Tracking Darkports for Network Defense

David Whyte[*] P.C. van Oorschot[*] Evangelos Kranakis[*]
School of Computer Science, Carleton University

Abstract

We exploit for defensive purposes the concept of *darkports* – the unused ports on active systems. We are particularly interested in such ports which transition to become active (i.e., become *trans-darkports*). Darkports are identified by passively observing and characterizing the connectivity behavior of internal hosts in a network as they respond to both legitimate connection attempts and scanning attempts. Darkports can be used to detect sophisticated scanning activity, enable fine-grained automated defense against automated malware attacks, and detect real-time changes in a network that may indicate a successful compromise. We show, in a direct comparison with Snort, that darkports offer a better scanning detection capability with fewer false positives and negatives. Our results also show that the network awareness gained by the use of darkports enables active response options to be safely focused exclusively on those systems that directly threaten the network. Finally, our evaluation of darkports using three different network datasets illustrates that they are scalable and offer the ability to rapidly characterize and group hosts in a network into different *exposure profiles* that can be used to detect successful compromises or unauthorized network activity.

1 Introduction

The Internet is saturated with "nonproductive" network traffic that includes an estimated 25 billion global intrusion attempts per day [22, 39]. A precursor to most of these intrusion attempts involves some form of reconnaissance activity to identify vulnerable systems or to determine the best point of access into a target network. Automated tools methodically probe large blocks of Internet address space seeking vulnerable systems for recruitment into botnets [9, 25, 6, 24, 2]. Large numbers of worm-infected systems randomly scan the Internet searching for susceptible systems to exploit. Perhaps most worrisome for a network operator is when a determined adversary directs specific scanning activity solely against their network searching for weaknesses to provide them with an entry vector. This type of reconnaissance is typically precise, deliberate, and focused.

A variety of complex heuristics have been successfully developed to detect scanning activity including the observation of connection failures [12, 29], statistical measures [13, 31], abnormal network behaviors [34, 37, 7], and connections to network darkspace [8, 19]. Current scanning detection algorithms focus largely on observing and classifying external network behavior (i.e. incoming network connection attempts) to detect scanning systems, although many types of sophisticated scanning techniques (e.g. botnet scanning, slow scanning) make it difficult or impossible to accurately determine root-cause origins of scanning activity.

In contrast, exposure maps [35] were proposed to detect scanning activity by passively observing legitimate traffic and attack scans (active scanning) directed at a target, and especially observing how internal hosts *respond* to external connection attempts. Preliminary investigation suggested they were suitable for detecting sophisticated scanning activity directed at an enterprise network, with greater interest in what an adversary is searching for, than in who is scanning the network. Successful connections to internal systems would be characterized in exposure maps to define the currently active external interface to the network. In contrast to remote network security auditing techniques (e.g. Nmap [10]), exposure maps were asserted to facilitate an efficient, low-effort method to identify network vulnerabilities, with exposure status continually updated.

In this paper, we pursue these ideas and introduce *darkports* which we define as unused ports (i.e. ports with no service responding) located on active hosts. Darkports provide a method to detect in real-time unauthorized

[*]Contact author: dlwhyte@scs.carleton.ca. Version: 03 February, 2007.

service offerings from a host; these may indicate a successful compromise (e.g. a darkport suddenly starts to respond to a connection request). We propose a number of applications of exposure maps, and discuss our results and experiences of using exposure maps and darkports in three different network environments that range in size from less than a hundred to hundreds of thousands of users: (1) a lab network with a well-defined security policy and small user population, (2) a university network with a medium-sized user population (i.e. a thousand users), and (3) a backbone network. Our experiments are designed to explore the different applications of exposure maps.

First, we validate a preliminary assertion from our position paper [35] that exposure maps are very effective at detecting both simple and sophisticated TCP scanning activity directed at an enterprise network. In a direct comparison, our scanning detection capability was significantly better than the well-known Snort [28]. Exposure maps exhibited both lower false negative and positive rates during our evaluation, and provided the ability to detect additional sophisticated scanning activity directed at the network.

Secondly, we show that the identification of darkports during the construction of the exposure maps provides network-centric knowledge enabling fine-grained automated responses, e.g. to identify and deny specific systems network access when they are found to be performing scanning activity and thereafter trying to access a legitimate service on a host in the network (common behavior for autorooters and worms [23]). This introduces the ability of selective automated response: a focused real-time active response option that limits the introduction of new access control rules to deny those scanning systems directly threatening network assets (i.e. those targeting actual services offered by the network). We emphasize the subtle point, that systems that scan for services not offered by the network are simply identified (i.e. scan attempt recorded) but otherwise ignored (e.g. no access control rule introduced to block the associated source IP address). This ability to initiate selective automated response reduces network configuration changes, complexity errors (e.g. by avoiding a dramatic increase in router/firewall rules, and possibly leading to a self-imposed denial of service), and avoids unnecessary performance degradation of network security devices [4, 38].

Thirdly, we illustrate how exposure maps may be used on both enterprise and backbone networks to logically classify systems into *exposure profiles* that identify and group systems according to the services they offer. We discuss the practical application of exposure profiles and how they can be used to identify malicious network activity (e.g. botnets and worm outbreaks). The technique requires very little computational overhead and easily scales to large enterprise environments or even backbone networks (see Section 6).

Exposure maps and darkports differ from current scanning detection techniques as they rely on identifying the services offered by the network instead of tracking external connection events. The result is a scanning detection technique in which the utilized system detection state does not grow in proportion to the amount and fluctuation of external network traffic, but rather increases only with the number of services offered by the network, regardless of the size of the network and the external network activity. This obviates the need for shrinking time windows or timeouts to accommodate increases or bursts in network traffic, allowing scan detection with a footprint of a single packet or a frequency of hours or days between probes. As an added benefit, maintaining information about internal hosts in the network instead of external host activity provides the necessary network-awareness to answer in real-time questions that should be asked after a scan is detected, such as “What information has been revealed as a result of the scan?”, and “Has the network behavior changed?”

The remainder of this paper is organized as follows. Section 2 refines the basic idea of exposure maps and darkports. Section 3 discusses how exposure maps can be used for a variety of security applications. Section 4 describes our implementation, and the evaluation datasets and methodology. Section 5 presents our evaluation results, including a comparison to Snort, and discussion of advanced scanning heuristics. Section 6 discusses the scalability and stability of exposure maps, including resilience to attacks. Section 7 presents further discussion and limitations. Section 8 reviews related work. We conclude in Section 9. Appendix A contains supporting data and analysis of a distributed scan.

2 Exposure Maps and Darkports

We first describe the constituent components of exposure maps, how exposure maps are constructed and how they define the darkports within the network. We focus on exposure maps relative to TCP ports; for UDP see Section 7.

COMPONENT DESCRIPTION. Exposure maps passively identify the services which have been confirmed (through an observed response during a training period) as being offered by the hosts of a given network. TCP packets with the SYN flag set start the three-way connection handshake. When a connection request is sent to a specific destination IP address/port, if a service is bound to that port and the port is listening (open), the target host response is a packet with SYN ACK flag set, to start a session. Listening services, because they respond to connection attempts or incoming packets, leak information to scanners; they typically correspond to the active ports in a network and can be tracked in terms of what we define below as the HEM and the NEM. Once verified as permitted activity, the HEMs and NEM define the authorized access to individual hosts and the network.

More specifically, a *host exposure map* (HEM), associated with a fixed IP address (host), is the set of ports observed responding to external connection attempts within a predefined period. For each active host i in the network, HEM_i is a set of elements each of which begins with the IP address of i, followed by a port number j; there is such an element for each $port_j$ that has responded to a connection attempt within a predefined period. In symbols, we can abbreviate this as $HEM_i = \left\{ IP_i : port_j \;\middle|\; port_j \ was \ observed \ responding \right\}$.

The HEM is the externally visible interface of a host and can be considered to represent information leakage from the host that may reveal characteristics that can be used to exploit it. Subsequent to the training period, as additional ports respond to external connection attempts, by definition the HEM is augmented by these ports.

The *network exposure map* (NEM) is defined as the collection of HEMs in a given network N at any given point in time. The NEM defines how we expect the network to respond to external connection attempts. In symbols, $NEM_N = \bigcup_{i \in N} HEM_i$. We will often drop the subscript N in NEM_N when the target network is implied by context. This also allows the natural definition of NEM_S for any subnetwork $S \subset N$, i.e. where S is a subset of the populated IP addresses in N.

In an implementation, once the NEM has been built, the individual HEMs that comprise it can be checked for compliance with the network security policy. A NEM that complies with the network security policy is called a *vetted NEM*. We assume that any service (IP address/port pair) not compliant with the network security policy will, once detected, either be shutdown, or implicitly becomes part of the security policy. Thus, movement from a NEM to a vetted NEM is always possible.

We define the *darkports* on a given (real) host as those ports that have not been observed offering any services, and thus are not expected to accept external connection requests.[1] The set of darkports for a host is the complement of its HEM. The set of darkports for a network is the union of the darkports on all its populated hosts. For example, a host with a HEM of only three TCP ports 22, 80, and 443 would have $2^{16} - 3$ TCP darkports i.e., all TCP ports excluding these three. If a darkport responds to an external connection attempt, it becomes a *trans-darkport*. This occurs either when a host offers a new service (whether authorized or rogue), or a connection is made to a service that was not accessed during the training period. Either event causes the HEM to expand, and by definition the NEM expands and will differ from the vetted NEM. Once a trans-darkport is detected, this change can be checked against the network security policy so that the vetted NEM can be updated or any unauthorized service can be stopped.

EXPOSURE MAP CONSTRUCTION AND MAINTENANCE. In summary, exposure maps are created by passively observing a target network's responses to incoming connection attempts (both legitimate connections and scanning attempts) over a training period. Every time a host responds to an external TCP connection attempt, the IP address and port of the host offering the service is recorded. During the training period, each host in the network will reveal services that it offers; the corresponding ports are recorded in its HEM. After the training period, the vetted NEM can be used to identify all the active hosts on the network by their representative

[1] Although a connection attempt to any port at a darkspace IP address (no hosts assigned) will not accept a connection attempt, we restrict the term darkport to unused ports on a populated host address.

HEMs. Thereafter during ordinary network operation, passive observation of network packets continues, and for each connection attempt (i.e., each TCP SYN packet) compliance with the vetted NEM is tested in real-time. If the services offered by a host expand beyond the vetted NEM, an alert is generated to provide notification that trans-darkports have been detected; this indicates to the network operator that either the vetted NEM needs to be updated, or some form of unauthorized activity is occurring.

In general, an important consideration for any technique that requires a training period is that any existing malicious activity (e.g. unauthorized services) may become part of the baseline. In our particular case, a HEM can be verified against an existing network security policy to detect any unauthorized service offerings by the host. The required length of the training period will vary with each network environment depending on a number of factors including number of active hosts, network security policy, permitted user applications, and frequency of service usage; see Section 6 for further discussion.

3 Applications of Exposure Maps and Darkports

Exposure maps provide network-centric knowledge sufficient to enable a variety of security applications. Among these, the subsections below discuss scanning detection, automated response, network discovery and asset classification, and large event detection and identification.

3.1 Scanning Detection Using Exposure Maps

MOTIVATION. Panjwani et al. estimate that 50% of attacks against systems are preceeded by some form of network scanning activity [23]. Current scanning detection algorithms are generally designed to identify and classify suspicious network activity as scanning activity using attribution to a particular source or sources. These algorithms are effective at detecting wide-range reconnaissance activities that can be defined as the rapid scanning of large blocks of Internet addresses in the search for a specific service or vulnerability. This is characteristic of autorooters [32] and worm propagation. Autorooters are composite tools that augment basic port scanning functionality by launching an attack as soon as an open port is located on a target system [1]; they are often used for the rapid enrollment of vulnerable systems into botnets of tens or hundreds of thousands of compromised systems [2]. Simple scanning worms propagate by indiscriminately probing the Internet as rapidly as possible to locate and infect vulnerable systems. Scans from autorooters and scanning worms can usually be attributed to the *true* source as the scans themselves are the first stage of the actual exploit attempt (e.g. a response, from the target, to a TCP SYN connection request will start the exploit in the same session).

In contrast to such indiscriminate scanning, skilled adversaries will go to considerable lengths to mask their activities. Numerous sophisticated scanning techniques allow stealthy, focused scanning of a predetermined target (host and/or network); some of these make attribution to the scanning source impractical, rendering most current scanning detection techniques ineffective. The following techniques belong to this category.

Slow scanning activity against a network or host can be spread out over days or weeks. Over time, these scans will simply be lost in the network *noise*, never exceeding scanning detection thresholds (i.e. being outside of the allocated detection system state).

Indirect scanning occurs when an attacker uses one system (or systems) to scan a target and another system to attack the target. This separation defeats attribution attempts. If the scanning activity from the scanning system is detected (e.g. blocked at a network router, or by system administrator intervention), the attacker simply uses another scanning system. A slightly more sophisticated variation uses *throw-away* scanning systems, i.e. previously compromised systems that have little value to an attacker other than being able to provide a disposable platform to perform tasks. Any scanning activity traced back to the source, will be attributed to the owner of the compromised system.

Distributed scanning occurs when multiple systems act in unison using a divide and conquer strategy to scan a network or host of interest. Typically, one system will act as a central node and collect the scanning results from all participating systems. Distributing the scanning activity reduces the scanning footprint from any single system and thus reduces the likelihood of detection. An extreme version of distributed scanning involves an attacker using a botnet to scan a target in a coordinated manner resulting in very stealthy scans. A relatively small botnet of a few thousand systems can be used to scan thousands of ports or hosts with only a single packet sent from each bot (all with unique IP addresses).

USE OF EXPOSURE MAPS FOR SCANNING DETECTION. A vetted NEM is constructed as previously described. A connection attempt to any port-IP combination not present in the vetted NEM (i.e. a darkport or darkspace) is defined as an *atomic scan* event. The 5-tuple (source IP, destination IP, destination port, protocol,[2] timestamp) of any atomic scan events is recorded for further analysis to secondary storage (hard disk) in the scanning activity log file. This approach requires only that the NEM information be maintained in system detection state (not the darkports or external connection requests), thus allowing detection of even very slow or distributed scans, using only a small amount of main memory (see Section 6). In contrast to most scanning detection techniques that rely on the identification and correlation of external connection events to detect scans, we thus do not require strategies like reducing the detection time window in which connection events are tracked or timeouts, to accommodate network traffic fluctuations.

To fully scan all the TCP services on a network of n hosts a scanning tool would need to scan E = $n * 2^{16}$ ports. For instance, in a Class C or /24 network (254 hosts excluding broadcast addresses), $\approx 2^{24}$ unique TCP port/host pairs could be scanned. In practice, often only a subset of available ports is scanned, as attackers try to locate well-known services in the reserved port range (i.e. 0-1023) or backdoor trojan ports listening on ephemeral ports. Let A be the actual number of services scanned in a network, i.e. the number of unique IP/port combinations of all the detected scans. Within A, each atomic scan event can result in one of three possible outcomes: (1) a probe directed against a darkspace address, (2) a probe against a darkport (note: such a host has a HEM), or (3) a probe sent to a host on an active port (an entry in the NEM). Figure 1 shows the general relationship between the potential service ports scanned (E), actual service ports scanned (A), darkports scanned, and the NEM for a network.

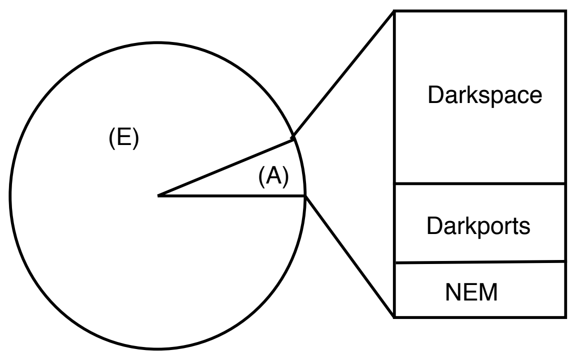


Figure 1: Scanning Potentials versus Network Exposures. (E) denotes potential scans. (A) denotes actual scans.

Unlike most attribution-based scanning detection techniques, the scanning detection approach does not rely on identification of the scanning source to detect scans against a network. Thus, it can detect certain classes of sophisticated scanning techniques that make determining the root cause of the scanning activity impractical. However, this approach does not preclude us from the use of some form of attribution *post* scan detection. Scanning worm propagation and autorooters are two prevalent examples of scanning activities where immediately denying the scanning source access to the network is both relevant and important. In these cases, a successful scan (i.e. one triggering a response from a host) typically leads to an immediate attack from the scanning systems (see Section 5.2). Other post scan detection activities may include the use of heuristics to classify atomic scan events into their respective scanning campaigns. An example of such a heuristic is given in Section 5.1.2 to identify and correlate the atomic scan events that comprise a distributed scan.

3.2 Automated Response using Exposure Maps

Exposure maps can be used in an automated response application as follows. When a new connection request is observed, the destination IP address and port are compared with the vetted NEM to determine if there is a match (see Figure 2). If there is no match to an entry in the NEM, the connection is considered a scan and the source IP address is added as an element in a *scanners* list (implemented e.g., using a hash table). The 5-tuple

[2] Here, the protocol is TCP or UDP.

(as in Section 3.1) that characterizes the connection attempt is then recorded as an atomic scan event in the scanning activity log file.

On the other hand if there is a match, the source IP is checked against the scanners list. If the source IP address matches an entry in the list, the 5-tuple that characterizes this connection attempt is recorded and connection should be dropped as this entity has previously undertaken reconnaissance activity against the network. Our implementation is passive and only produces alerts that could enable some form of containment (e.g. ACL change), but does not actually do the latter; one option would be to integrate this application on a network device capable of performing containment such as a firewall. If the source IP address does not match an entry in the scanners list, the connection is permitted; the entity has no previous history of scanning activity and is connecting to a valid service offered by the network.

The vetted NEM provides context to determine if an incoming connection request is part of a scanning campaign and whether it will likely elicit a response. This information provides us with the precision to limit containment to (e.g., automatically block) only those scanning systems targeting services offered by the network (see Section 5.2). Containment could alternately be performed using a number of network devices including firewalls, routers, or intrusion prevention systems using current scanning detection techniques. However, given the prevalence of scanning activity, frequent dynamic updates to these core network devices would be required in order to stop attacks in real-time, and would pose a number of challenges. For instance, Bobyshev et al. [4] have shown that the size of access control lists (ACLs) and the frequency of dynamic updates can significantly impact router CPU utilization and forwarding capabilities. Furthermore, the addition of multiple blocking rules may make ACLs and configuration files cumbersome and hard to vet by network personnel. In fact, frequent configuration changes to these network devices may actually decrease the overall security posture of the network over time [38]. Our technique allows a precise active response option to be taken exclusively against the most critical known threats to the network namely, those scanning systems targeting services offered by the network. Scanning systems trying to access services not offered by the network are noted (i.e. in the scanning activity log and the scanners list) but no action is needed or taken to block the connection.

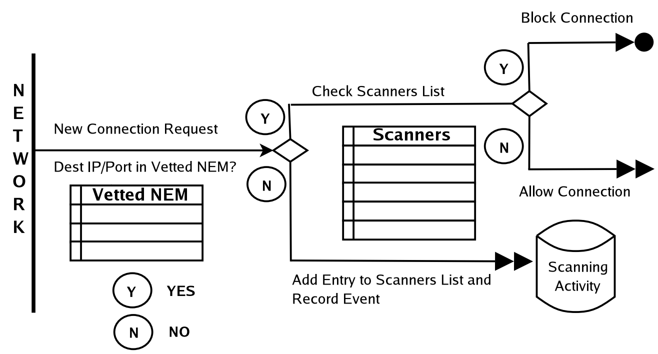


Figure 2: Exposure Map Automated Response Logic.

Our analysis on a four-week network data set reveals a majority of scanning attempts directed against services not offered by the network (see later discussion of Figure 3). In the instances when the scanning was directed against a service offered by the network, an attack always followed (see Table 4 and discussion in Section 5.2). Thus, our approach can significantly reduce the frequency and number of updates to the ACLs of network security devices while providing a measured and robust security response to real-time threats.

3.3 Exposure Profiles: Host Discovery and Asset Classification

In large network environments, it may be useful to discern the number and types of systems within the network which offer services to external entities and logically group them together. Exposure maps provide a mechanism to identify and group hosts that offer similar services into the same *exposure profiles*. As an example, the following four exposure profiles could be generated based on the perceived risk to the network:

1. *Low Risk*: web, DNS, mail, printing, network management.
2. *Medium Risk*: open proxies, P2P services.
3. *High Risk*: known worms, known trojan backdoors.
4. *Unknown*: ephemeral ports that do not correspond to a well known application or service.

In general, the exposure profiles used would vary greatly in terms of the number and types of services in each, depending on the specific network. In our example, the low risk profile includes only well-known *traditional* services offered by the host. The medium risk profile indicates hosts that offer non-malicious but potentially risky services. The high risk profile denotes those systems that offer a service on a port that has known malicious activity associated to it. Finally, the unknown profile contains those hosts that offer services on high order ports that do not correspond to a well-known application or service. Logically grouping hosts by the contents of their HEMs provides a means to rapidly apply some action to a collection of similar systems if required (e.g. deny network access to hosts in the high risk profile to limit potential malicious activity). Furthermore, a change to a host's darkports may move it from one profile to another and necessitate some real-time action be taken on that specific host.

3.4 Large Event Detection and Identification

In contrast to the previous three applications for exposure maps, we now discuss one for which we do not provide any direct experimental evidence in Section 5. In some network environments in which the services offered to external users are tightly controlled, the vetted NEM and darkports will remain static. In other environments, the NEM may change frequently due to a relaxed network security policy that allows hosts in the network to use and offer a variety of services and applications. In the latter situation, it would be common for darkports to become trans-darkports and be reflected in the NEM. Regardless of the type of network, the ordinary NEM maintenance generates alerts are generated when a trans-darkport is detected. These alerts can be viewed by a network operator or alternately processed by some form of software heuristic designed to correlate events or detect "large events". The following scenario shows how trans-darkports (and thus changes to HEMs and the NEM) can provide insight into ongoing large-scale network activity.

Assume the exposure map technique is deployed in a backbone or loosely controlled network, wherein users are permitted to install a variety of applications and services on their system. During the course of monitoring, a number of alerts are generated indicating that the same trans-darkport was detected on a number of systems simultaneously (e.g. port 6667, IRC). Such a change in activity on a multitude of systems within a very short time period indicates some form of coordinated communication has occurred (e.g. responding to command and control sessions as part of a botnet). This method of monitoring trans-darkports to allow real-time network change detection may be best used after logically assigning hosts to exposure profiles as per Section 3.3.

4 Evaluation: Datasets and Methodology

To evaluate how darkport observation can be applied in different network environments, we developed and tested a software implementation. The software is installed on a commodity PC connected to the network by a 10/100 network interface card. The three different network datasets used for testing ranged from a small very secure network to the equivalent of an ISP backbone link.

CCSL DATASET. The CCSL network is a small university research network of 62 Internet reachable addresses connected to a university Internet accessible Class B network. All systems access the Internet through a firewall not permitting inbound connections unless initiated by an internal system. The CCSL dataset consists of four weeks (September 2006) of network traffic collected in pcap files in front of the network firewall.

M2C DATASET. Measuring, Modeling and Cost Allocation (MC2) is a project involving a variety of institutes in the Netherlands that provides a publicly accessible repository of anonymized packet header network trace data [14]. The repository contains header-only network traces from three specific network locations, with source and destination IP addresses anonymized using tcpdpriv [18]. We selected 21 network traces from network location 3, a gigabit link to a Dutch academic and research network for over 1000 students and staff; these network traces compose seven full days of network activity in November 2003.

MAWI DATASET. The WIDE project [36] operates a nation-wide research and development testbed in Japan, interconnected to a number of similar testbeds around the globe. The WIDE working group Measurement and Analysis on the WIDE Internet (MAWI) [16] has a publicly available packet trace repository taken from a number of sampling points. We used traces from samplepoint-F, a 100Mbps trans-Pacific link, and specifically only 15 minute network traces taken in November 2006 at the same time every day over a contiguous one week period; the source and destination IP addresses were anonymized using a modified tcpdpriv [17] (see discussion in Section 7). An average of 235K unique IP addresses are observed in each network trace file.

EVALUATION METHODOLOGY. To test the scalability of various applications of exposure maps, it was important to understand how they would react in large network environments with a diverse user population using a variety of software applications. Accordingly, we evaluated the scalability of the prototype on the M2C and MAWI datasets. These datasets only contain anonymized packet headers; the type of obfuscation used on the source and destination IP addresses is consistent within a single network trace but not across multiple network traces (see Section 7). For such datasets, the concept of training period is not applicable (i.e. the same host communications in different network traces may be mapped to different IP addresses), and the scanning detection application was not evaluated on them. However, the volume of network traffic and diversity of the user population in these datasets makes them ideal to test NEM scalability as well as the logical grouping of hosts into exposure profiles. To complement these tests, we tested scanning detection and selected automated response capabilities on the CCSL network dataset; its network boundaries are known, allowing the NEM to be validated against a known network security policy. Additionally, having access to the full network traces, post scan detection analysis was possible to confirm our experimental results when comparing actual scanning detection capability with Snort.

5 Evaluation Results

We first tested the ability of exposure maps to perform scanning detection by performing a side-by-side comparison with Snort [28]. We then show how exposure maps can be used to detect sophisticated scanning activity; analyze the effectiveness of using the exposure map scanning detection capabilities to perform a real-time fine-grained automatic response to attacks; and validate the network discovery and asset identification feature of exposure profiles in both a medium-sized enterprise and ISP peering point network environment.

5.1 Results: Scanning Detection

As discussed, the CCSL network dataset has a NEM comprised of three HEMs (see Table 1). Two of these have three active ports; the third has one active port. The NEM thus has in total seven port/IP entries.

5.1.1 Scanning Detection Comparison with Snort

We compared scanning detection results with Snort on the CCSL network dataset. We used a one-day training period to construct the NEM; it stabilized within the first 20 hours of network traffic. Snort's preprocessor, sfPortscan [27], performs port scanning detection and allows operations on decoded packets before they are sent on to the Snort detection engine. sfPortscan provides the capability to detect TCP, UDP, and ICMP scanning; its sensitivity is set using the *sense level* parameter (low, medium, or high). We focused on TCP scans at sense level high. Three types of scans were detected by Snort in the CCSL dataset: 1) *portscans* (single host scans multiple ports on a single host); 2) *distributed portscans* (multiple hosts scan multiple ports on a single host); and 3) *portsweeps* (single host scans a single port on multiple hosts).

The implementation detected 740 885 atomic TCP connection events (scans). Figure 3 shows the relationship between legitimate connections attempts and TCP scanning attempts. The upper bound on the possible TCP scanning footprint is $E = 62 * 2^{16}$. The actual scanning footprint we detected was $A = 2\,342$ unique TCP port/IP combinations (including all seven entries in the NEM). With 26 *live* systems in the network, the number of darkports is $DP = 26 * 2^{16} - 7$ (the seven entries in the NEM are excluded). To compare exposure maps with Snort, we applied Snort's scan definitions to group the scans[3] we detected.

[3]Recall that a scan is defined by the NEM as an atomic TCP connection attempt.

Host	TCP Ports	Description
10.0.0.1	25, 631, 993	SMTP/IPP/IMAP
10.0.0.2	22, 80, 443	SSH/HTTP/SSL
10.0.0.3	22	SSH

Table 1: Details about NEM for CCSL network.

	Snort	Exposure Maps
Port Scans	127+1	127
Distributed Port Scans	54+14	54
PortSweeps	7871+42	7871
Other Scans	0	461
False Positives (total)	57	0
False Negatives	461	0
Unique Scanners	322	813

Table 2: Scanning Detection Comparison. "+n" are scans that are false positives.

Snort detects scans by counting RST packets from each perceived target during a predetermined timeout interval [15]. Before declaring a scan, 5 events (i.e. RST packets) are required from a given target within a window. The sliding timeout window varies from 60 to 600 seconds by sensitivity level; at the highest level, an alert will be generated if the 5 events are observed within 600 seconds. Exposure maps do not employ a timeout window; the 5-tuple of atomic scan events are simply recorded and stored, whereafter a number of heuristics can be used to classify the scans detected (see Section 5.1.2). On the other hand, Snort does not require a training period for scanning detection.

Table 2 summarizes the results. Snort detected a total of 8 052 scans initiated by 322 unique scanning systems, while the NEM detected 8 513 scans initiated by 813 unique scanning systems – all of the 8 052 scans detected by Snort, and an additional 461 scans initiated by 461 unique systems not identified by Snort. These are denoted *other scans* in Table 2; they encompass a variety of scanning techniques not included in the sfPortscan scanning definitions, e.g., scans from a single host to a single port on a single host, slow scans with scan intervals of greater than 15 minutes, and a single host scanning multiple ports on multiple hosts. In the next section, we discuss in detail some heuristics used to detect distributed scans. 57 of the scans Snort detected were false positives, the majority caused by legitimate RST packets traversing the network. At the high sense level, a moderate amount of false positives are expected by normal network activity.

FALSE NEGATIVES. We relied on the output of Snort to provide a baseline of the scanning activity within the dataset. As mentioned, we detected all scans identified by Snort, plus another 461 scans which Snort missed. Thus relative to Snort, for this dataset, our analysis for exposure maps (Table 2) revealed no false negatives.

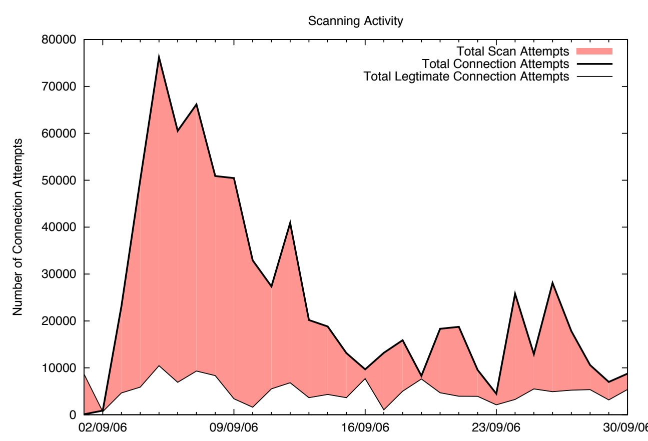


Figure 3: Scanning Activity Directed at the CCSL Network.

Exposure maps (once vetted against the security policy) define the authorized access to the network from external sources. Connections attempts or scans outside these maps are considered a possible scans. Scans directed against a port/IP combination contained in the NEM are not considered a scan but rather a connection attempt to a valid service; this might potentially then be a source of false negatives, and to claim otherwise (i.e. zero false negatives in general) would imply unknowable knowledge of the intent of the party requesting the connection. For instance, a scan to port 443 of host 10.0.0.2 (see Table 1) in the CCSL network would not be recorded as a scan. In practice, although this specific event in a scanning campaign would not be detected, the overall scanning campaign would likely be detected using exposure maps for scan detection, as in most cases we would expect with high probability scans to occur against other hosts in the network not offering SSL (i.e. port 443 darkports). Scanning activity directed solely at the HTTP server would remain undetected and be a source of false negatives. However, we expect that would be an actual attack rather than scanning activity; we do not claim that exposure maps can detect attacks (that are not preceded by scans).

FALSE POSITIVES. Through user error or misconfiguration, a connection attempt might be made to a host or service not offered by the network. In this instance, the intent of the connection attempt was not to scan some portion of the network, but rather it is simply a failed attempt to access a legitimate service. Regardless, this activity would be classified as a scan as an attempt was made to connect to a host/port pair not listed in the NEM. Again, given that there is no way to measure the *intent* of a connection attempt, we must classify these events as scans. While no false positives occurred in our CCSL dataset test (vs. 57 by Snort), we do not claim this in general. False positives will be generated whenever new legitimate services are introduced on the network or services are utilized which were not accessed during the training period (with identification as a trans-darkport until the service has been added to the vetted NEM). We expect trans-darkports to occur infrequently in tightly controlled enterprise environments (e.g. in most government departments, financial, and health care).

5.1.2 Exposure Map Advanced Scanning Detection

Recalling Fig. 2, the scanning detection application identifies connection attempts to darkports within a network, with a 5-tuple extracted from each atomic scan event and recorded in a log file, from which a number of heuristics can be developed to help classify and correlate these events into respective scanning campaigns. Here, we give a few examples of such heuristics to detect distributed scanning attempts.

Attackers may disperse the scanning activity among several sources to reduce the overall scanning footprint in an effort to evade detection. To detect distributed scanning we propose classifying the scan events using the following criteria. 1) *Scanning events and target destination ports*. The number of scanning events per unique source IP address is determined, through analysis of the scanning log, over a configurable time interval (e.g. seconds, days). Similar amounts of scanning events from individual sources are grouped into clusters, which are then grouped by target destination ports. This final comparison reveals scanning systems that share the same scanning frequency (i.e. number of scan packets per unit time) and target service. We consider clusters of three or more scanning sources that target the same destination ports as a distributed scan; the number of systems in a cluster is configurable. 2) *Source IP proximity and target destination ports*. Scanning events are first sorted by unique source IP address. Scanning sources in the same one-quarter class C subnet address range (e.g. /26) are grouped into a cluster. These clusters are then grouped by target destination ports. This reveals scanning systems sharing a similar contiguous address space (which could indicate a single entity *0wns* the scanning systems) and target (i.e. service). Again, we consider clusters of three or more scanning sources that target the same destination ports as a distributed scan.

Using these distributed scan heuristics, we detected three distributed scans in the CCSL network dataset (see Table 3). The first consisted of three source IPs targeting port 80 (HTTP). The scanning campaign was directed against the entire IP address range of the CCSL network (i.e. 62 systems). Once the scanning activity completed, no attacks were detected from these scanning sources. In fact, the only network activity exhibited by the systems participating in the distributed scanning campaign in the network trace was this specific distributed scan. The second distributed scan consisted of 11 systems targeting port 22 (SSH). The scan was also directed at the entire IP address range. Two of the hosts in the CCSL network offer services on port 22. In contrast to the first distributed scan, two of the scanning systems attacked both systems in the CCSL network that offered the service (c.f. Table 1). We describe this scanning activity and attack in greater detail in Appendix A.1. The third distributed scan detected consisted of 9 scanning systems targeting ports 53 (DNS) and 25 (SMTP). Two of the hosts in the network offer services on ports 25 and 53 respectively. Again, all hosts in the CCSL network were scanned with an attack immediately following on the system that offered port 25.

The distributed scanning detection heuristics described above illustrates how atomic scan events detected

# of Scanners	Scanned Ports	# of Hosts Scanned	Follow-on Attack
3	80	62	No
11	22	62	Yes
9	25, 53	62	Yes

Table 3: Three Detected Distributed Scans.

NEM Entry	Scan/Attack Entities	Scans or Attacks
10.0.0.1:25	5	5
10.0.0.1:80	12	18
10.0.0.1:443	3	3
10.0.0.2:22	40	4 545
10.0.0.2:80	17	120
10.0.0.2:443	4	9
10.0.0.3:22	40	10 601

Table 4: Scan Activity as Prelude to Attack.

and recorded through exposure maps can be processed to detect sophisticated scanning activity. Other heuristics may be developed that use the raw output from exposure maps to identify other types of simple or sophisticated scanning activity (e.g. slow scanning). For instance, as an example third heuristic, to detect slow scans to a particular service (i.e. port) one can use the timestamp feature from the recorded scan events. Some time-constrained set of detected atomic scan events is sorted by source IP address. Using the timestamp as a reference, scan intervals of less than 5 minutes from a particular source IP address to the same destination port are ignored. This heuristic would detect scans from a single host to the same destination port on multiple hosts with a scan interval of 5 minutes or greater.

5.2 Results: Active Response

Of the 813 scanners detected by the NEM in the CCSL dataset, 66 launched a total of 15 301 scans intermingled with attacks (unsuccessful) against the network, e.g., repeated attempts to relay mail through the mail server, and attempted logins to an SSH service. Mail relaying is prohibited by our mail server and the responses from the mail server to the attacking system indicate that no relaying occurred; analysis of the network traces also showed that the repeated SSH login attempts were all unsuccessful. Some of these systems scanned and attacked multiple services; this explains why the number of scan/attack entities in Table 4 is 121, while the actual number of unique IPs addresses was 66. With the exception of a single distributed scan (see Table 3), two characteristics of this activity occurred: (1) scanning was always the precursor to the actual attack, and (2) whenever a scan was directed against a service offered by the network (i.e. entry in the NEM), an attack followed once a response to the scan was sent. This “scan then attack” activity fits the profile of autorooter or worm activity as previously described. The attacks were directed against four services offered by the network: SMTP, HTTP, SSL, and SSH.

Without the knowledge of what services are offered and in active use by the network, in a standard perimeter defense all 813 scanning system source IPs over the four week period might need to be blocked at the router or firewall. The NEM provides up-to-date knowledge of the external interface of the network, indicating which minimal set of scanning systems should be blocked. The NEM, coupled with the technique of Section 3.2, would require that only 66 source IP addresses be denied access. This represents a 92% reduction in the number of dynamic updates to the network security ACLs.

5.3 Results: Exposure Profiles

Exposure profiles offer the ability to passively perform host discovery and identification. To determine how well exposure maps can be used to identify and group hosts with similar HEMs into exposure profiles, we tested this feature on three network traces chosen from each of the M2C and MAWI datasets. We classified all the HEMs using the profiles of Section 3.3; specific TCP ports for the types of applications listed in the example profiles are listed in Table 5. These specific profiles were selected to demonstrate the feasibility of using HEMs

Table 5: Exposure Profiles' TCP Port Assignments

Risk Level	Ports
Low	21, 22, 23, 25, 53, 80, 110, 113, 119, 143, 443, 554, 993, 995, 1755, 1863, 5050, 5061, 7000
Medium	1080, 2126, 2128, 3124, 3126-8 3389, 4444, 4660-72, 5555, 6257 6346, 6347, 6348, 6660-9, 6699 6881-9, 7123, 8000, 8080, 5126 5128, 46000, 50500
High	135, 445, 1433, 6969, and other known malware programs.

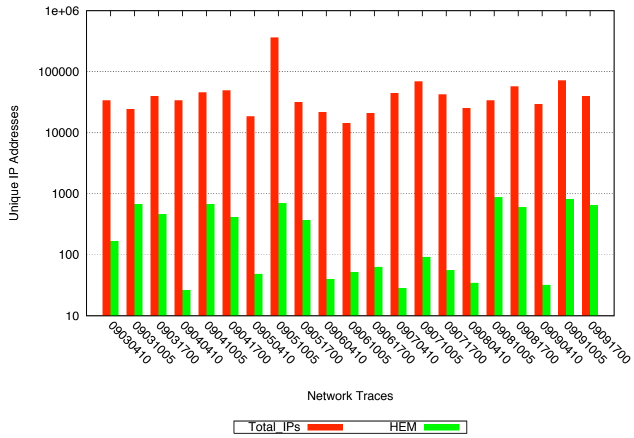
Table 6: Exposure Profiles

MAWI					
Network Trace	# of HEMs	Low Risk	Medium Risk	High Risk	Unknown
200611091400	7 554	6 997	106	184	346 (4.6%)
200611101400	8 006	7 090	147	414	359 (4.5%)
200611111400	5 967	4 091	475	68	434 (7.3%)
M2C					
Network Trace	# of HEMs	Low Risk	Medium Risk	High Risk	Unknown
20030903-0410	167	58	0	108	1 (.59%)
20030903-1005	685	627	4	34	30 (4.3%)
20030903-1700	470	452	1	0	17 (3.6%)

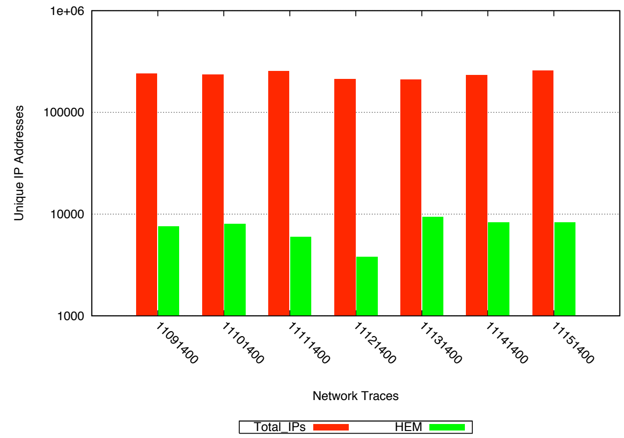
to group their respective hosts into specific network profiles. Accordingly, the number of profiles and specific services included in each are configurable for different network environments.

Exposure profiles can be used to rapidly partition hosts based on their HEM into subsets of the NEM. For instance, Table 6 summarizes the number of HEMs (hosts) within four exposure profiles for the selected network traces from the M2C and MAWI datasets. The Unknown profile refers to HEMs that offer services on ports not listed in one of the other three profiles of Table 5. A HEM's placement in a profile in this example is determined by the *highest* risk service it offers; a HEM that contains entries 22, 80, and 1080 would be included in the Medium risk profile. In none of Table 6 traces were more 8% of hosts unclassified (i.e. left over in the Unknown profile). Inclusion even in this Unknown profile provides valuable information: the corresponding hosts offer external users some unknown service which may contravene the network security policy. A more conservative alternative would be to place the Unknown HEMs into the High risk profile.

Exposure profiles are useful for other applications. An ISP could use exposure profiles in response to global cyber events (e.g. worm outbreak, new exploit, botnet DDoS attack) creating an applicable profile to identify hosts that are at risk or exhibiting signs of a successful compromise. Accordingly, to ensure the efficient use of resources, different monitoring thresholds and security applications could be applied to the subset of hosts within the network based on their exposure profile. For instance, hosts that belong to a server farm exposure profile (e.g. hosts that offer HTTP services in a network enclave) might be afforded a different type or greater level of monitoring than hosts with other exposure profiles. At the most basic level, exposure profiles could be used to simply differentiate between clients and servers in a network, or for network operators simply as a method to discover and baseline the services offered in their network.



(a) Number of HEMs in the M2C network traces.



(b) Number of HEMs in the MAWI network traces.

Figure 4: Exposure Map Variation due to Network Activity (best viewed in color).

6 Scalability and Stability of Exposure Maps

The size of exposure maps will be determined by numerous factors, the two most important being the number of distinct hosts using the monitored link (or network), and the variety of applications those hosts use. We tested exposure maps on the M2C and MAWI network datasets to determine scalability in larger and more diverse environments; individual traces were hundreds of thousands of unique source and destination IP addresses. Table 8 (Appendix A.2) summarizes the number of unique hosts observed in various network traces, the sizes of the corresponding NEM (in terms of number of IP:port entries), and the number of HEMs (i.e. active hosts offering a service). In these traces it turned out that the number of HEMs constructed (and thereafter requiring maintenance and updating) never exceeded 5% of the total number of unique IP addresses. The average HEM size (corresponding to the number of services offered by a single host) was similar for all network traces and surprisingly small: less than 1.2 services on average. Apparently, the majority of hosts (servers) offer only a single service to client systems. Figures 4(a) and 4(b) illustrate how the number of HEMs varies with the number of distinct IP addresses monitored in the network traces (the y-axis is log scale). The smaller the NEM, the less detection system state that needs to be maintained and the greater the scalability.

The resource consumption of exposure maps includes system detection state and disk storage. The former refers to storage for the features extracted from network events that must be maintained at all times in main memory, providing the wireline speed context to build and maintain the exposure maps as well as perform its various applications.[4] A number of techniques are used by other network-based scanning detection approaches to limit their use of allocated system resources (CPU cycles, main memory, disk storage), e.g., connection timeouts, reduction of monitoring windows, fixed sized memory buffers and analyzing only certain events/protocols. The disk storage usage by exposure maps will depend on the detected scanning activity, increasing with the number of atomic scan events recorded. We now discuss the system detection state and disk storage requirements for the various applications of exposure maps, and computational overhead.

SCANNING DETECTION. For scanning detection with exposure maps, (1) the NEM must be constructed and maintained, and (2) atomic scan events must be written to disk in the scanning activity log file. To understand the amount of detection system state for (1), consider the largest NEM observed in Table 8, consisting of 11 207 entries (MAWI dataset). Each NEM entry contains six bytes: four for IP address and 2 for port. The total memory footprint for this NEM is $6 * 11\,207 \approx 67K$ bytes plus additional overhead for storage in a data structure (e.g. hash table, Bloom filter). Thus with an allocation of 100K or 200K in system detection state, we

[4] Depletion of this finite resource, due to traffic volume or an intentional DoS attack, can overload and defeat a detection system. Resilience to attack is discussed separately; see end of Section 6.

could perform scanning detection for this link that contained the traffic of 210K unique systems. As new connection requests are received, a single lookup is performed on the NEM with the destination IP and port fields from the incoming request to determine if there is a match. The small amount of detection system state coupled with the minimal computational overhead required to determine if an incoming connection request matches a port/IP pair in the NEM (i.e. a single lookup) make this technique suitable for use at wire speed even in large enterprise environments.

To estimate the disk storage required for atomic scan events in the scanning event log file, we examined the MAWI and CCSL datasets. Although the scanning detection technique was not applied to the MAWI dataset, we can approximate the amount of scanning activity occurring within the network traces by examining the number of unacknowledged connection requests (i.e. SYN packets); these can occur during normal network activity (e.g. connection timeouts, resets), but also provide a method to estimate the number of connection attempts to an unavailable service (i.e. port with no service listening). The largest number of unacknowledged connection attempts in the MAWI network traces of Table 8 was 190 247. The unoptimized 5-tuple that represents each atomic scan event, in character delimited ASCII, requires 44 bytes of storage. To store one week of scanning activity for the MAWI dataset (a full 24 hours network trace per day) would require 5.7 Gbytes; or for one month, 169Gbytes.[5] In contrast, to store the 780 885 atomic scan events detected in the CCSL dataset (4 week period) in the scanning activity log file would take 33Mbytes. Assuming the dataset represents an average level of scanning activity, an entire year of scanning activity for the CCSL network ($\approx$ 391Mbytes) could be stored on a single CD or USB key.

Long term event storage is useful for applying heuristics to detect sophisticated scanning activity (e.g. slow scanning) and scanning trend analysis. Part of our future work includes optimizing the way that atomic scan events are stored, to significantly reduce the disk storage required e.g., through the use of binary output, scanning event aggregation, and compression.

AUTOMATED RESPONSE. The automated response application is more expensive on system detection state than the scanning detection application due to the scanners list (recall Section 3.2). Each entry in the scanners list requires an additional 4 bytes (plus additional overhead for the hash table data structure). As connection requests are received, an additional lookup is required (i.e. a check against both the NEM and the scanners list) to determine if the source IP address matches an entry in the scanners list. We detected 831 scanners in the CCSL dataset. With an additional allocation of less than 4K in system detection state, we could enable the automated response application. Over time, the scanners list will grow and would have to be managed (reduced) so that some predetermined limit of system detection state was not exceeded (see Section 7).

EXPOSURE PROFILES. The exposure profile application is the least expensive in terms of system resources. To build exposure profiles, we need to construct and maintain the NEM as for the scanning detection application. The HEMs in the NEM are simply sorted and logically grouped by their respective ports into the respective profiles. There is no requirement to write any information to secondary storage.

STABILITY. The stability of a NEM will vary greatly depending on the environment in which it is used. In an enterprise network with a tight network security policy (e.g. government, finance, health care), we would expect the NEM to stabilize quickly and thus be suitable for use in a scanning detection technique. As noted in Section 5.1.1, in our university lab network the NEM stabilized in 20 hours. In other environments, service usage may vary by day of the week. In a network environment with an open network security policy, the NEM may scale but not stabilize as new hosts continually enter and leave the network (e.g. mobile users) and new applications and services are continually added to client systems (e.g. P2P file sharing, open proxies). Furthermore in the core network of an ISP the concept of network boundaries and universal network security policies are not applicable. In these *fluid* network environments, exposure maps remain useful, e.g. as a tool to perform network discovery and asset identification through the application of exposure profiles, as discussed in Sections 3.3 and 5.3.

DOS ATTACKS. A potentially serious attack on many scanning detection mechanisms is one that specifically targets the detection system. In this context, we review the general construction and maintenance of

[5] A commodity external hard disk with this storage capacity costs $\approx$ \$100.

basic exposure maps, plus the three main applications considered (scanning detection, automated response, and exposure profiles).

The construction and maintenance of basic exposure maps appears resilient to DoS attack. Incoming scans (bursts or sustained activity) do not increase an exposure map's size (i.e., the number of HEM entries), which reflects only the number of services offered by the corresponding host. Incoming scans do need to be passively monitored, and connection requests are checked for matches against the NEM; however, the processing required for this is minimal, and we would expect any problems caused by volume of requests to cause other elements of a network to fail, e.g., having adverse affect on core network devices such as routers, or firewalls. Similar to basic exposure maps, the exposure profile application appears resilient, as neither disk storage nor system detection state are adversely affected by attack; exposure profiles rely only on exposure maps to logically group system devices based on the services they offer.

In the scanning detection application, secondary storage may be adversely affected by a large botnet DoS effort, because detected scanning activity is recorded. For example, for a 100,000 system botnet executing a scanning campaign on a target network, three simultaneous scans by each bot would consume 13.2Mbytes in the scanning activity log. A sustained scanning effort by such a botnet would exhaust disk storage in most networks. However, such an attack would also likely cause core network devices to fail as noted above.

The automated response application would experience the same impact on disk storage as the scanning detection application, plus system detection state would be consumed for source IP addresses added to the scanners list (as incoming connection requests to port/IP combinations outside the NEM result in new scanners list entries). A botnet of size 100,000 would consume 400Kbytes of (scanner list) system detection state; this state consumption does not increase after the first scan from each source IP address. The most successful attack would likely be an attacker intentionally trying to exhaust scanner list state by spoofing source IP addresses during a large scanning campaign; this could adversely affect the platform executing the automated response application.

7 Further Discussion and Limitations

INDICATIONS AND WARNINGS. Another application of exposure maps is as follows. As in Section 3.1, each connection attempt to a darkport is considered a scan against the network and this activity is recorded. Over time, the scanning activity detected by exposure maps can be analyzed to determine specific scanning activity patterns or long-term trends. For instance, a sudden burst in scanning activity directed against a service offered by the network may prompt the network operator to confirm the patch level for the software associated with that service. A number of open source security sites could be consulted (e.g. CERT) to determine if the activity may be the result of an emerging exploit or zero-day vulnerability. In the event no suitable explanation is found, the network operator may choose to closely monitor activity to the hosts that offer this service until the scanning activity returns to normal levels.

NON-STANDARD PORTS. One of the strengths of the exposure map approach (and all discussed applications herein) is that it need only maintain very little state when operational. It need not inspect or decode the content of a TCP connection, but only to observe external connection attempts (i.e. SYN packets) and record the IP address and source port if there is a response (SYN-ACK). Thus, exposure maps use port numbers to identify the offered service. Although port numbers are a good indication of the type of service offered, users may choose to install services that use non-standard ports, e.g., an HTTP server using port 8080 or 8000 instead of port 80. Of course, use of non-standard ports may limit access as client systems must know the listening port number before accessing the service. This has the greatest potential impact on exposure profiles, which group systems according to the services they offer; a standard server application using a non-standard port may be misclassified into another profile. In the case of creating a NEM for scanning detection, this issue is less of a concern; non-standard port usage should be detected after training when the NEM is vetted.

MODIFICATIONS FOR UDP EXPOSURE MAPS. Each UDP datagram can be regarded as a discrete event and a potential new communication between hosts. As UDP is connectionless, other measures must be taken to identify and track communications streams between host pairs exchanging packets. For instance, two hosts observed exchanging UDP datagrams could be considered participating in a *session*, with the host that initiates the exchange considered the client. A host that responds (after receiving an initial UDP packet) by sending back a UDP packet is regarded as a server and its corresponding UDP source port is regarded as open. A HEM for the UDP protocol, associated with a fixed IP address, is the set of ports observed responding to an initial incoming UDP packet that signifies the start of a UDP packet exchange.

To implement UDP exposure maps, the first occurrence of an incoming UDP packet with a unique source IP and destination port combination could be kept in state in a sliding time window. The source IP address and source port of the incoming UDP packets are compared with the destination IP address and destination port of outgoing UDP packets. When a match occurs, the UDP packet with the earliest timestamp determines which host initiated the packet exchange and thus is considered the client trying to access a service. Using this method to determine which host in the communication offers the service (vs. responding to a connection attempt in TCP), the definition of NEM, darkports, trans-darkports as well the construction and maintenance of UDP exposure maps is analogous to that of TCP exposure maps.

ANONYMIZED HEADERS. Two of the network datasets used in our evaluation (i.e. the M2C and MAWI traces) contained only packet headers, anonymized using tcpdpriv (see Section 5.1), which scrambles IP addresses to preserve privacy [18, 17] (IP addresses are permuted). It is important that all occurrences of a specific IP address are consistently mapped to a single address within a dataset, to allow meaningful analysis of the network traffic. IP mapping consistency between datasets is also desirable for this reason. Longer consistency is more convenient for analysis but also makes it easier to defeat the anonymization and recover private information [5]. The M2C and MAWI datasets did not maintain consistency between datasets so each network trace must be analyzed separately. Given their short duration, it was not possible to use these datasets as candidates to validate the scanning detection and automated response capabilities of the technique as even a minimal training period was not possible. Furthermore, the traces contained no payload information precluding post-evaluation analysis on the scanning detection results (e.g., false positive and negative analysis).

8 Related Work

The basic idea of exposure maps was introduced in a position paper [35], and developed as an example of an attribution-free scanning detection technique. Preliminary analysis revealed that it could detect both sophisticated and simple forms of network scanning activity. Although not tested, exposure maps were also proposed to detect changes in the services offered in a network which a network operator could verify as either authorized activity or an indication of a successful attack.

Active scanning[6] software, both open source and commercial, allows a security audit on a host or network [10, 33, 26]. Active scanning can be an integral part of a security audit to confirm that a host or network is in compliance with the network security policy. This activity however, can be costly in terms of human resources as it requires personnel to perform the required scanning activity (i.e. configure and operate the software) and interpret the results. Furthermore, active scanning provides only a *snapshot* in time of the active hosts and services in a network. Any new hosts or services offered by the network will only be detected at the next scheduled active scanning session.

Passive scanning techniques continuously monitor the hosts and services available in a network. *Extrusion detection* [3] refers to identification of unauthorized internal network activity by inspecting outbound network traffic; its ultimate goal is the identification of outgoing attack attempts from compromised internal systems in order to stop them from reaching their target. The passive asset detection system (PADS) is signature-based passive detection software with a rules engine to identify hosts and services running on a network by inspecting outbound network traffic [30]. It was created to provide supplementary information when performing active scanning of a network. Snort is an open source IDS that has scanning detection capability [28] through the use of the Snort preprocessor sfPortscan [27], by observing the RST packets within the network for a predetermined

[6] Active scanning involves injecting packets into the network in order to elicit some observable response.

timeout window [15]. If five RST packets are detected from a suspected target within a configurable timeout window, an alert is generated.

Both Leckie et al. [13] and SPICE [31] use probabilistic models to detect scanning activity. These approaches attempt to assign connection probabilities to internal hosts based on the observation within normal network traffic conditions as a benchmark. Scanning systems are detected as they are assigned probabilities based on their current connection behavior which is measured and compared against the a priori connection probabilities assigned to the internal hosts. Jung et al. [12] use their Threshold Random Walk algorithm to identify scanning hosts, based on the observation that scanning systems will contact hosts and ports that are not available more often than benign systems would be.

Network *darkspace* is the unused IP addresses in a network and thus it should have no legitimate network activity directed to it; connection attempts to IP addresses that have no hosts assigned to them is considered anomalous. A number of commercial products (e.g. [19, 8]) make use of network darkspace to detect malicious network activity. A *darknet* is typically a large unused block of Internet-routable darkspace monitored for inbound packet activity. The larger the darknet, the better the darknet's ability to detect scans and attacks during an observation period [21, 20]. A related but subtly different approach by Harrop et al. [11] uses *greynets*, defined as regions of darknet address space that contain some active systems (i.e. some of the IP addresses in the darknet are assigned to active hosts). One of the motivations is that it is not possible for most enterprise network operators to have large regions of contiguous unused address space assigned to them. However, it would be useful to have some means to detect anomalous events if dark space was available on the network. Interspersing valid light (i.e. used) and dark (i.e. unused) addresses throughout a network will make it difficult for malware to avoid targeting greynet addresses and thus avoiding detection.

9 Concluding Remarks

We are the first to exploit the use of exposure maps and introduce the concept of darkports. In contrast to darknets and greynets [11, 8, 19], even densely populated enterprise networks can make use of exposure maps as they exist on live hosts. The overall exposure map technique is based on a simple premise that is efficient to implement – it requires the passive observation, recording, and maintenance of a list of the services offered by the hosts in a network. This simplicity translates into a very efficient use of system detection state and computational resources that easily scales for use in large enterprise and backbone networks.

Exposure maps can be used to perform scanning detection, enable fine-grained automated responses to deny access only to those scanning systems that directly threaten hosts in the network, and identify potentially infected systems through the classification into exposure profiles based on the services they offer. During our evaluation, our implementation of the exposure map scanning detection application had fewer false positives and negatives in a direct side-by-side comparison with Snort. The exposure map scanning detection approach, through the passive recording of all the services offered by the network, provides an *awareness* of active hosts, network darkspace, and darkports allowing network-centric context that increases the fidelity of scanning detection.

In an open network environment, the diversity of user population and permitted activity may make the enforcement of a single comprehensive network security policy impractical. Furthermore, mobile or transient users may make determining a stable baseline of all the services offered by hosts in the network infeasible. In such environments, exposure maps remain flexible enough to be configured to monitor a subset of the network to protect core network assets. A NEM could be composed of a single HEM (e.g., primary web server or for host-based intrusion detection) or several HEMs (e.g., web server farm), allowing a network operator to focus on these mission critical servers. We have developed a full implementation of this approach in software that we plan to make available to the public.

References

[1] G. Bakos and V. Berk. Early detection of Internet worm activity by metering ICMP destination unreachable activity. In *SPIE Conference on Sensors, and Command, Control, Communications and Intelligence*, April 2002.

[2] P. Barford and V. Yegneswaran. An Inside Look at Botnets. *Special Workshop on Malware Detection*, Advances in Information Security, Springer Verlag, 2006.

[3] R. Bejtlich. *Extrusion Detection, Security Monitoring for Internal Intrusions*. Addison Wesley, first edition, 2006.

[4] A. Bobyshev, P. DeMar, and D. Lamore. Effect of Dynamic ACL (Access Control List) Loading on Performance of Cisco Routers. In *Computing in High Energy Physics*, Feb. 2006.

[5] S. Coull, C. Wright, F. Monrose, M. Collins, and M. Reiter. Playing devil's advocate: Inferring sensitive information from anonymized traces. In *To Appear in Proc. of the 14th Annual Network and Distributed System Security Symposium*, Feb. 2007.

[6] D. Dagon, C. Zou, and W. Lee. Modeling botnet propagation using time zones. In *Proceedings of the 13th Annual Network and Distributed System Security Symposium (NDSS'06)*, February 2006.

[7] D. Ellis, J. Aiken, K. Attwood, and S. Tenaglia. A behavioral approach to worm detection. In *Proceedings of The Workshop on Rapid Malcode*, 2003.

[8] Forescout Technologies Inc, Forescout product. http://www.forescout.com/wormscout.html.

[9] F. C. Freiling, T. Holz, and G. Wicherski. Botnet tracking: Exploring a root-cause methodology to prevent distributed denial-of-service attacks. In *ESORICS*, pages 319–335, 2005.

[10] Fyodor. Remote OS detection via TCP/IP stack fingerprinting. *Phrack*, 54, December 1998.

[11] W. Harrop and G. Armitage. Greynets: a definition and evaluation of sparsely populated darknets. In *Proceedings of the 2005 ACM SIGCOMM Workshop on Mining Network Data*, pages 171–172, 2005.

[12] J. Jung, V. Paxson, A. Berger, and H. Balakrishman. Fast portscan detection using sequential hypothesis testing. In *IEEE Symposium on Security and Privacy*, pages 211–225, 2004.

[13] C. Leckie and R. Kotagiri. A probabilistic approach to detecting network scans. In *Eighth IEEE Network Operations and Management Symposium (NOMS 2002)*, pages 359–372, 2002.

[14] M2C Project. (Measuring, Modeling and Cost Allocation for Quality of Service). http://arch.cs.utwente.nl/projects/m2c/.

[15] B. Malmedal. Using netflows for slow portscan detection. Master's thesis, Department of Computer Science and Media Technology, Gjovik University College, 2005.

[16] MAWI Working Group. *Traffic archive*. http://tracer.csl.sony.co.jp/mawi.

[17] MAWI Working Group. Guidelines for Protecting User Privacy in WIDE Traffic Traces. *The WIDE Project*, October 1999.

[18] G. Minshall. Tcpdpriv. http://ita.ee.lbl.gov/html/contrib/tcpdpriv.html.

[19] Mirage Networks. Mirage NAC. http://www.mirage networks.com.

[20] D. Moore. Network telescopes: Tracking denial-of-service attacks and Internet worms around the globe. In *LISA*, 2003.

[21] D. Moore, G. Voelker, and S. Savage. Inferring Internet denial of service activity. In *10th USENIX Security Symposium*, 2001.

[22] R. Pang, V. Yegneswaran, P. Barford, V. Paxson, and L. Peterson. Characteristics of Internet background radiation. In *The Internet Measurement Conference IMC*, 2004.

[23] S. Panjwani, S. Tan, K. Jarrin, and M. Cukier. An Experimental Evaluation to Determine if Port Scans are Precursors to an Attack. In *International Conference on Dependable Systems and Networks*, pages 602–611, July 2005.

[24] R. Pethia. Attacks on the Internet 2003. *Congressional Testimony, Subcommittee on Telecommunications and the Internet*, November 2003.

[25] M. A. Rajab, J. Zarfoss, F. Monrose, and A. Terzis. A multifaceted approach to understanding the botnet

phenomenon. In *Internet Measurement Conference 2006 (IMC'06), Proceedings of*, October 2006.
[26] RemoteScan Corporation, RemoteScan. http://www.remote-scan.com.
[27] D. Roelker, M. Norton, and J. Hewlett. sfPortscan. 2004. http://cvs.snort.org/viewcvs.cgi/snort/doc/README.sfportscan?rev=1.6.
[28] M. Roesch. Snort - lightweight intrusion detection for networks. In *LISA*, 1999.
[29] S. Schechter, J. Jung, and A. Berger. Fast detection of scanning worm infections. In *7th International Symposium on Recent Advances in Intrusion Detection (RAID 2004)*, September 2004.
[30] M. Shelton. Passive Asset Detection System. (PADS). http://passive.sourceforge.net.
[31] S. Staniford, J. Hoagland, and J. McAlerney. Practical automated detection of stealthy portscans. In *7th ACM Conference on Computer and Communications Security*, 2000.
[32] M. Tanase. Introduction to Autorooters: Crackers Working Smarter, not Harder. http://www.securityfocus.com/ infocus/1619.
[33] Tenable Network Security, Inc., Nessus Vulnerability Scanner. http://www.nessus.org.
[34] D. Whyte, E. Kranakis, and P. van Oorschot. DNS-based detection of scanning worms in an enterprise network. In *Proc. of the 12th Annual Network and Distributed System Security Symposium*, Feb. 2005.
[35] D. Whyte, P. van Oorschot, and E. Kranakis. Exposure Maps: Removing Reliance on Attribution during Scanning Detection. Position paper (5 pages). In *USENIX HotSec 2006*, Aug. 2006.
[36] WIDE Project. (Widely Integrated Distributed Environment). http://www.wide.ad.jp/about/forward.html.
[37] M. Williamson. Throttling viruses: Restricting propagation to defeat malicious mobile code. In *18th Annual Computer Security Applications Conference (ACSAC)*, 2002.
[38] A. Wool. A Quantitative Study of Firewall Configuration Errors. *IEEE Computer*, 37(6):62–67, 2004.
[39] V. Yegneswaran, P. Barford, and J. Ullrich. Intrusions: Global characteristics and prevalence. In *SIGMETRICS*, 2003.

A Supporting Material

A.1 Anatomy of a Distributed Scan

In this section, we discuss one of the distributed scans summarized in Table 3. The actual date of the scan was September 9, 2006. Using the *source IP proximity and target destination ports* heuristic (see Section 5.1.2), we detected three (unexpected) distributed scanning campaigns in the CCSL dataset. One of these was executed by a set of 11 scanning systems working in concert targeting the same destination port on all hosts in the target network.

All 62 systems (targets) received a single connection request to port 22 (SSH). The number of atomic scan events from the 11 scanning systems ranged from 2 to 7. The network address space was scanned in non-sequential order and the entire scan lasted 1 second. Table 7 summarizes the characteristics of this distributed scan. Table 7 shows the first 10 scans of the scanning campaign. During the entire campaign, the source IP address varies among the 11 scanning hosts with no occurrence of two atomic scan events arriving one directly after the other from the same source IP. Approximately 2 minutes after the scan concluded, 2 of the systems that participated in the distributed scan attacked (unsuccessfully)[7] precisely the two systems in the network that offered SSH.

Atomic Scan Events
18:01:30 10.0.138.232 > 192.168.1.3.22
18:01:30 10.0.138.237 > 192.168.1.6.22
18:01:30 10.0.138.229 > 192.168.1.9.22
18:01:30 10.0.138.237 > 192.168.1.2.22
18:01:30 10.0.138.229 > 192.168.1.10.22
18:01:30 10.0.138.226 > 192.168.1.14.22
18:01:30 10.0.138.236 > 192.168.1.18.22
18:01:30 10.0.138.230 > 192.168.1.24.22
18:01:30 10.0.138.234 > 192.168.1.16.22
18:01:30 10.0.138.234 > 192.168.1.11.22

Table 7: Distributed Scan Characteristics.

[7] Analysis of the network traces showed that the repeated SSH login attempts were all unsuccessful (see Section 5.2).

A.2 Exposure Maps Constructed From the MAWI and M2C Datasets

Table 8 provides supporting material referenced by the scalability discussion in Section 6.

Table 8: Exposure Maps Constructed From the MAWI and M2C Datasets.

MAWI					
Network Trace	Unique IPs	NEM Size	# of HEMs	Avg HEM Size(services)	File Size GBytes
0611091400	240 377	8 105	7 554	1.073	1
0611101400	235 189	8 628	8 006	1.078	2
0611111400	255 615	6 474	5 967	1.085	4
0611121400	213 340	4 510	3 811	1.183	4
0611131400	210 786	11 207	9 414	1.190	4
0611141400	233 472	8 746	8 251	1.059	4
0611151400	258 144	8 884	8 262	1.075	4
M2C					
Network Trace	Unique IPs	NEM Size	# of HEMs	Avg HEM Size(services)	File Size MBytes
0309030410	33 666	171	167	1.024	84
0309031005	24 323	693	685	1.012	234
0309031700	39 706	482	470	1.026	196
0309040410	33 840	29	26	1.115	51
0309041005	45 785	695	676	1.028	248
0309041700	48 992	441	417	1.058	187
0309050410	18 267	53	49	1.082	58
0309051005	35 950	708	696	1.017	311
0309051700	31 824	391	375	1.043	144
0309060410	21 993	49	40	1.225	81
0309061005	14 541	56	52	1.077	52
0309061700	20 992	71	64	1.109	55
0309070410	44 882	31	28	1.107	56
0309071005	69 354	100	92	1.109	107
0309071700	42 307	62	56	1.107	78
0309080400	25 368	39	35	1.114	47
0309081005	33 972	889	869	1.023	378
0309081700	57 374	613	597	1.027	223
0309090400	29 484	32	32	1.000	47
0309091005	71 293	886	829	1.069	378
0309091700	40 234	686	650	1.055	223