA Information Survivability Conference & Exposition 2001 (DISCEX II)*, Volume I, IEEE Computer Society, Anaheim, CA, 12-14 June 2001, pp. 279-286.

[10] C. Payne, T. Markham, "Architecture and Applications for a Distributed Embedded Firewall", *Proceedings of the 17th Annual Computer Security Applications Conference*, New Orleans, LA, December 2001.

[11] R. Balzer, N. Goldman, "Mediating Connectors: A Non-ByPassable Process Wrapping Technology", *Proceedings of the DARPA Information Survivability Conference & Exposition 2000 (DISCEX I)*, Volume II, IEEE Computer Society, Hilton Head, SC, 25-27 January 2000, pp. 361-368.

8. Acknowledgements

This work is supported by the Defense Advanced Research Projects Agency (DARPA) under Contract # F30602-98-C-0012.