Identifying Network Scanning Tools

Robert Floodeen,
Spectrum Comm Inc.

Kenneth van Wyk,
KRvW Associates, LLC.

Goals

- Today
  - Explanation on a framework to identify network scanning tools
  - Identify potential benefits to determine next steps
- Future
  - Complete analysis algorithm for identification (ODU Ph.D. Candidate in Statistics, Raymond McCullum)
  - Reassess the potential benefits
  - Build and provide the framework to the community

What

- Process Packet Capture and Net Flow Data associated with Network Scanning
- Attempt to Identify (not just detect) Network Scanning tools and techniques used
- Attempt to Catalog the pertinent details of a Specific Scan and it’s associated tool

Why

- Focus our investigative resources and efforts
- Improve our existing Analysis capabilities
  - Added information to existing alerts
  - Potentially generate stand alone alerts
- Improve our ability to classify and analyze the scanning noise
  - Instead of looking at 50 or 1,000 at a time, what if we can review 1,000,000,000?

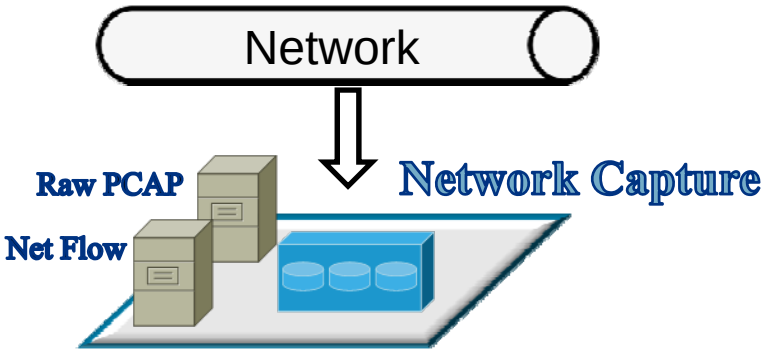
How

- Build a Framework which consists of 4 modules.
  - Provide communication between the modules and external presentation
  - Ensure ease of use to install / remove apps within each module
    - Threshold Count today, tomorrow Threshold Random Walk
    - Our analysis algorithm for identification will not be the only algorithm working in the Identification module

The 4 Modules

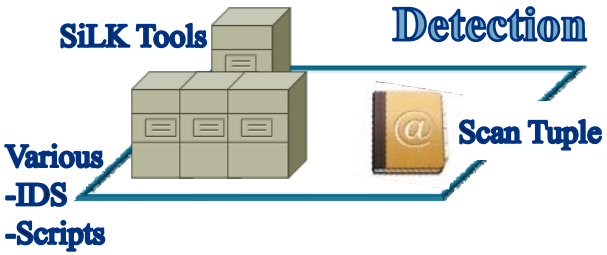
- Network Capture
- Detection
- Identification
- Catalog

Network Capture Module



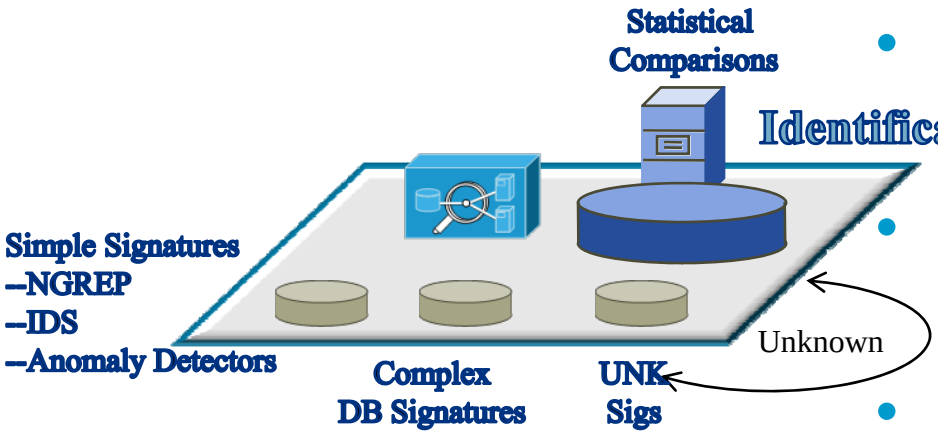
- Simple and Clean
- Capture Network Data

Detection Module



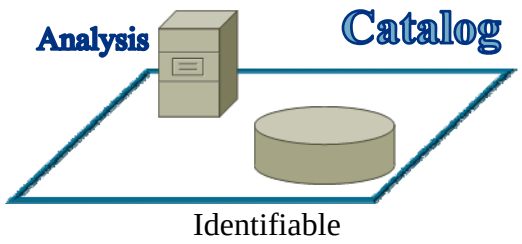
- Detection is advanced enough
- Provide “snap-in” capability for existing Detection Techniques
- Export “tuple” of detected scans

Identification Module



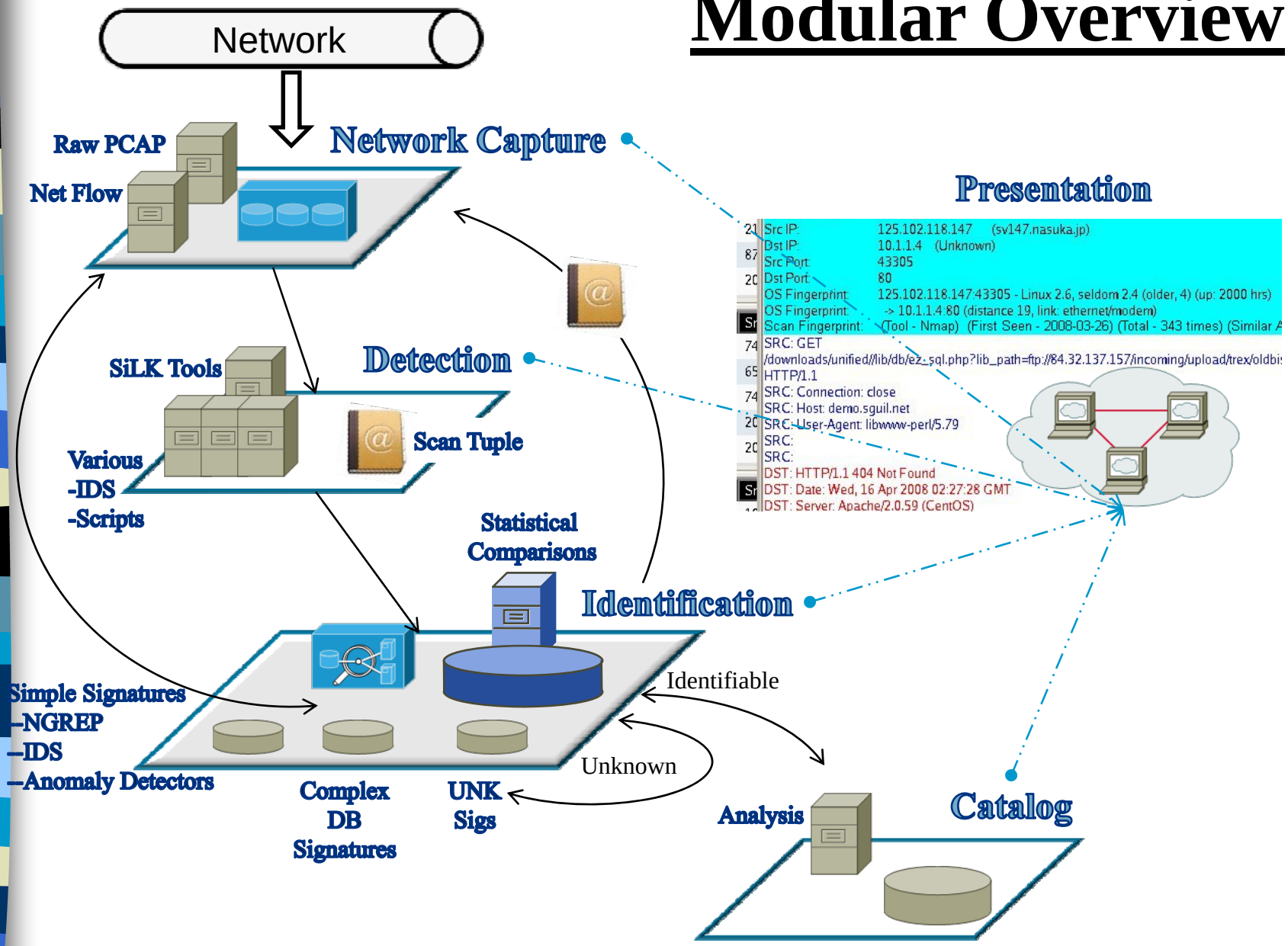
- Simple Signatures
- Complex DB Signatures
- Percentage of closeness to existing signatures for Unknown Detects
- Future work – Build the Statistical Engine

Catalog Module

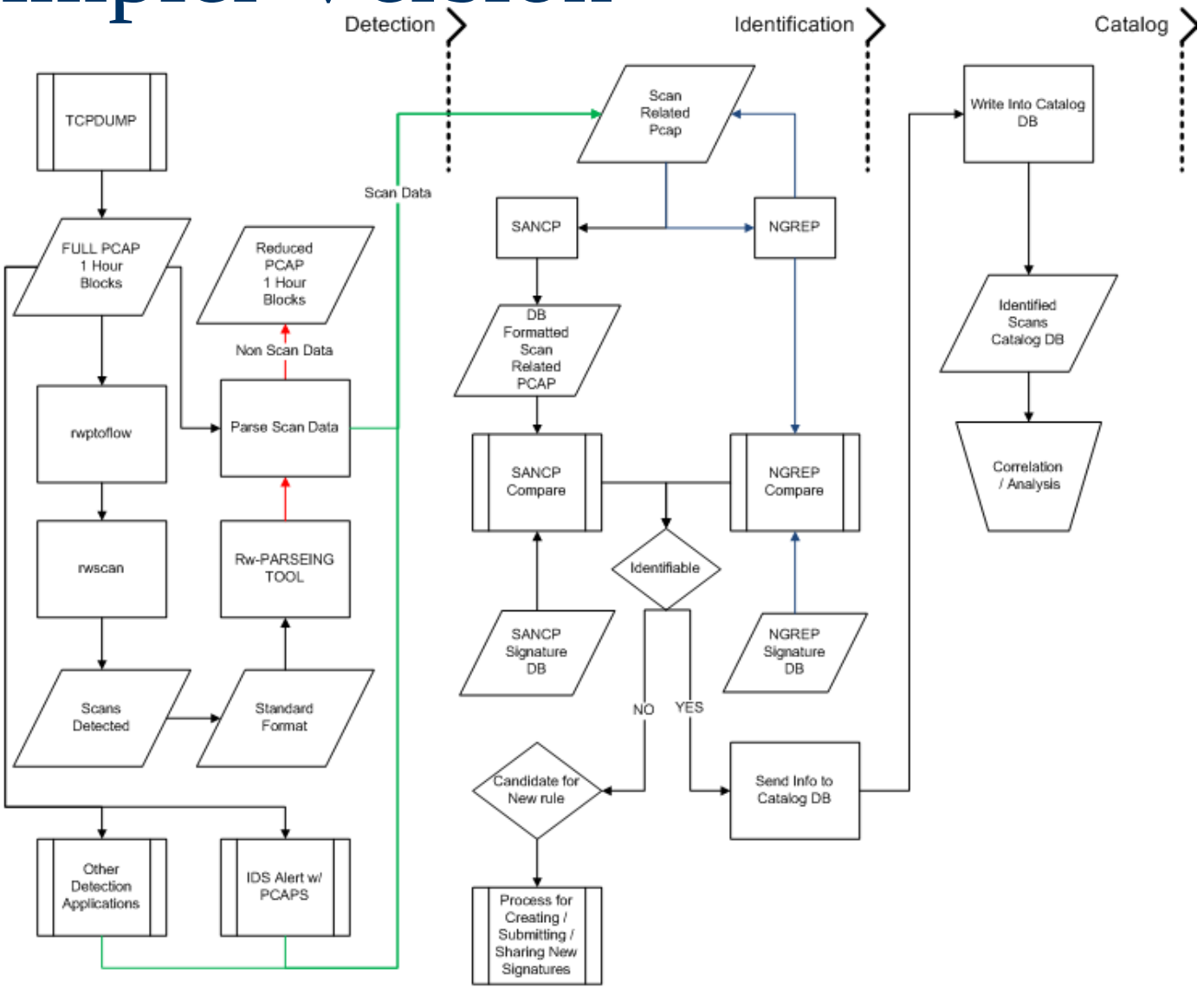


- Store Generalized Tool Information
- Store Specific Identified Instances
- Provide Analysis Capability and Handle Analysis Requests

Modular Overview



Simpler Version



System Benefits

- Full pcap data reduction
- Log Data Reduction
- Free up Intrusion Detection systems from looking for scans
- Pare down false positives

Analysis Benefits

- Finger Printing an Attacker
- Expanding the scope of an incident
- Knowing what we don’t know

Analysis Benefits

- Directing Resources in a triage method
- Information gained about your system (what are they after?)
- Reviewing previously unidentified scans

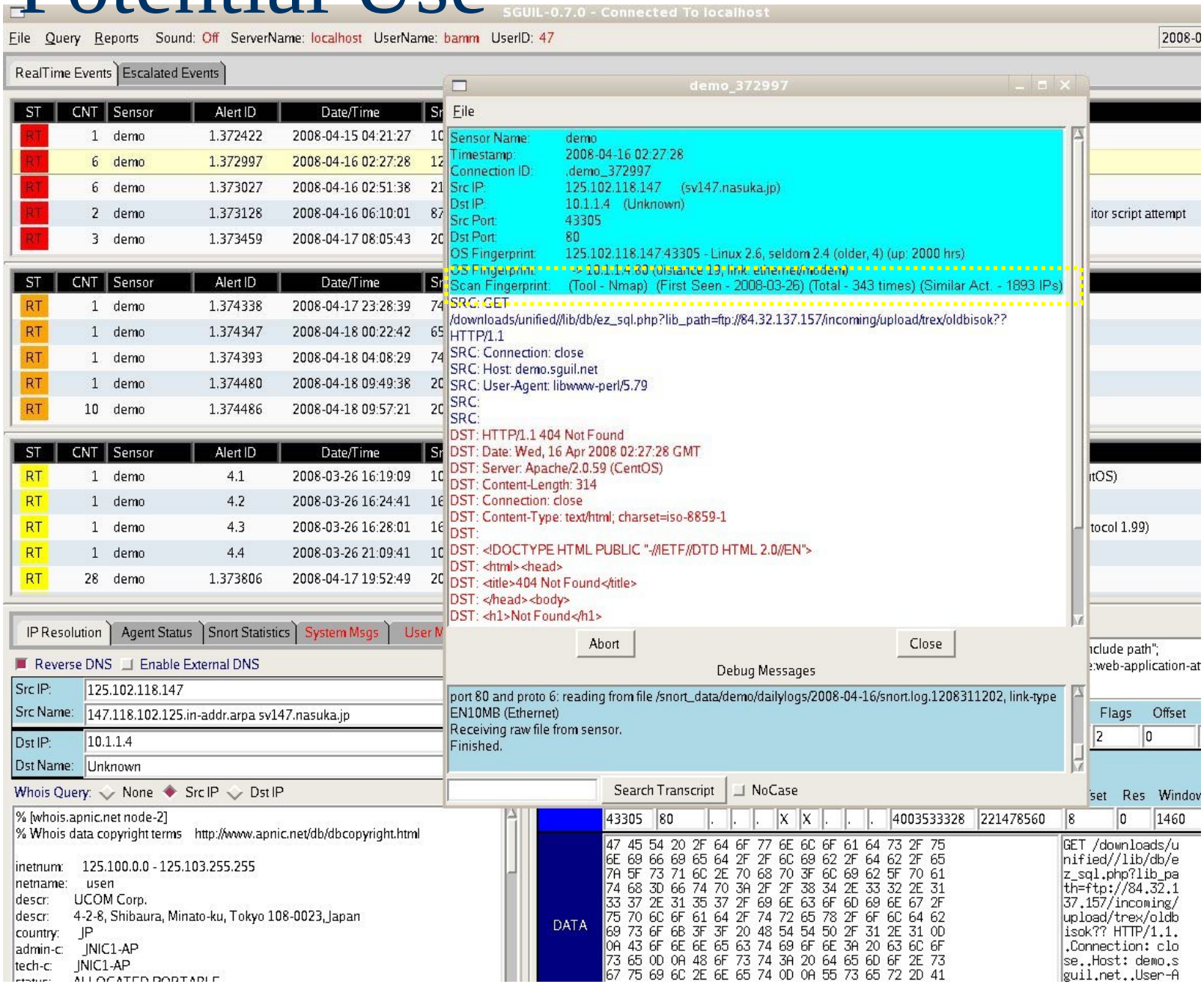
Metrics

- Detected scan to attack ratio
- Identified scan to attack ratio
- Attack to non-detected scan ratio
- Known to an unknown scan ratio

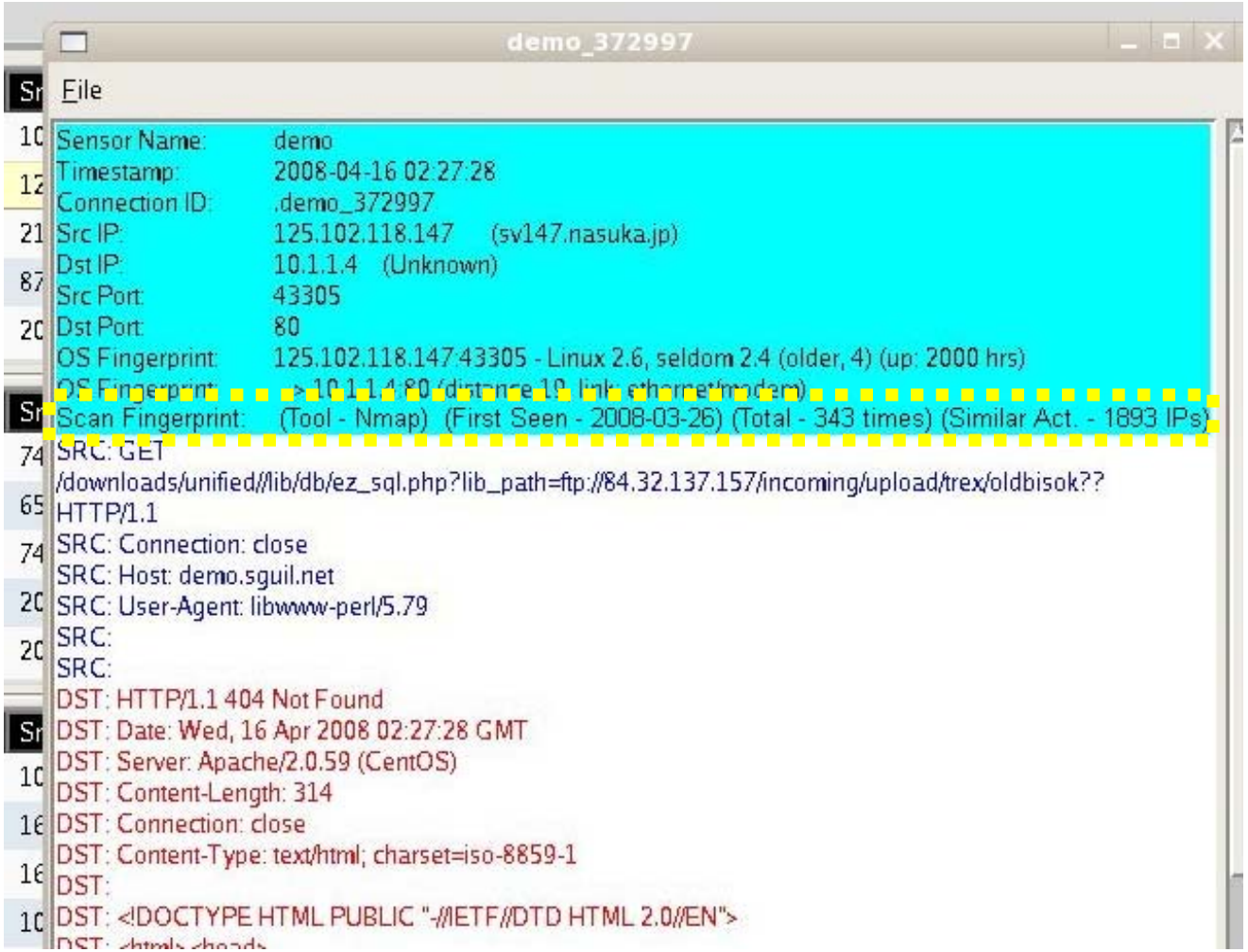
Metrics

- By
  - tool
  - classification of tool
  - target
  - classification of target
  - network block
  - size of scan (packets/bytes in/out)
  - Detected/Identified/Unique over a series
- During a specific time of day
- Associated with a new vulnerability over time

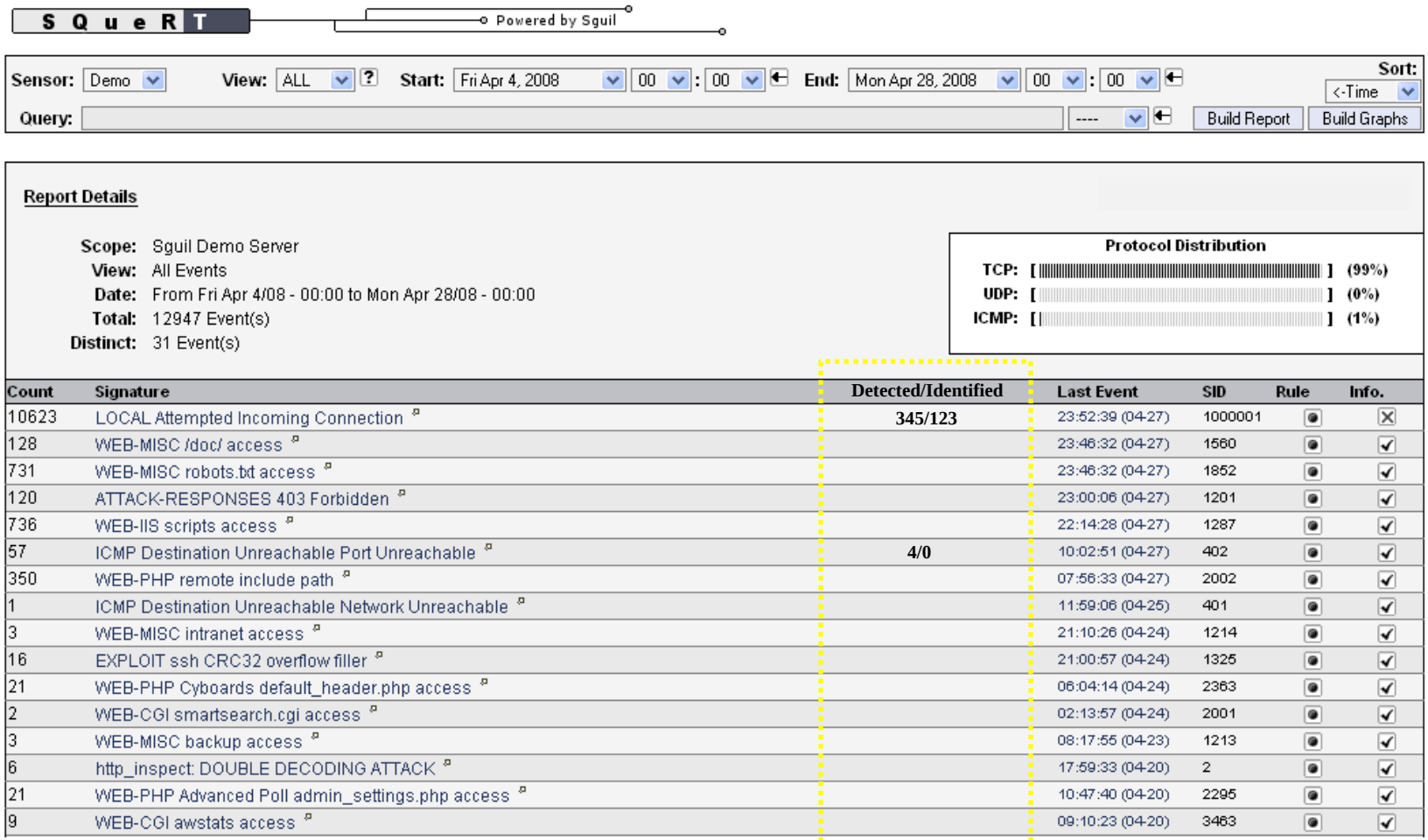
Potential Use



Close Up – Sguil Transcript



Potential Use - SQueRT



Count	Signature	Detected/Identified	Last Event	SID	Rule	Info.
10623	LOCAL Attempted Incoming Connection	345/123	23:52:39 (04-27)	1000001	●	☒
128	WEB-MISC /doc/ access		23:46:32 (04-27)	1560	●	☑
731	WEB-MISC robots.txt access		23:46:32 (04-27)	1852	●	☑
120	ATTACK-RESPONSES 403 Forbidden		23:00:06 (04-27)	1201	●	☑
736	WEB-IIS scripts access		22:14:28 (04-27)	1287	●	☑
57	ICMP Destination Unreachable Port Unreachable	4/0	10:02:51 (04-27)	402	●	☑
350	WEB-PHP remote include path		07:56:33 (04-27)	2002	●	☑
1	ICMP Destination Unreachable Network Unreachable		11:59:06 (04-25)	401	●	☑
3	WEB-MISC intranet access		21:10:26 (04-24)	1214	●	☑
16	EXPLOIT ssh CRC32 overflow filler		21:00:57 (04-24)	1325	●	☑
21	WEB-PHP Cyboards default_header.php access		06:04:14 (04-24)	2363	●	☑
2	WEB-CGI smartsearch.cgi access		02:13:57 (04-24)	2001	●	☑
3	WEB-MISC backup access		08:17:55 (04-23)	1213	●	☑
6	http_inspect: DOUBLE DECODING ATTACK		17:59:33 (04-20)	2	●	☑
21	WEB-PHP Advanced Poll admin_settings.php access		10:47:40 (04-20)	2295	●	☑
9	WEB-CGI awstats access		09:10:23 (04-20)	3463	●	☑

What are we Cataloging?

- Tool Information
  - Default
  - Specific
- Connection Tuple +
  - Src/Dst IP(s) – Src/Dst Port(s) – Time
  - Flags – Protocols – Protocol unspecific, OS dependent, or tool crafted options - etc..

Way Ahead

- Document and share potential benefits
  - To include test cases and bench marks
- Build the frame work
- Build the correlation engine
  - Work with Raymond McCullum (ODU Ph.D. candidate, Statistics)

Questions

- Robert Floodeen
  - Floodeen@outbreaksecurity.com
- Kenneth van Wyk
  - Ken@krvw.com

Backup Slides

Tool	Hosts	Mbit/sec	KBytes/sec	Avg. Packet size	Avg. Packet/sec	Kbytes	Packets
Nmap	Single	0.082	10.223	64 bytes	158.852	229.805	3,571
	Multi	0.069	8.652	62 bytes	138.296	527.394	8,430
Superscan	Single	0.003	0.437	82 bytes	5.263	15.507	187
	Multi	0.004	0.449	71 bytes	6.239	44.854	624
SNScan	Single	0.001	0.142	88 bytes	1.6	1.418	16
	Multi	0.019	2.335	65 bytes	35.493	110.945	1,687
Guardian	Single	0.075	9.367	66 bytes	141.786	843.074	12,762
	Multi	0.061	7.680	63 bytes	120.598	1,632.508	25,637
Nessus	Single	0.042	5.253	70 bytes	74.886	946.798	13,499
	Multi	0.036	4.513	70 bytes	64.232	2,441.158	34,747

Identifying Network Scanning Tools

Robert W. Floodeen
Spectrum Comm Inc
Rob.Floodeen@sptrm.com

Kenneth R. van Wyk
KRvW Associates, LLC
Ken@KRvW.com

This paper is intended to augment and accompany its corresponding slide presentation.[1]

Abstract

Network traffic from automated network scanning tools, if detected, is often discarded as noise; however there seems to continually be a small contingency of researchers working on improving the algorithms employed to detect these scans. No matter what facet is being investigated: the fastest method, the smallest number of packets to detect, lowest false positive rate, lowest false negative rate, or even separating out the data as a whole, researchers generally continue to focus on just detection. In this initial paper we argue for expanding beyond detection and thinking about a comprehensive framework linked to network scan activity which includes Raw Data Management, Detection, Identification, and Cataloging. While simple identification on single packets occurs today, what has been lacking is (1) a detailed automated examination of captured scanning traffic, and (2) a generic framework for facilitating a modular approach to using various but appropriate algorithms and rule sets. We argue that by thinking beyond detection to include identification and cataloging, we will be able to take advantage of the existing detection algorithms to increase the capability to classify the scanning noise. This should help us better understand which of these unknowns should be further investigated.

Description and Background

Network Scanning Tools, by nature, create traffic on the networks they are scanning.

We define scanning tools to mean:

- Port scanners
- Network security scanners
- Vulnerability scanners (to include malicious code)
- Application vulnerability scanners
- Automated penetration testing tools

---

[1] We would like to thank Raymond McCollum from Old Dominion University for input into the framework and specifically working the advanced statistical procedures, which will be explained in a future paper.

We would like to thank Dr. Brett Tjaden at James Madison University for directed guidance on the framework.

Depending on the scope of the target system(s) and the speed of the network scanning tool, a large amount of data can be created over a short period of time. The table below summarizes the benchmark metrics for several common network scanning tools:

Tool	Hosts	Mbit/sec	KBytes/sec	Avg. Packet size	Avg. Packet/sec	Kbytes	Packets
Nmap	Single	0.082	10.223	64 bytes	158.852	229.805	3,571
	Multi	0.069	8.652	62 bytes	138.296	527.394	8,430
Superscan	Single	0.003	0.437	82 bytes	5.263	15.507	187
	Multi	0.004	0.449	71 bytes	6.239	44.854	624
SNScan	Single	0.001	0.142	88 bytes	1.6	1.418	16
	Multi	0.019	2.335	65 bytes	35.493	110.945	1,687
Guardian	Single	0.075	9.367	66 bytes	141.786	843.074	12,762
	Multi	0.061	7.680	63 bytes	120.598	1,632.508	25,637
Nessus	Single	0.042	5.253	70 bytes	74.886	946.798	13,499
	Multi	0.036	4.513	70 bytes	64.232	2,441.158	34,747

Intrusion Detection Specialists or Incident Handlers might not deal with associated raw packet capture data (pcaps) from a reconnaissance scan with every incident they face. In the view of the Authors, this is the result of largely unmanageable volumes of data coupled with current standard practices of incident response. On a “per incident basis” pcaps from reconnaissance scans might get a mention in a report summary, potentially a sample page of alerts, or if fully included, done so simply because of a legal or operational requirement, not because of any perceived intrinsic value.

With the increase in scans and growth of the internet we have seen a trend of ignoring scans, until they either cross a threshold or are tied into a specific incident. They no longer are an incident in and unto themselves. Some initial anecdotal evidence we’ve gathered includes:

When inquiring our peer group, the initial response is

1. There is no value, scan data is just noise

Then, after a discussion on the possibilities we end up with one of the below:

2a. I don't care if I know what the scan is; I only care when I see something new

2b. I don't care what the scan is; I just want to remove it from my "real data"

We identify a short list of the top reasons why scan data is just noise:

1. It occurs so often and there is so much of it, the benefit gained vs the resources consumed equates to a waste of time
2. Reactive measure, not proactive,
3. Any possible information leakage has already occurred,
4. It is not illegal in most environments to ask a system what services or version it offers,
5. "Real attackers do not scan from their attack systems"

This leads to our initial question, is there real value in going through the effort of systematically identifying and tracking the activities of automated network scanning tools?

In 2005 a research paper by Susmit Panjwani et. all. at the University of Maryland attempt to answer the question "*if Port Scans are Precursors to an Attack?"* in (2). They established a test bed system where they captured network data, specifically port scans, ICMP scans, vulnerability scans, and successful attacks. Their conclusion of interest is that they observed about 10% of systems port scanned are followed by an attempted attack. However systems that are port scanned and then vulnerability scanned are attacked 50% of the time. Identification can quickly tell the difference from a port scan and a vulnerability scan. If their numbers are off by a factor, the amount of effort required vs the benefit will still shift in favor of the system owners.

Furthermore identification, not just detection, of scanning tools will present a better method to store the data, which will allow greater trend analysis and correlation over a much longer period of time then is currently being done. The results of which, while not a "silver bullet", would help direct resources and occasionally shortcut a few of the processes in handling an incident. Later in this paper, we will describe our view of how the cost vs the reward, once identification occurs, firmly shifts back in favor of the Incident Handler.

As was suggested earlier, generating a positive cost to benefit return around large scale network scan detection and analysis requires a new approach which will bring great efficiency. The following sections will introduce our approach and the conceptual framework on which it is constructed.

Section 1 - Proposed Approach to a Network Scanning Analysis Framework

It is not the goal of the authors to create a new tool from scratch which would simply be put in place, like an appliance, and left to operate. Instead we are going to present a brief introduction to a conceptual framework and provide considerations for supporting areas. We will then

---

[2] An Experimental Evaluation to Determine if Port Scans are Precursors to an Attack, Susmit Panjwani, Stephanie Tan, Keith M. Jarrin, and Michel Cukier, *dsn*, pp. 602-611, 2005 International Conference on Dependable Systems and Networks (DSN'05), June 2005.

discuss some of the potential benefits and provide a short list of analysis that could be done if such a framework existed.

The field of Intrusion Detection/Prevention is one of measure counter-measure. The opposing force is constantly on the look out for newer, faster, easier, undetectable, methods and tools for compromising systems. This framework would encourage development and rapid introduction of better tools.

Our conceptual framework is divided into four general layers with each having the ability to modularly add or remove applications, databases, and analysis tools as required. Our layers are: *Raw Data*, *Detection, Identification, and Catalog.* Like the concept behind the OSI Reference Model, each layer would be self contained with information passing between them in an expected and controlled manner. This separation allows new modules to "snap in." This permits individual components to enter (or potentially leave) as they are developed or improved. Also like the OSI Reference Model, trying to keep everything to a single layer is not always a clean solution, so communication from one layer to the next is predictable and managed.

The four framework layers are defined to clearly delineate visually the operation or role of a given section.

The following subsections are a brief introduction and considerations for supporting areas and sections in the framework.

Raw Data

Raw Data layer of our framework is created from collection at external boundaries prior to any filtering. It should be standardized but not limited to just *tpcdump* and *Cisco IOS Netflow*. As the requirement for what is captured is driven by the upper layers of the framework, future use must drive capture and storage (e.g. tcpdump data).

Also, it is implied that internal collection would be of little value. This may or may not be true. However, the greatest value for identification is where the largest data population against the largest pool of systems can be captured. Having said that, because of the module design of the framework, internal use may very well have value, we simply have not focused in that direction.

Detection

The Detection layer holds the interface and boundary control necessary to incorporate the scanning tools into the larger analytical framework. Our view is that detection should be tool agnostic supporting not only those popular today but also those of the future. Components in the detection layer will parse data from the Raw Capture layer. When a scan is detected, a record containing identifiable features should be passed to the Identification section. The Identification Section would then pull out the subset identified to analyze it.

Scan Detection is gaining popularity by groups like NETSA at CERT/CC and the associated FloCon community. We anticipate the Scan Detection Engine will mature over the next several years. Our initial work is employing the SiLKtool set as it is supported by the groups mentioned above. While the exact tool may change, the detection framework should not need much rework when it occurs.

Also, we need to take into account things we can not currently identify, but could possibly do so, in the future. For example, a new scanning tool is released. We can detect (by its general behavior) that it is some sort of automated network scanning tool, but we cannot currently identify it. Lack of identification should mean storage for future identification, not discard. So we should consider a method to include detected but unidentified scan data as well.

By using the SiLKtool set, we can demonstrate both of these tasks. We can detect scans to feed into the identification phase and also store data of unidentifiable scans. Again, by separating out the Scan Detection into a different layer, we can deal with it in an abstract way, as long as it can provide and perform the following functions and data:

1. Input formats
2. Output formats
3. Storage Formats
4. Filtering Capabilities

Identification

Currently, as a community, we can loosely define our ability to identify network scanners as analysis on one dimensional descriptive states because we are just using one dimensional values, not entire vectors. This form of analysis, while intuitive, does not take advantage of our available computing resources and statistical analysis theories. The standard view of the problem, that is there is too much data for effective analysis, is in our view incorrect. We believe that this overload of data is created from the incorrect application of simple statistical methods. This is not to say there is no complex analysis in the community today. The IDS/IPS community, with their anomaly detection approaches, are an excellent example of a starting point. With slight adjustments, we can modify the way we detect and increase the way we identify, however this will require advanced use of Descriptive Statistics.

This large amount of data, applied in such a method will actually increase the statistical capability. Also, due to the nature of the TCP/IP stack we actually gain a few more benefits for data reduction, which creates analysis reduction. The TCP/IP stack at the packet level has a defined range of acceptable values in a majority of its fields. These

arbitrary variables with limited to zero variance decrease the amount of work required. This would be similar to saying a password can only use numbers. However, there is a cost in performance and resources, where a trade off will be required for speed. This trade off will be in the probability of a true identification. So one of the data reduction ideas is to use other modular tools and run all of these programs in serial, not parallel. So staying with the overall concept of a modular framework is important, even though we have a potential method for identification we are working towards.

In order to take advantage of these tools, a defined set of points in a packet should be agreed upon. If we were to dissect a TCP packet, what information would be important and what information would be extraneous. It is important to then take the identified information and save it in such a way so it can be quickly manipulated and tested. Below, we suggest items for consideration in the TCP/IP Stack. While we could dissect and define every area in a packet, the math does not care, we should prioritize the information, so that we can identify what items should be dropped last as the consumption of resources on the system increases.

- IP
  - Identification, TTL, Protocol, Source IP, Destination IP, Options
- TCP
  - Source Port, Destination Port, Sequence Number, Acknowledgment Number, Flags, Options
- Data
  - Anything after TCP Options and before the respective footers

Identification will have to maintain its own storage of identification features, be those signatures or correlations.

Identification will interface with Raw Capture and Catalog layers when a new signature is entered into the system to regressively test the previously unidentified scans. It will also have to establish a method to determine complexity of a tool. Initially this may be operator controlled.

Identification might be helped by knowing the attacking host and target operating system. A passive OS fingerprinting tool should be considered.

Catalog

Once Identification is done, a catalog entry is made. Identification should consider what values to pass to the Catalog section as well as back to the Detection and Raw Capture sections when a scan is not identified. It should be cataloged that it was detected and noted that it was not identified, the detection engine might try another module tool to detect it, and at the same time the Raw Capture section needs to save the data for future analysis.

The analytical benefits are realized once the scanning tools have been identified and stored for analysis. In a simple example it is enough to say that a standard rational database is *enough*. However, the more identifications, the stronger the analysis, so a simple rational database might not have the performance requirements years after implementation.

Items we consider important for cataloguing:

- Scanning Tool
  - Name of tool
  - Version of tool
  - Type of tool [Port, Application, Vulnerability, etc.]
  - Options used [if multiple options of significant difference is identifiable]
    - Maintain how often a given option is used (rare options are of more interest, initially)
    - Difficulty to use the option (Opinion)
  - Packets
    - Count each direction
    - Total bytes each direction
- Source IP
  - IP Address
- Source Port
  - A single port, a list of ports, or a range of ports
- Dest IP
  - A single IP, a list of IPs, or a range of IPs
- Dest Port
  - A single port, a list of ports, or a range of ports
- Date
  - Date/Time tool started
- Information collected
  - What is the general purpose of the tool and what types of general information (to include operating systems and applications) does it try to collect.

Below are items which might be helpful from an operational perspective, however might be captured elsewhere. It should be noted that some of these items listed are stored for the long-term. This makes likely several scenarios where incremental changes will occur to individual data elements (e.g. IP-to-Domain). As changes of this nature have great impact in one dimensional analysis, we are targeting framework capabilities which can create lineage relationships to date elements. Supplemental analysis items include:

- Source IP
  - Domain Name, Host Name, Geo-IP location, Operating System (p0f, pads, etc..), ISP, Contact info, Autonomous System,
- Date

    - Date/time tool stopped, total time tool ran, first time tool was seen with this IP, count of times tool has been seen
- Target System (Dest IP)
    - Level of Importance
    - Operating System
    - Major Applications running with version and patch level

An issue to be addressed is how to handle multiple runs of the same tool from the same system against multiple systems. Would this be a single record with a span of IP's, or would this be multiple records? Is the deciding factor the type of scan, or the unique systems touched?

Finally, the catalog section should provide the ability to sterilize data for export to a larger public community. This would be of value when a given method of scan is being used very successfully with attacks.

Section 2 – Benefits

This section assumes the framework is successful in creating an interoperable environment in which large volumes of network scanning data can be retained and analyzed over the long term. It should be pointed out again, that the authors are not suggesting a "silver bullet" or that these benefits will work in every case.

We start with the simple benefits which could be applied to intrusion detection, prevention, and network sniffing systems used for in depth analysis and conclude with the analysis benefits for Incident Handlers, Attack Sense and Warning, and Intrusion Detection/Prevention Analysts.

System Benefits

Full pcap data reduction. (point 2b from Description)
It is a general practice to keep as much data as possible for as long as possible for data analysis. Unfortunately this is not practical over a long period of time. Since this is not practical, many organizations will have each sensor delete data in order to write new data. This keeps pcaps on the sensors for the longest period of time. If we can detect a scan, it is of enough effort to delete the scan from the pcap data, especially the vulnerability assessment scans with their size. This is not a complicated task and can easily be done with out our framework. However, that would create a loss of any long term analysis benefits.

It should be noted that deletion of scan logs could be against your local Standard Operating Procedure (SOP) or in worse case hamper legal prosecution of an intrusion. Please take the proper procedures to ensure this is acceptable in your operation.

Log Data Reduction
In the US Department of Defense, there is a requirement that certain systems have logging turned on and they do not overwrite themselves. We have seen denial of service on systems because their security logs have filled up and a security administrator was not around to review and clean the logs. Essentially the system would not allow anyone to login who was not an administrator.

If we can identify a scan and determine what they do to a system, we can then safely remove the associated logs from those systems. This is not limited to just a web server (attempted login attempts), or file server, FTP, etc.. This would also have value in removing the logs from your IDS and Network Security Monitoring tools.

Free up Intrusion Detection systems from looking for scans
As mentioned in the introduction paragraph of Dedicated Server(s) subsection, this is not a driving benefit. In other words the majority of operations might not see an immediate benefit based on their deployment and what they are watching. But for those few who have already *tuned out* scan data for this very reason, it will give them back an ability to manage the reconnaissance and vulnerability data being thrown at their systems.

Pare down false positives
An Intrusion detection system could generate alerts without regards to Operating System of the system being defended. With a passive Operating System fingerprinting tool, the ability to determine if a scan will have an impact could be considered.

Also, with better identification techniques, if the framework was to find a supporting role in a NSM, the correlation engine could use the alert information and correlate it with other alerts from the same source IP, which will provide an analyst with a broader picture.

Analysis Benefits

Finger Printing an Attacker
An attacker could be a single person/system, or a group of people and systems. We are globalizing, and the price to participate as an attacker is a laptop and a network connection. But the attackers are not the only ones growing, so are the targets. Given a sufficient system (multiple networks that share the information from the framework, not necessarily owned or controlled by a single entity) it would be possible to correlate a specific method of scanning, given that they do something unique.

This fingerprint could then be expanded to all records to determine if other IP's have conducted similar activity. We would expect a portion of these findings to be tied back to instructional videos (shmoocon, youtube), reading materials (books, blogs), or in best case, defined methodologies by the attacker.

Expanding the scope of an incident
If we can Finger Print the associated scanning technique to an intrusion, we could possibly expand the scope of the incident by that technique. If multiple IPs were used to conduct the scanning, this would help identify them, even though they might not have been involved with the incident. By potentially extending the scope, leads could be generated to other incidents not yet identified.

Knowing what we don't know (from 2a in the Description section)
There are three scenarios in this benefit. The first is the ability to identify a modification to a tool of some complexity. The statistical analysis promises to identify a value of closeness. The concept of a skilled attacker could be one who modifies open source scanning tools to meet a specific need.

Nmap is still one of the best scanners. The rate of change in the tool selection for performing a scan is low. Over time, we will identify the majority of the popular and unpopular tools, which will leave those not so often used. The second scenario asks the question "why are we not seeing this very often?" Is it home made, designed for a specific task, or simply brand new? Just like the sentence in section 2a, we might not care how many times a specific Nmap tool scans us, but as soon as something new appears, we would be interested.

If we can couple this benefit with "Finger Printing an attacker", then we would be *very* interested to determine when a known scanning technique by a known attacker suddenly changes their technique. To take it a step further, we would also be interested in the intensity of the scan, the systems targeted, and the time between scans.

Directing Resources in a triage method
From the benefit above "Knowing what we don't know", one of the first and obvious benefits to this framework is the ability to Identify what is unknown. Matched with the ability to establish a complexity level, we can then start to triage unknowns.

On the opposite side of that concept is the "Finger Print an Attacker", also a benefit listed above. We can proactively identify techniques used and elevate them in the triage processes to ensure proper actions are taken in a timely manner. It would be assumed that the two common actions would be to deny access to systems, or allow the traffic through to be monitored. It is possible that this traffic could be redirected into a Honey Network.

To reiterate the point, this type of activity and techniques could be shared with a larger community, alerting them to items to watch for.

Information gained about your system (what are they after?)
How can we determine what they are after? It is clearly easy to identify that a port 80 scan is probably looking for web servers. By identifying the tool, it could be possible to

identify what web server application (apache, iis) it is looking for, without spending minutes to hours pouring over the individual packets.

Reviewing previously unidentified scans
Like a zero day, a newly release tool will not have a signature out of the box. Currently, that means our community would simply ignore and lose any potential analytical value from it. With our framework, this information would be stored and retested when new signatures are defined. Just like the concern for backward secrecy, suddenly identifying a newly release tool could produce leads currently unnoticed by the Incident Handling Staff.

Metrics and Trending
We are suggesting several initial metrics as part of this process:

- Maintain a detected scan to attack ratio
- Maintain an Identified scan to attack ratio
- Maintain an attack to non-detected scan ratio
- Maintain a known to an unknown scan ratio

The suggested metrics could be broken down into multiple subgroups. At the least a short list of these subgroups would be:

  - By each tool
  - By each type of tool
  - By each target
  - By each type of target
  - By each network block
  - During a specific time of day
  - Associated with a new vulnerability over time
  - By size of scan
  - By number of packets

These metrics could be used in communication with other groups without giving away sensitive data.

Also, these metrics could be used to define precursors for future warnings. Intuitively it would seem that the larger the networks (broken into defined segments) the more value these precursors would hold. So this is a metric we are interested in following to determine if that is the case.

Conclusion and Way Ahead

While analysts might not be lacking the skills to deal with scans, they are lacking the tools in order to apply those skills effectively due to the sheer number of scans, the time to review each one, and system resources consumed. Our suggested Framework will provide a modular environment in which communication between the layers will be controlled. This approach will pave an avenue previously left unattended and shift the

value of expending resources on scanning data back into the favor of the analyst (without having to write an incident report on every scan detected).

The next step is to have an open dialogue about the potential benefits, risks, and added management issues with such a tool. While doing that, we will be building the framework and populating it with existing solutions for each of the modules. We will also have to design the catalog database and determine the overall best scheme for data management. This will consider interaction with existing schemes already deployed in support of IDS/IPS. When the simple framework is built and the alpha tool is completed, work with Raymond McCollum on a specific algorithm for identification will commence. The last and most important aspect to a long and continued success is the ability for a user to generate a new identification signature from an unknown tool. Tools like Nmap have tackled this issue well, and we will have to do the same, to remain relevant and useful.