I'm Not Sure If We're Okay

Uncertainty for Attackers and Defenders

Mark E. Fioravanti II
Department of Computer Sciences
Florida Institute of Technology
150 W. University Blvd.
Melbourne, FL 32901
mfioravanti1994@my.fit.edu

Matt Bishop
Department of Computer Science
University of California, Davis
1 Shields Ave.
Davis, CA 95616-8562
mabishop@ucdavis.edu

Richard Ford
Department of Computer Sciences
Florida Institute of Technology
150 W. University Blvd.
Melbourne, FL 32901
rford@fit.edu

ABSTRACT

Asymmetry and uncertainty have been written about at length in the context of computer security. Indeed, many cutting edge defensive techniques provide system protection by relying on attacker uncertainty about certain aspects of the system. However, with these defensive countermeasures, typically the defender has the ability to derive full knowledge of the system (as is the case in, for example, Instruction Set Randomization), but the attacker has limited knowledge.

In this paper, we concern ourselves with the case in which neither the attacker nor the defender have perfect knowledge of the system, but where the level of uncertainty tolerable to both parties is different. In particular, we explore scenarios where the attacker's need for certainty is lower than that of the defender, and ask if non-determinism can be used as a weapon. We provide an example in the malware arena, demonstrating the use of quorum sensing as a potential application of this technique. We argue that this idea of mutual uncertainty is a new paradigm which opens the way to novel solutions in the space.

CCS Concepts

•Security and privacy → Operating systems security; *Software security engineering;* **•Social and professional topics → Computing / technology policy; •Computer systems organization → Architectures;**

Keywords

uncertainty, asymmetry, randomization, defense, computer security

1. INTRODUCTION

Uncertainty. The term has become something of a buzzword in the security community of late, with ideas such as Address Space Layout Randomization (ASLR) [25, 27] and Moving Target Defenses [10] garnering many column inches at conferences — and funding announcements — over the last few years. However, in almost every case, these solutions are based on the defender (i.e. the person who deploys the technique) having the ability to derive full knowledge of the system, leaving the attacker at an information disadvantage, essentially creating a "knowledge asymmetry" between the two parties. The idea is that the attacker cannot overcome this disadvantage, and will therefore be rendered unsuccessful in attacking the system.

We posit that approaches based on such knowledge asymmetry are only one possible way of leveraging uncertainty in the computer security arena. In this paper we explore an alternate approach: systems where both the attacker and the defender have uncertainty about the state of the system being defended. Whereas in the example above knowledge asymmetry can be breached by the inadvertent leaking of information from one party to another, a defender cannot inadvertently leak something she does not have. Operationally, in some instances, one party can tolerate more uncertainty than the other yet still complete her mission. In such a case, there is an opportunity to exploit this difference.

In what follows, we present a general model of the joint uncertainty paradigm and derive some implications about this symmetry. We then describe an experiment in "digital quorum sensing". This technique, inspired by bacterial quorum sensing, allows for weak exchange of information between a group of loosely connected entities. Using digital quorum sensing, we show how the attacker can leverage a technique that provides (at best) equal information to attacker and defender, yet is beneficial for the attacker only.

Finally, we consider the utility of the technique in other possible settings.

2. PARADIGM AND MODEL

Let P be a random variable with value drawn from the set $\{attacker, defender\}$ (abbreviated "a" and "d", respectively). Consider a second variable X that encodes some useful information about the system under question. Let $H(x)$ represent the uncertainty (entropy) in the random variable x.

The usual form of defense is to have:

$$H(X|P = \text{a}) > 0 \quad \text{and} \quad H(X|P = \text{d}) = 0$$

because the defender knows, or can derive, precise information about the state. As noted, defenses such as ASLR follow this traditional paradigm because, while the attacker

ACM ISBN 978-1-4503-2138-9.
DOI: 10.1145/1235

does not know the placement of pages in memory, so

$$H(X = \text{memory\_location}|P = \text{a}) > 0$$

However, the defender can determine where those pages are in memory by looking at the page tables, so

$$H(X = \text{memory\_location}|P = \text{d}) = 0$$

This model immediately leads to three requirements for a system to implement the joint uncertainty paradigm for a given variable X:

1. **Global availability**: the object that produces the values that X takes on must be accessible to both the attacker and defender.

2. **Stochastic process**: the value that X takes on cannot be deterministic. Otherwise, both the attacker's and the defender's uncertainty will be 0. Thus, these values must be produced by a stochastic process.

3. **Synchronization**: the assigning of a value to X must be done before the attacker accesses the resource that X refers to; thus, a synchronization element is present.

As a very simple example, consider the rotational scheme of Huang and Ghosh for protecting web services [9]. They advocate a set of different software stacks, rotated randomly, to provide a larger attack surface than would one stack. Here, X is the stack to be used. It is accessible to both the attacker and the defender, meeting the first requirement. It is randomly chosen, meeting the second requirement. Finally, the assignment of the particular stack to be used when an attacker requests service (the assigning the value to X) must be done before the attacker accesses the stack, meeting the third requirement.

We now present two examples. The first is a modification to ASLR to add uncertainty for the defender. The second is an implemented digital quorum sensing example to which apply the model to show that, indeed, both the attacker and the defender have uncertainty.

3. EXAMPLE: UNCERTAIN ASLR

Return to our example of ASLR from the introduction. When using ASLR, the network defender is not affected as they have complete information about the system and in particular where the ASLR routine has placed the memory segments; despite the use of randomness, the defender has complete knowledge of memory layout. Thus, if an attacker gains the same access, she can observe what the defender observes, eliminating her uncertainty also.

However, there are a couple of weaknesses to this defense. First, if the adversary is able to force the system to leak information about the memory layout, the adversary can modify their exploit accordingly. Secondly, if the ASLR uses a source of (pseudo)random numbers with low entropy when positioning code and data, the attacker can attempt multiple attacks and from them determine the information needed for a successful exploit. Third, the attacker can largely ignore the address randomization by spraying memory with NOPs, and hope a return will fall in the correct location to pass control to code that the adversary has injected. Ultimately, ASLR is deterministic in nature from the defender's perspective, and may be non-deterministic from the adversary's perspective. If the adversary is able to gain enough information about the system, the adversary's perspective becomes similar enough to that of the defender to enable the adversary to defeat the defense.

To eliminate the defender's knowledge, we posit a "magic memory management unit" (MMMU) that generates random addresses at which to place pages and/or segments of a process' memory. All memory accesses go through the MMMU, so neither the attacker nor the defender can observe memory directly. Thus, all physical memory addresses are inaccessible to everyone. Now the defender has no idea of the mapping between virtual and physical memory addresses; all she can do is supply a virtual address, which the MMMU maps into the physical one, and then performs the desired function (read, write, or something else). The attacker is in the same position as before; the ASLR randomizes the physical layout of the process' memory.

Consider our three requirements. Here, the random variable X represents the physical addresses; for expository purposes, we pick a virtual address (the exact one does not matter), and let X be the corresponding physical address.

1. **Global availability**. The object that produced the value of X is accessible to both the defender and attacker, because both need to be able to access memory. As all such accesses must go through the MMMU, and cannot bypass it to get directly to physical memory, the defender and attacker have equal access.

2. **Stochastic process**. The value that X takes on is random by the definition of the MMMU. Thus, these values are produced by a stochastic process.

3. **Synchronization**. The value of X is assigned before the attacker can access the resource that X refers to, here memory locations. Clearly neither the attacker nor defender can reference process memory before the process is instantiated by loading it into memory. Thus, this requirement is met.

A key observation for the successful use of the MMMU is that, while the entropy of the random number generator is important, equally critical is the entropy of the *random addresses*. So an attacker observing the inputs to the MMMU might be able to deduce information about the memory layout. To prevent this, we suggest an approach that uses trial and error.

Consider the process making a function call. The process does not know the actual function address. Instead, at compile time, the compiler wrapped the function call with trampoline code that will enable the process to try a set of addresses, and the function call will fail only when none of the addresses are correct. When the process begins, the MMMU dynamically provides a set of n possible addresses for each function call, where n is chosen as a trade-off between speed and robustness). The MMMU keeps track of these values and the association between them and the address of the function.

When the process calls the function, the MMMU checks that the provided address is one of the addresses it gave that process. If not, it immediately signals the operating system that a suspicious event has occurred, and identifies the process; the MMMU then returns failure.

If the address is one it gave the process, the MMMU checks to see if that address maps to the function. If so, it notes that

it has entered the function at address a. Once the function completes (that is, a return or other termination occurs), the MMMU randomly generated n new addresses, replaces the previous n addresses in the process, and updates its set of addresses. But if the address is not the correct one, the MMMU rejects it; the trampoline code in the process traps the error and tries the next address in the set of n.

The point here is that neither the attacker nor the defender know which of the n addresses invokes the function. Further, as a new set of n addresses are invoked each time the function call succeeds, knowing that an address was the right one does not enable either the attacker or the defender to use that address again. Hence the address of the function is still unknown.

The MMMU is a hypothetical component that replaces traditional MMUs. It would provide uncertainty for both attacker and defender, and so is a "thought experiment" introducing our idea to provide extra security in the sense that, even if the attacker knows everything the defender does.

We now move on to a non-thought experiment.

4. EXPERIMENT: QUORUM SENSING

A single, lone pathogenic bacterial cell poses little threat to a host. The host's immune system can react and remove it with little difficulty. If a pathogenic cell switches between multiple strategies to minimize its interactions when it is alone or isolated, and replicates as resources become available, a host's immune system may not identify these cells as a threat and allow the bacteria to form a colony. Once a colony has formed, the pathogenic bacterial cells can switch their strategies and attack the host, and at that point the host's immune system may no longer be able to clear the pathogenic bacterial cells. Indeed, some social strategies are more beneficial if the cell is isolated or alone while other social strategies are more beneficial if the cell is part of a community. The start of bacterial bioluminescence, the production of biofilms [2, 7, 19, 22], the release of virulence factors [19, 22], extracellular uptake of DNA through bacterial competence [19], and creation of persister cells [14, 15, 17, 26] are examples of behaviors that are typically more beneficial if the bacterial cell is participating in a community than if the bacterial cell is behaving asocially. The bacterial cells require a communication channel which allows them to determine where there are sufficient cells present to switch their behaviors.

This led to the question of whether malware could take advantage of a communication channel similar to that used by bacteria. A study of this communication mechanism, called Digital Quorum Sensing (DQS), examined whether it was viable [6]. A central requirement of the communication scheme was that it only pass the minimal amount of information to allow the targeted malware to conduct its mission. The study focused on targeted malware that was created by a determined adversary [20] to disrupt a mission, rather than malware designed to infiltrate a network and exfiltrate information. Targeted malware is most likely only concerned with the overall infection state of the network, specifically whether the network is sufficiently infected for the malware to conduct its mission.

The study used a statistical covert timing channel [1, 11–13, 18, 30] to communicate the overall infection state of the network by modifying the properties of a shared resource, in this case the time distribution of network traffic. The communication would be a signal which results in the slight skewing the packet transmission times, but in such a way that those times were not skewed enough to impact enterprise network services or (legitimate) communications. The communication channel is probabilistic in nature as it is not always present in the traffic of a single infected host. Thus, a broader approach is needed to observe the signal—inspecting the traffic of the entire network. A network defender monitoring the traffic would have several hypotheses to chose from. Perhaps the packets were not emitted because the host is not transmitting information; perhaps the packets were delayed because the host is performing other computations or because the host is potentially infected with malware. Normal network traffic is not consistent and is often bursty, so the network defender could not automatically deduce that the delay is caused by a host infected by malware.

4.1 Thought Experiment

As an illustration of the proposed communication scheme, consider the following thought experiment. A large number of participants are invited to a cocktail party, hosted by Alice. This party is interesting to Eve who knows some individuals at the party are willing to help (Ian) in the Eve's nefarious schemes, but Eve does not know which individuals these are. Alice is only interested in having certain types of parties, so she hires the services of a Walter, the warden, whose responsibility is to ensure that the correct types of parties are being held. To complicate things, Ian may be traveling to the party as Bob with a fake ID, so even though Walter can check IDs he cannot automatically assume that any one not named Ian isn't up to anything nefarious. Eve is interested in having parties that are not what Alice desires. Eve is also not interested in the particular individuals present at the party. Eve only wants to know when there are enough Ians to manipulate the type of party to one the sponsor does not want (one of the Eve's nefarious schemes).

Eve recognizes that the manipulation will be successful if there are enough Ians present. If a single Ian attempts to modify the type of the party on their own, Walter will quickly identify the offender and remove them from the premises. If there is a large population of Ians, then the Ians can overwhelm Walter and disrupt the party. Eve could devise a scheme in which there are a series of secret handshakes that allow Ians to identify each other. They can then count the number of times they used the secret handshake. This has the disadvantage that if Walter knows about the secret handshake, then Walter can identify the guests present at the party who are Ians by backtracking each and every individual that person has talked to and see if they used the secret handshake. The secret handshake becomes a way that both Eve and Walter can know with confidence if a particular party-goer is willing to assist the Eve or not — and can *both* act accordingly..

Alternatively, Eve could take advantage of something in the party that all party-goers share, for example the background noise, and exploit it as a communication channel. During the course of the party, people will talk, and this talking generates a level of background noise. When someone wants to speak intimately to another at the party, that person typically lowers her voice, so the background noise at the party decreases. Eve could devise a communication channel that works as follows. Divide each hour into 5-minute intervals beginning at the top of the hour. All of the

Ians will either carry on a normal conversation during each 5 minute interval, or do not talk at all during that interval.

If this "periodic pause communication channel" is used when there are only a few Ians at the party, then there may be a slight reduction in the background noise at the party. If there is a more significant presence of Ians, the periodic reduction in the background noise will be more pronounced. Walter who is monitoring the party may notice the periodic reduction in background noise. Walter must then distinguish between some hypotheses. One is that the noise reduction occurred naturally, as a result of the flow of conversation. A second is that the reduction in background noise was maliciously inserted as a form of communication. Even if Walter suspects that something malicious is going on, he must identify each person who was quiet during that period, and for each of these individuals determine whether the delay in their conversation was malicious. Unfortunately, now Walter has another issue that they must consider: "who else was listening?" Not all of the Ians may have been communicating at the time when the quiet intervals occurred.

Eve may decide that they have an advantage but be unhappy that Walter can potentially identify and remove a large number of Ians before the party's type has been changed to a nefarious one. Eve may be willing to tolerate more uncertainty because in this case she cares about the density of Ians, whereas Walter needs to identify who is willing to help. They modify the periodic pauses and introduce an element of chance. The resulting scheme could be as simple as generating a random number between 0 and 9 inclusive at the beginning of the 5 minute interval, and pause conversing when the number generated is 0. This means that roughly 10% of the Ians will pause in the party during any particular 5 minute interval. The resulting reduction in background noise will not be as drastic as before, and there will be an increased chance that the Ians will guess incorrectly that they should converse during that interval, but to Eve guessing wrong is not as bad as Walter guessing wrong. It is also more likely that Walter will not notice that the background noise has changed. Both Ian and Walter must determine if the reduction in background noise is significant enough to act upon. If Walter realizes that something is amiss, Walter now has three different hypotheses to choose from when inspecting individual guests who did not pause during the 5 minute interval. These hypotheses are that the guest is they are a normal (non-Ian) guest, or that the guest is an Ian who opted not to participate during this window. A guest who did converse during the 5-minute interval is either a normal, non-Ian guests whose conversation naturally paused, or an Ian who is actively communicating during this interval. There is no longer a straightforward distinction between Ians and non-Ians.

In an attempt to gain more insight into the motives of the party goers, Walter may attempt to enlist additional recording equipment and hire the requisite support staff. This is an additional expenditure of resources that both Alice and Walter must deal with, but does not cause Eve to modify her strategy. Playing back a video does not offer any new insight, other than increasing the amount of time Walter must spend attempting to discern motives between Ians and non-Ians. Walter is still stuck playing a game to determine why a specific party-goer chose to pause to pick up a drink from a server, leave the party to powder their nose or even take a breath. All of these recordings consume more resources but may not even help Walter make progress in determining if a party goer is really an Ian.

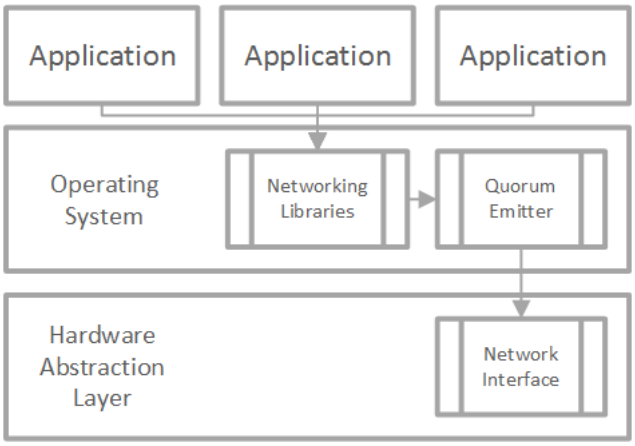

Figure 1: Quorum Emitter into the Operating System Diagram

Analogous to relying upon the background noise at the party, a determined, network-based adversary could rely on the global packet distribution on a network. In effect this would allow the targeted malware several opportunities to evade detection. Assembling a system which relies on the existing network traffic for a communication scheme is not trivial and several different technologies are required to implement such a solution.

4.2 Proof of Concept Study

Now consider a large enterprise network in which most of the hosts are infected. Each host is inserting the covert timing channel into all traffic being emitted from that host. If the signal only has a 10% chance of being inserting into the traffic, then it would be reasonable to expect that about 10% of the infected hosts are emitting the signal at any single point in time. A network defender could monitor the traffic from a specific host and may not notice the signal being emitted from that host as most of the time the signal is not present, and even when present, it has only a slight impact on the packet distribution. If instead of monitoring a single host, the network defender (and the adversary) monitor all traffic passing by the network interface of a host, they would be more likely to detect the signal as it is the union of the individual signals from multiple infected hosts. The signal will be more pronounced as the number of infected hosts increases. The individually infected hosts are not important; only the presence of the community is important. This is similar to bacterial quorum sensing, where pathogenic bacterial cells release autoinducers into the environment, and when a critical concentration threshold is crossed, their behaviors change. With sufficient population social strategies are more productive for survival.

The proof of concept consists of two different components: an emitter and a receptor. The quorum emitter, illustrated in Figure 1, resides on the host and monitors all traffic being passed to the host's network libraries. The quorum emitter injects a delay according to a predefined set of emission probabilities. If it is determined that a packet should be delayed, a transmission delay is inserted which prevents the packet from being delayed by a small mount of time (experimentally

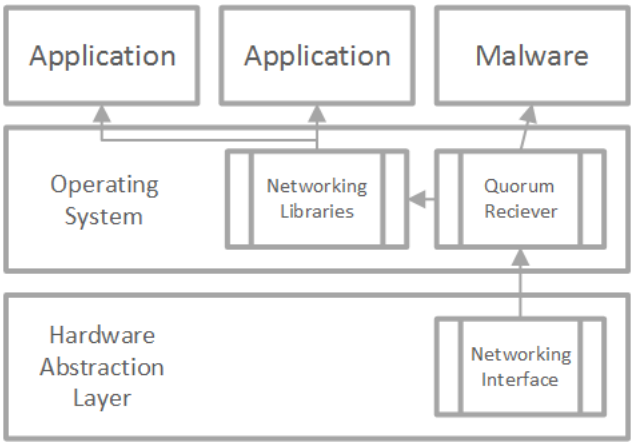

Figure 2: Quourm Receptor Integration into the Operating System Diagram

this delay was on the order of 40 ms). It would be reasonable to assume that an adversary using this covert timing channel would select a profile in which the delays introduced by a specific host are small enough that they can be easily attributed to sources other than insertion of a covert timing channel.

The second component of the DQS system is the receptor (see Figure 2). This component is more complex than the emitter. It relies on Digital Signal Processing (DSP) and performs statistical testing. Although an adversary can select a more traditional command and control scheme, the statistical covert timing channel offers more resilient communications in networks where modifying the contents of packets or using a new protocol will allow the defenders to identify the malware. Like the emitter, the receptor is embedded within or hooked into the host's networking libraries, but unlike the emitter the receptor does not modify the traffic. It only monitors the traffic being received by the host.

In the original experiment [6], the calculations that the receptor performs are more computationally intensive than those of the emitter. When the host receives a packet, the receptor only records the time of arrival. The receptor maintains a count of packets arriving at the host, and it performs a Discrete Fourier Transform (DFT) on the collected time series. The receptor maintains an "ideal" emitter to compare the observed signal against a correlation function. The result of the comparison is recorded.

Follow on experiments in which a revised emitter was tested on a network using existing network traffic revealed in the presence of small amounts of network traffic, the correlation DFTs was insufficient to reliably measure the signal. The revised emitter used the Linux kernel's Traffic Control APIs to induce a periodic notch in which packets were delayed by 20 ms. Now instead of artificially inflating the packet counts to induce a signal, the emitter relied on existing packets to embed a small signal in the traffic. This has the effect of not attempting to mimic an existing protocol which could be detected [8]. A revised detection scheme was developed; in which the packet counts are collected and accumulated over much longer periods (i.e. 6 hours) and the packets were counted in accumulated windows. The counts were analyzed with the Kolmogorov-Smirnov test with the null-hypothesis that the data was parametric in nature. This test is important in that parametric and non-parametric data requires different sets of statistical tests. The results of the tests indicated that the null-hypothesis can be rejected (p-values were consistently $< 10^{-6}$). The next null-hypothesis is that the data collections were all originated from the same distribution. Friedman's test with a significance level of 5% in conjunction with the Bonferroni correction, to determine if the result is statistically significant. The statistical significance of the result varied as the likelihood of delaying packets was changed in the experiment. If the result of the Friedman test returned a significant result, the Wilcoxon rank sum test could be used to determine which periods contained anomalous data. The most interesting results occurred when the probably of delaying packets was set to 10%. In these cases there was a significant result when monitoring the packet counts for the overall network traffic, but if a filter was applied on a per host level each of the tests returned results which indicated a non-significant result. This means that over all a signal could be detected on the network, but if a network defender attempted to narrow down the hosts which were causing the anomaly, no individual host appeared to deviate significantly from the norm.

The adversary and the network defender face similar difficulties in determining if a signal is present. The adversary must statistically test to determine the probability that a signal is present, and this requires that multiple tests be performed. These tests require an additional piece of information, specifically the significance level that is required to reject the null hypothesis. A higher significance level means that the DQS receptor is less likely to trigger on false positives, but the presence of a stronger signal means that network defenders are more likely to notice the signal. A lower significance level means that the receptor is likely to act upon false positives and may prematurely transition, at this point the adversary must weigh the risk of detection against a premature transition to the social state.

During this study, different communication channels were discussed. Of these, the statistical covert timing channel was unique in that it allowed information to be exchanged that was actionable for the adversary but not the network defender. Even if the network defender obtained a sample of the malware and reverse engineered the statistical covert timing channel, they were not able to deterministically identify which hosts were infected by monitoring the network traffic. If the communication channel was identified and monitored, the network defender like the adversary could monitor the traffic to determine the overall infection state. The initial research led us to question if there was a more general class of strategies available in which an adversary relies on strategies that are based on uncertainty.

4.3 Model Application

In order to attempt to identify other situations in which an adversary may attempt to utilize the DQS signaling scheme, a formal model of the communication channel must be developed. The system described in Section 4 was reviewed and the essential characteristics of the prototype were identified. The characteristics identified are those that would allow an adversary to identify resources which could be manipulated in a way that signaling could be performed. By understanding the requirements of the model that the adversary is using

to identify possible communication channels, a network defender could also leverage the same model to attempt to determine where signals may be exist in their network. As a result of understanding the model the network defender has the opportunity to develop ways to not only detect the signal but also develop possible countermeasures.

We first show that the characteristics fit the requirements given in Section 2 for a joint uncertainty approach. We also show that the autoinducer in Section 4.2, which is the network packet counts, are an appropriate random variable.

- **Global Availability:** The autoinducer should be a shared resource which significant number of entities routinely influence. The autoinducer should not be a resource which is only available or accessed by a single entity. Additionally a large number of entities should routinely interface and influence the resource. In the prototype experiment the overall packet counts on the network are observable by hosts on the network (provided network level architectures are not filtering the traffic observed by hosts). This also means that the participating entities should frequently access the autoinducer. If the autoinducer is only infrequently accessed, significant change in behavior on the part of a single entity would be identifiable. For example, if all hosts on a network are relatively quiescent, a host which suddenly contributes a significant about of network traffic would be easily identifiable.

  Further, provided that network traffic is not overly restricted by the network architecture, any host can monitor the traffic received by its NIC. Networks will commonly have broadcast traffic such as Address Resolution Protocol (ARP), Microsoft NetBIOS (if Microsoft Windows clients are present), and possibly multicast-DNS (mDNS). Typically there are a large number of hosts on the same network and all of these hosts are exchanging information periodically. Thus, the characteristics of the autoinducer meets this requirement.

- **Stochastic Process:** The entity should interact with the autoinducer in a way that is that can be analyzed statistically but cannot be predetermined. The process should not be deterministic in nature, for example if the system has a Finite State Machine (FSM) the state transitions cannot be modified as it would violate the FSM's transition table. The adversary could not randomly modify the FSM to cause it to occasionally transition to a state that normally is undeceivable. This is one of the most important features of the model, the adversary will need to contribute minor modifications to the local statistics of the autoinducer. These contributions are small enough on the individual level that they fall well within the normal statistics of the system. When all of these contributions are added, the small deviations are in the same direction so they are cumulative in nature while the remaining unaltered communications are essentially random so they will cancel out when accumulated.

  Although the individual protocols are deterministic in nature, e.g. the TCP protocol relies upon the three-way handshake to start the session, the subsequent number and duration of the messages is not part of the protocol. HTTP is commonly used as a TCP payload HTTP responses depend on the HTTP requests. HTTP is no longer used to just convey static information and the dynamic nature of the content return in the response means that even subsequent identical HTTP requests can receive different HTTP responses. Thus, the autoinducer may be viewed as a stochastic process.

- **Synchronization:** All of the entities in the network are able to maintain clock synchronization such that knowing the times on multiple hosts there are only minor differences. These differences must also be minor over longer time periods. In follow-up experiments, the proof of concept required that all of the hosts maintain a clock drift of less than 20 ms in 6 hours.

  Networks commonly make use of protocols such as NTP to maintain clock synchronization between hosts. Hence the autoinducer meets the synchronization requirement.

Based on this model, a number of different channels can be identified. If a network is configured such that packets are frequently fragmented, instead of delaying a packet by a few milliseconds the autoinducer can rely on periodically fragment a packet before transmission. These channels could be used at the network level and although the communications channel so far discussed focused on the network level, intra-host communication may also be possible.

5. ASYMMETRY & NON-DETERMINISM

Based on our model and analysis of it, our intuition is that there is a role for shared uncertainty as a defense. In the following sections, we examine how non-determinism has played a role in malicious code, and then examine the space more broadly, looking at the problem from a wider perspective.

5.1 History of Non-Determinism in Malware

The battle between attackers and defenders is perhaps clearest in the malware space. Here, defenders have typically been forced to respond to attackers who can probe defenses at will.

A good example of a state where attackers and defenders have very different needs with respect to certainty are "infection markers". The goal of these markers to is prevent a piece of malware from repeatedly infecting the same file. For example, some early malware would mark files as infected by setting the seconds field on the file's timestamp to a particular value, and then check for this value as an infection marker.

Of course, such a marker is not reliable. Many legitimate files (presumably 1 in 60) will be uninfected but have a seconds field in the timestamp that indicates infection. Furthermore, if an infected file is moved and the timestamp changes, some infected files will be marked as uninfected. Regardless of these risks, this strategy was both common and effective for attackers. Using the seconds field of the timestamp for detection purposes is acceptable to the attacker (who simply misses an the opportunity to infect a file) or, on occasion re-infects a file that is already infected. Conversely, detection by timestamp is impractical for the defender (who would suffer an excruciatingly high rate of

false positives). Similarly, infecting an infected file whose timestamp was changed is acceptable to an attacker (in the worst case, either the file is corrupted or the infection is pointless) but marking a file as clean due to its timestamp is not acceptable for the defender. In essence, the attacker and defender have vastly different tolerances for error, making the "soft" signal of the seconds field only advantageous to one party.

Another example of tolerance for uncertainty is found in malware that attempts to obscure its entry point (called Entry Point Obfuscation (EPO) malware [28]). In an EPO infection, the virus sometimes uses a heuristic to determine where to implant the initial entry point of its code. Typically, the malware will attempt to attach itself to a function preamble, in the hope that at some point the function will be called and the malware executed. The technique is somewhat unreliable, but again is "good enough" for the attacker to leverage.

In the case of EPO, the risk taken by the malware author is that the code may not be executed, and hence the infection is inactive. Similarly, the defender may not be able to detect the malware reliably as a fundamental foundation of current-generation virus scanning, emulation, may not execute the actual entry point as it may be either deep within the host code or rely on user input.

5.2 On Asymmetry and Uncertainty

Asymmetry has been discussed in the context of information security and cybersecurity [16, 24]. Liang and Xiangsui have argued that exploiting the asymmetrical nature of cyberspace is essential in the future, especially when participation in symmetric conflicts will guarantee that an adversary cannot win. It has been said that this asymmetry exists in computer security because the adversary only needs to find one weakness, but the defender needs to protect against exploitation of *any* weakness. Pavlovic described this situation as the "fortification principle" [21]. The attack previously discussed in Section 4 is an asymmetric attack because it relies on the asymmetric information requirements of the adversary and the defender.

The effect of uncertainty upon the actions of entities varies. These actions depend on the nature and tolerance to error. Consider an adversary who knows that a high-value target is in one of three buildings. They can destroy one building, two buildings, or all of the buildings. The course of action that the adversary selects will depend on the risk of destroying the wrong building. If the adversary is intent on minimizing the collateral damage, she may opt not to destroy all three buildings at the risk of missing the target. If the adversary does not care about the loss of the other buildings, she can destroy all three buildings to be sure of destroying the target. The uncertainty caused by the lack of information will cause the adversary to act differently based on their tolerance to risk. In the first case, the adversary will prefer deterministic information; she is interested in being certain of the particular structure that the high-value target is in. In the second case, the adversary is not affected by the lack of information; knowing that the high-value target is in one of the three buildings is sufficient to have all three buildings, and therefore the target, destroyed.

"Non-deterministic" attacks take advantage of the different information requirements of the actors involved. The defender needs certainty while the adversary can get by without it – a particular confidence may be enough. For example, an adversary launching an attack on a remote host may be comfortable with an exploit that only works 1% of the time, or even less. The adversary can either launch the attack repeatedly until the host is successfully compromised, or can attempt to exploit a population of hosts large enough that, with very high probability, at least one will be compromised. The network defender is unable to rely on such a probabilistic approach because if even one host is compromised, regardless of how low the probability of such a compromise is, the defense has failed. An enterprise would not be able to function if their software worked only 1% of the time; indeed, even enterprises which guarantee 99.99% uptime still have difficulties when that 0.01% of downtime occurs.

Defensive strategies have attempted to exploit randomness to their advantage. Note that this is different from exploiting bilateral uncertainty. As an example, many modern operating systems use Address Space Layout Randomization (ASLR) in an effort to thwart exploits that rely on fixed addresses. Section 3 described the asymmetry in uncertainty in this defense.

In these situations, the network defender is limited by their need for determinism to operate successfully. The adversary can accept a non-deterministic approach in which the likelihood that their attacks are successful is high enough that they can still accomplish their mission. In other words, the network defender can accept no risk of failure in the defense mechanisms. But the adversary can accept a very high risk of failure of any particular instance of the attack by creating enough instances to ensure that at least one will work.

An interesting observation is that the nature of covert channel exploitation and side channel attacks invert the usual "uncertain attacker, certain defender" paradigm. When two attackers communicate using a covert channel, their goal is to communicate in a way that looks random to the defender but not to the attackers. So here, the attackers have minimal to no uncertainty (depending on the nature of the channel and signaling mechanism), whereas ideally the defenders have maximum uncertainty. Side channel attacks, which are effectively covert channel exploitations where the "sender" is a system rather than an active entity, again has the attacker deriving some degree of determinacy out of what to the defender appears to be non-deterministic.

Along the same lines, the technique of generalization for anonymization and data sanitization [3] adds uncertainty with the goal of preventing the attacker from determining the unsanitized data to a precision sufficient for compromise (where "compromise" is defined from the attacker's point of view). It also may add uncertainty for the defender, those who are generalizing the data. For example, if the sensors gathering the data do so in a coarse manner, the data may arrive at the defenders' site in a generalized form. The defenders then generalize it further, to prevent specific ranges from being revealed. In this case, these meet the paradigm of joint uncertainty.

A method of protecting sensitive information is through the use of negative surveys [4, 5]. A researcher uses a negative survey to query a subject in a way that allows the subject not to expose potentially sensitive information. This work is different in a few respects from our idea. First, the researcher is different from the attacker. The researcher is

willing to use a negative survey that affords the subject opportunities to preserve sensitive information but allows the researcher to potentially gain useful information. On the contrary, the attacker is actively seeking to exploit information from the subject and is willing to generate arbitrary and potentially intrusive queries to get the desired information. In addition, the techniques proposed in this paper are based on both the attacker and defender dealing with situations in which they are not interested in deterministic information (e.g. Eve in the cocktail party from Section 4.1) or have deliberately given up access to deterministic information (e.g. the defender using the MMMU discussed in Section 3). With negative surveys, the subject being queried has complete access to the information in question.

Finally, there has been work on the idea of shared system uncertainty. van Dijk et al. [29] examined a simple security game where players did not know the game state until they played a move. The work spawned research into optimum strategies under mutual uncertainty, and could provide an interesting framework for a theoretical model for the techniques we are using. Schöttle and Böhme [23] have also explored the idea of uncertainty in stereography. Stenography is an interesting topic, but the effect on the channel's value as a result of its discovery are slightly different. Both the techniques described in this paper and communication via steganographic channels rely on modifying existing resources within the environment. But agents using a channel such as the digital quorum sensing channel are able to operate in an environment where the defenders are aware of the channel, whereas the discovery of a steganographic channel defeats the purpose of using steganography. The use of steganographic techniques in conjunction with the techniques discussed in this paper could open some interesting opportunities for the attacker and defender, especially with regards to identifying potential channels that could be used to exchange information.

5.3 The Paradigm, Redux

While we have illustrated our idea in several examples, there is a wider lesson to be learned. In each case, uncertainty as to state played a role — and this uncertainty was shared by both attacker and defender. For example, for self-infection checks in malware, neither side has additional information about the state of a file, but this level of uncertainty penalizes the defender more than the attacker. That is, the asymmetry in the information requirements make the strategy advantageous to one side.

The important difference between our approach is that in our system, attacker and defender share the same level of uncertainty — they differ not in their knowledge, but in their requirements for usable information. We argue that where such an approach can be used, it is more powerful than existing approaches because, quite simply, one cannot leak information one does not have. In ASLR, for example, attackers use probes to slowly gather information about the address space of the target machine. This is possible because the defender actually knows the location of various structures in memory.

We envisage applications of this paradigm where a system is designed to contain only the level of certainty needed to accomplish a particular action and nothing more. For example, consider a GPS system that is embedded in a vehicle. If our driver only needs to know approximate location, but there exists a threat where an attacker needs to know the location at much higher precision (perhaps to detonate a roadside device or target a missile) our existing design paradigms would like to install a regular high precision GPS and attempt to blind the attacker to its output. We argue a better approach is to limit the fundamental accuracy of the location device in the vehicle at the hardware level, so the GPS system, even if fully compromised, provides only enough positioning for the defender, but not the attacker.

Such an approach of deliberate uncertainty should come as no surprise to anyone: it is an approach that has been used in military operations and espionage for as long as their have been wars, where each member of a team may have been given only that information that is critical to their performance of a particular task. We ask why computer security should not work the very same way. In this argument, we are not conflating incorrect information with missing information: both have a role in developing a new model of computer security that is based on the deliberate use of bidirectional uncertainty as a defensive or offensive technique in the cyber realm.

6. CONCLUSIONS

In this paper we have argued that shared uncertainty about a system can be a useful technique for both attacker and defender. We argue that either side can leverage an asymmetry in the information requirements that need to be met in order for a particular mission to fail. We further argue that such uncertainty is preferential, where possible, to defenses that rely on only one party having uncertainty about the state of a particular system.

We illustrate this by exploring an experimental system that allows for limited coordination of malware activity in a way that is meaningful to the attacker, but extremely difficult to leverage for the defender. In such a system, both attacker and defender have the same knowledge: the probability that a certain percentage of machines are infected, but this knowledge has radically different usefulness to each party. We envisage this difference in value to be an interesting area of exploration, ripe with possibilities.

In our arguments, we draw a distinct border between systems that rely on asymmetric knowledge — that is, where either the attacker or defender knows something that the other does not — and systems where both parties potentially have access to the same information. Defenses such as ASLR are quite different from the type of scenario we have considered. Instead, where there is uncertainty in a signal that cannot be overcome by knowledge of a shared secret, differences in the acceptable error rate or tolerance for uncertainty for attacker and defender become the defining factor for utility. Such asymmetric attacks should be explored as a matter of urgency because like it or not, both attacker and defender will use whatever means available to gain an upper hand. In our own work, it seems that the different goals of attacker and defender give the advantage quite firmly to the attacker; more research is needed to see if this can be changed.

An interesting question, not explored in this paper, is the level of uncertainty needed in the attacker's view, and in the defender's view, to make the joint uncertainty model an effective defense. Can one devise a general methodology for deriving the variables, and the minimum entropy associated with those variables, to ensure the defender's information is

useless to the attacker even if it should be revealed? The uncertainty in the attacker is clearly related to that of the defender, and *vice versa*; but what is the dependence? How does all this relate to the specific goals of the attacker, or assets and resources protected by the defender?

In closing, we note that there has been a great deal of focus on techniques that rely on limiting the adversary's knowledge of the system. For example, Moving Target defenses rely on the attacker not knowing the movement of the protected resource (whereas the defender does). We do not see this as a bad thing, and believe such approaches will exist for a long time in the security world as they can provide significant protection. However, we ask if there is a larger potential role for more fundamental uncertainty in the attacker/defender playbook. In the real world, these kind of asymmetries are common. In the digital world, however, the use of these approaches is much scarcer.

7. ACKNOWLEDGMENTS

The authors would like to thank Dr. Marco Carvalho, Evan Stoner and Troy Toggweiler from the Harris Institute for Assured Information, at the Florida Institute of Technology for allowed the use of network resources for conducting experiments. The authors would also like to thank Dr. Heather Crawford from the Harris Institute for Assured Information, at the Florida Institute of Technology for her contribution in revising the statistical tests that were required. Lastly the authors would like to thank Keith Johnson, from the Department of Computer Sciences, at the Florida Institute of Technology for providing access to additional network resources.

This material is based in part upon work supported by the National Science Foundation under Grants Number DUE-1344369 and DGE-1303211. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

8. REFERENCES

[1] S. Cabuk, C. E. Brodley, and C. Shields. Ip covert timing channels: design and detection. In *Proceedings of the 11th ACM conference on Computer and communications security*, pages 178–187. ACM, 2004.

[2] J. Costerton, P. S. Stewart, and E. Greenberg. Bacterial biofilms: a common cause of persistent infections. *Science*, 284(5418):1318–1322, 1999.

[3] R. Crawford, M. Bishop, B. Bhumiratana, L. Clark, and K. Levitt. Sanitization models and their limitations. In *Proceedings of the 2006 New Security Paradigms Workshop*, NPSW '06, pages 41–56, New York, NY, USA, Sep. 2006. ACM.

[4] F. Esponda and V. M. Guerrero. Surveys with negative questions for sensitive items. *Statistics & Probability Letters*, 79(24):2456–2461, 2009.

[5] F. Esponda, K. Huerta, and V. M. Guerrero. A statistical approach to provide individualized privacy for surveys. *PloS one*, 11(1):e0147314, 2016.

[6] M. E. Fioravanti and R. Ford. Bacterial quorum sensing for coordination of targeted malware. In *Malicious and Unwanted Software: The Americas (MALWARE), 2014 9th International Conference on*, pages 101–108. IEEE, 2014.

[7] B. K. Hammer and B. L. Bassler. Quorum sensing controls biofilm formation in vibrio cholerae. *Molecular microbiology*, 50(1):101–104, 2003.

[8] A. Houmansadr, C. Brubaker, and V. Shmatikov. The parrot is dead: Observing unobservable network communications. In *Security and Privacy (SP), 2013 IEEE Symposium on*, pages 65–79. IEEE, 2013.

[9] Y. Huang and A. K. Ghosh. Introducing diversity and uncertainty to create moving attack surfaces for web services. In *Moving Target Defense: Creating Asymmetric Uncertainty for Cyber Threats*, volume 54 of *Advances in Information Security*, pages 131–151, New York, NY, USA, 2011. Springer.

[10] S. Jajodia, A. K. Ghosh, V. Swarup, C. Wang, and X. S. Wang. *Moving target defense: creating asymmetric uncertainty for cyber threats*, volume 54. Springer Science & Business Media, 2011.

[11] M. H. Kang, I. S. Moskowitz, and S. Chincheck. The pump: A decade of covert fun. In *Computer Security Applications Conference, 21st Annual*, pages 7–pp. IEEE, 2005.

[12] M. H. Kang, I. S. Moskowitz, and D. C. Lee. A network pump. *Software Engineering, IEEE Transactions on*, 22(5):329–338, 1996.

[13] B. W. Lampson. A note on the confinement problem. *Communications of the ACM*, 16(10):613–615, 1973.

[14] K. Lewis. Persister cells, dormancy and infectious disease. *Nature Reviews Microbiology*, 5(1):48–56, 2007.

[15] K. Lewis. Persister cells. *Annual review of microbiology*, 64:357–372, 2010.

[16] Q. Liang and W. Xiangsui. *Unrestricted warfare*. PLA Literature and Arts Publishing House Beijing, 1999.

[17] N. Möker, C. R. Dean, and J. Tao. Pseudomonas aeruginosa increases formation of multidrug-tolerant persister cells in response to quorum-sensing signaling molecules. *Journal of bacteriology*, 192(7):1946–1955, 2010.

[18] I. S. Moskowitz and M. H. Kang. Discussion of a statistical channel. In *Proceedings of IEEE-IMS Workshop on Information Theory and Statistics, Alexandria, VA*. Citeseer, 1994.

[19] W.-L. Ng and B. L. Bassler. Bacterial quorum-sensing network architectures. *Annual review of genetics*, 43:197–222, 2009.

[20] M. Oram. Determined Adversaries and Targeted Attacks. Technical report, Microsoft Corp., Redmond, WA, USA, June 2012.

[21] D. Pavlovic. Gaming security by obscurity. In *Proceedings of the 2011 workshop on New security paradigms workshop*, pages 125–140. ACM, 2011.

[22] S. T. Rutherford and B. L. Bassler. Bacterial quorum sensing: its role in virulence and possibilities for its control. *Cold Spring Harbor Perspectives in Medicine*, 2(11):a012427, 2012.

[23] P. Schöttle and R. Böhme. Game theory and adaptive steganography. *IEEE Transactions on Information Forensics and Security*, 11(4):760–773, 2016.

[24] W. Schwartau. Asymmetrical adversaries. *Orbis*, 44(2):197–205, 2000.

[25] H. Shacham, M. Page, B. Pfaff, E.-J. Goh,

N. Modadugu, and D. Boneh. On the effectiveness of address-space randomization. In *Proceedings of the 11th ACM conference on Computer and communications security*, pages 298–307. ACM, 2004.

[26] D. Shah, Z. Zhang, A. B. Khodursky, N. Kaldalu, K. Kurg, and K. Lewis. Persisters: a distinct physiological state of e. coli. *Bmc Microbiology*, 6(1):53, 2006.

[27] B. Spengler. Pax: The guaranteed end of arbitrary code execution. *G-Con2: Mexico City, Mexico*, 2003.

[28] P. Szor. *The art of computer virus research and defense*. Pearson Education, 2005.

[29] M. van Dijk, A. Juels, A. Oprea, and R. L. Rivest. Flipit: The game of "stealthy takeover". Cryptology ePrint Archive, Report 2012/103, 2012.

[30] L. Yao, X. Zi, L. Pan, and J. Li. A study of on/off timing channel based on packet delay distribution. *Computers & Security*, 28(8):785–794, 2009.