Software Engineering Institute

Network Profiling Using Flow

Austin Whisnant
Sid Faber

August 2012

TECHNICAL REPORT
CMU/SEI-2012-TR-006
ESC-TR-2012-006

CERT® Program

http://www.sei.cmu.edu

Carnegie Mellon

This material is based upon work funded and supported by United States Department of Defense under Contract No. FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

This report was prepared for the

SEI Administrative Agent
AFLCMC/PZE
20 Schilling Circle, Bldg 1305, 3rd floor
Hanscom AFB, MA 01731-2125

Table of Contents

List of Figures

List of Tables

Acknowledgments

Special thanks to George Jones in the CERT Program for helping produce the System for Internet-Level Knowledge (SiLK) command-line arguments and shell scripts.

Thanks to the organization that provided sample data for the case study.

Abstract

This report provides a step-by-step guide for profiling—discovering public-facing assets on a network—using network flow (netflow) data. Netflow data can be used for forensic purposes, for finding malicious activity, and for determining appropriate prioritization settings. The goal of this report is to create a profile to see a potential attacker's view of an external network.

Readers will learn how to choose a data set, find the top assets and services with the most traffic on the network, and profile several services. A case study provides an example of the profiling process. The underlying concepts of using netflow data are presented so that readers can apply the approach to other cases. A reader using this report to profile a network can expect to end with a list of public-facing assets and the ports on which each is communicating and may also learn other pertinent information, such as external IP addresses, to which the asset is connecting. This report also provides ideas for using, maintaining, and reporting on findings. The appendices include an example profile and scripts for running the commands in the report. The scripts are a summary only and cannot replace reading and understanding this report.

1 Introduction

A network profile is an inventory of all the assets on a network and their associated purpose. Such a profile can enable network administrators to better consider how decisions about configuration changes will affect the rest of the assets on the network. Security administrators can evaluate the profile for assets that violate policy and for any suspicious activity. Business administrators can use the profile to help guide long-term plans for network upgrades and staffing. As the profile changes over time, network operators and defenders can monitor for emerging concerns. This, in turn, can lead to policy changes and reallocation of network resources.

This report discusses the steps for creating a profile of externally facing assets on mid-sized to large networks serving thousands to hundreds of thousands of users. The steps involve analysis of traffic over ports, protocols, and other network flow (netflow) data available at the perimeter gateways. While some of the steps may be useful for profiling traffic on an intranet, there are additional issues related to intranets that are not addressed in this report. By the end of this tutorial, you should have a list of assets combined with the ports on which each is communicating and notes on any associated questionable activity.

The general steps for network profiling as detailed in this report are as follows: (1) gather available network information, (2) select an initial data set, (3) identify the active address space, (4) catalog common services, (5) catalog other services, (6) catalog leftover assets, and (7) report on findings. These steps can be turned into a cyclic feedback loop to maintain the profile as shown in Figure 1.

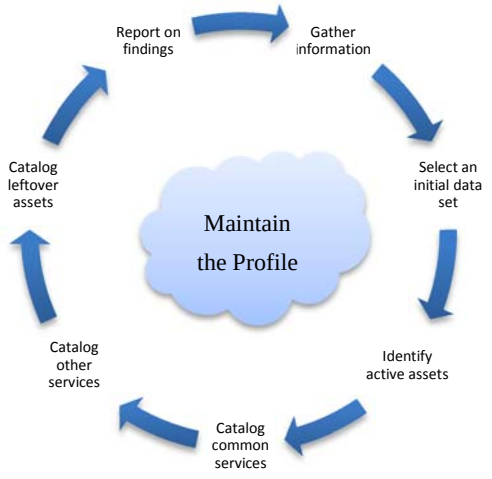

Figure 1: Network Profiling Process Cycle

Before beginning, ensure there are enough resources to devote to this exercise. This process should be completed within a fixed amount of time so that the results are relevant. For networks with relatively fixed assets, this process could take place over one to two months. For faster changing networks, it could take place in as little as one to two weeks. The amount of time and

other resources that it will take to complete the process depends primarily on the size of the network and how well the assets conform to networking common practices.

Throughout this report, validation will be discussed as a key component of the profiling process. It is essential to validate results before adding them to the profile. In general, there are two types of validation: active and passive. Passive validation uses only stored data without extra resources. Active validation involves using manual effort to validate initial findings by attempting to make connections to the specific IP address and using third-party references. For example, one can validate a potential web server by browsing to port 80 or by performing a name server lookup on that IP address. When manual validation is not possible through these means, communication with the owner or administrator of the device may be necessary, if possible.

This process uses netflow traffic to perform an analysis. We used the System for Internet-Level Knowledge (SiLK)[1] tool to collect and analyze the traffic, and we include examples of the SilK commands and results of the commands in each of the steps. However, the steps apply to any flow analysis tool.

1.1 Sample Data

We demonstrate how to create a network profile using sample data collected from the perimeter of an enterprise network. These data were anonymized after analysis to protect the confidentiality of the network owner without impairing the data's usefulness.

1.2 The SiLK Analysis Tool Suite

Because the case study in this report uses SiLK for analysis, you should understand how SiLK records flow data.

SiLK deals with uniflow traffic, meaning traffic coming into the network is recorded as separate flows from traffic going out of the network. Although SiLK differentiates between inbound and outbound traffic based on the source and destination IP addresses, it does not attempt to identify traffic as either client or server, as do some other flow platforms.

SiLK is configured by setting an address range for the internal network. The type of flow is then based on whether the source and destination IP addresses are inside or outside of that range. As shown in Figure 2, a flow of type "in" is defined as traffic with an external source address and internal destination address. A flow of type "out" is defined as traffic with an internal source address and an external destination address.

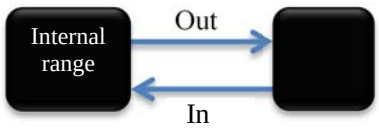

Figure 2: SiLK Flow Types

[1] For more information, visit http://tools.netsa.cert.org/silk/.

Web traffic is separated from all other traffic and is defined in SiLK by default as traffic to or from ports 80, 443, or 8080 and is labeled as “inweb” or “outweb” based on the same reasoning as flows of type “in” or “out.”

1.3 Keeping Track of Findings

A spreadsheet like Table 1 is extremely useful to record findings throughout the profiling process. The headers you choose will depend on the information that is needed about the network and may be adapted for each step of the process.

Table 1: Example Profiling Spreadsheet

Internal IP	Protocol	Internal Port	Internal Name	External IP	External Port	External Name	Comments

Throughout the process, record the commands and tools used to gather and validate the data. This record will enable automation of certain parts of the process, making future updates less labor intensive, and will allow for reproducible results. As an example, shell scripts have been included in Appendix B of this report. Note that the scripts are provided only for reference and may or may not be appropriate for a specific network.

1.4 Extending the Analysis

The steps in this report are, by no means, the only way to use network flow data to learn about a network. As you get comfortable using the features of the analysis tool, you should feel capable of delving into further detail if the traffic flows look interesting or out of place. Flow data can be used for forensics purposes, for finding malicious activity, and for determining appropriate packet prioritization settings, among other things.

2 Gather Available Network Information

Gathering any available information about the network prior to beginning the profile is an important step because it will help set the scope for the rest of the process. The types of information that could be collected include items such as address space, network maps, lists of servers and proxies, and policies governing network design. This information may be incomplete or out of date, but it provides, at a minimum, a starting point for known devices and a reference for potential problem points that may arise during the profiling process that require additional effort to validate. Familiarity with the organization's network and security policies is also beneficial, as it enables the profiler to notice discrepancies between the policies and the actual findings. The output of the profiling effort may reveal compliance issues or may suggest potential changes to security policies that are worth considering.

It is important to know what you expect to see when starting the profile, but it is just as important to realize that not everything about the network is known. For example, an old File Transfer Protocol (FTP) server dedicated to internal file sharing may have been temporarily opened to outside services but then forgotten about when that capability was no longer needed. Such carelessness can lead to lack of information, which results in holes in network security.

If time allows, consider conducting a quick assessment or penetration test to develop a network map and a list of exposed services on various machines. Many automated tools, some free such as nmap,[2] are available for network mapping, and most network monitoring solutions have built-in network mapping capabilities. Be sure to obtain permission to run active scans on a network before doing so, as initiating active scans and probes could violate company policy or negatively affect the performance of systems and services on the network.

When the profile is complete, update the network maps and lists of servers so that the process can be cycled through again in the future.

2.1 Sample Network Information

For the purposes of this report's case study, we initially assumed only the following about the network being profiled:

- size: thousands of users
- owner: a mid-sized organization using the network for its business purposes
- CIDR: 203.0.113.0/24 (203.0.113.0 – 203.0.113.255)

[2] "Nmap ('Network Mapper') is a free and open source (license) utility for network exploration or security auditing." Source: nmap.org

3 Select an Initial Data Set

Choosing the initial data set for analysis is important because it shapes the entire analysis. Take some time to obtain a good representative sample of data that still remains a reasonable size. A data set large enough to represent typical traffic is necessary, but it should be small enough to be able to iteratively process queries.

Before selecting the initial data set, understand how the sensor placement and flow collection configurations affect the available flow data.

3.1 Sensor Placement and Configuration

The importance of sensor placement should not be underestimated. Placement affects what flow data is and is not collected, as well as which IP addresses are associated with each flow record.

The following framework will help you decide the most effective sensor placement and configuration for the network you will profile.

Proper sensor placement for network profiling takes into account several considerations: the goal of the flow collection—in this case, network profiling—the network topology, and the network hardware in use. For example, some network hardware or network security devices, such as proxy servers or firewalls, can make visibility into the network difficult or impossible with flow data alone. The goal of this report's step-by-step process is to profile perimeter traffic to see what a network looks like when viewed externally by a potential attacker. Therefore, sensors should be placed on the external, or internet-facing, side of any perimeter networking devices. Sensor placement for other goals may have different requirements and is, thus, out of the scope of this report.

When a network is split up into intranets, it is tempting to profile each one individually. Remember, the goal is to profile the network from the view of an outsider, so place the sensors around the perimeter of the largest extranet that needs to be profiled. If necessary, divide the data collected by address blocks to view differences between intranets. Note that profiling anything except the entire network may leave out assets not intended to be left out.

Remote and business-to-business networks often have their own gateway into a network. Include these gateways when placing sensors at all access points to the network. Note any special access points like these so that you are aware that traffic across these sensors may be different than typical traffic at other sensors. This same reasoning applies to business continuity links, which should be included in the profile with a note that traffic at these sensors will be different than traffic at other sensors. Table 2 contains guidelines for sensor placement.

Table 2: Sensor Placement Guidelines

Configuration	Placement
Multiple exit points	Make sure all access points connecting the network to other networks are covered.
Network/security devices (proxies, NATs, firewalls, etc.)	Sensors should be placed on the external side of these devices.

Configuration	Placement
Intranets/extranets	Place sensors around the largest extranet that needs to be profiled.
Remote networks, failover access points	Place sensors at these access points, making a note of their special purpose.

Table 3 shows an example network with sensors placed on the internet-facing side of its two main gateways, as well as on the internet-facing side of its remote office gateway.

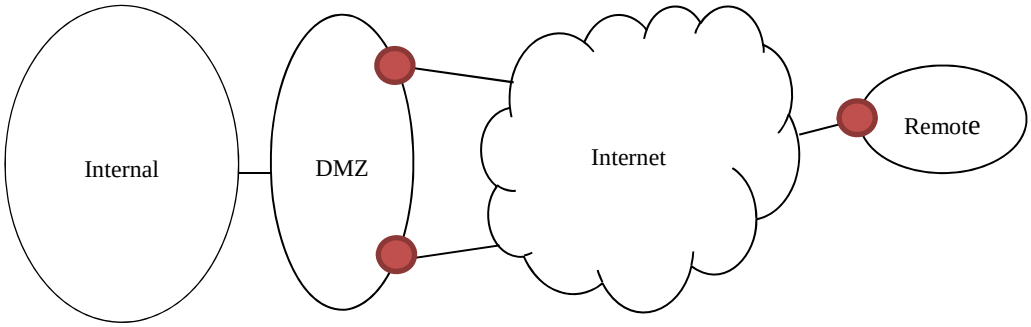

Figure 3: Example Sensor Placement

While working with the data, you should see plausible amounts of traffic for expected assets. For example, if a webserver has a 200 Mbps network interface card, expect to see traffic coming to and from that webserver at a rate of less than 200 Mbps.

3.2 Guidelines

Guidelines for selecting a sample data set are listed in Table 3. It is not necessary to ensure that the sample data set is representative of *all* traffic that crosses the network boundary. Once built, the profile will be reapplied to the rest of the data set to make sure nothing is missed. Selecting a sample data set should be done after the sensors are placed and network flow has been collected.

Table 3: Guidelines for Selecting a Data Set

Feature	Considerations
Duration	Start with at least an hour's worth of data. Add more data—up to a day's worth—until the query time starts to slow down. The query time for the entire initial data set should be 15-60 seconds.
Timing	Select the busiest time of day to quickly carve out the most common network traffic.
Direction	If the traffic is bidirectional and it is necessary to further reduce the sample size in order to reduce the query time, start by looking at outbound traffic—traffic that is generated by internal equipment.
Sampling	Avoid starting with sampled data if possible because it may mask some important and routine behaviors.
Network size	If the network is extremely large but divides into logical boundaries by IP address, consider separating the analysis into a few independent profiles and merging them after you complete the analysis.

For the sample network, we used the following command to choose a day's worth of data from Wednesday, September 28, 2011, which is representative of a typical workday, as the sample data set.

```
$ rwfilter --start-date=2011/09/28:00 --end-date=2011/09/28:23 \
--type=all --protocol=0- --pass=sample.rw
```

Note: rwfilter is the main SiLK command, which allows selection and partitioning of flow records based on various fields in the record.

In the case of the sample network, there was enough processing power to query a data set of this size. No other filtering was done based on start or end times, flow duration, address space, direction, or other characteristics.

The following example shows the command we performed on the sample data set and the result that it has 10,985,967 total records and is 571 MB in size.

```
$ rwfileinfo sample.rw
sample.rw:
  format(id)          FT_RWGENERIC(0x16)
  version             16
  byte-order          littleEndian
  compression(id)     none(0)
  header-length       156
  record-length       52
  record-version      5
  silk-version        2.4.2
  count-records       10985967
  file-size           571270440
  command-lines
                  1  rwfilter --type=all --start-date=2011/09/28:00 \
--end-date=2011/09/28:23 --protocol=0- --pass=sample.rw
```

3.3 Validating the Selection

After identifying a sample data set to work with, it is worthwhile to take a few extra minutes to do some surface analysis of the sample to confirm that it seems to represent the network. A quick inspection of the sample will save time that could otherwise be wasted on an improperly collected or filtered data set.

The highest volume data paths should be no surprise and should be obvious in the data set. Check typical ports, protocols, and address blocks to ensure the sample contains expected data. Table 4 includes a few possible tests for validating the selection.

Table 4: Validating the Initial Data Set

Expected Asset	Expected Sample Traffic	Method of Verifying the Sample
Web servers	TCP traffic with an internal source address and a *source* port of 80 or 443	Using a web browser, browse to those IP addresses.
Web clients	TCP traffic with an internal source address and a *destination* port of 80 or 443	Make sure gateways and/or proxy servers are in this list.
Email servers	TCP traffic with an internal destination address and a source port of 25	These addresses should match the mail exchanger (MX) records of the network.
A business-to-business VPN	IP protocol 50 (ESP) and 51 (AH); UDP port 500 (IKE) or TCP port 4500 (NAT-T)	The external addresses should be business partner networks.

3.3.1 Sample Network Data Set Validation

Because there was no network map or list of servers for the sample network, the sample data set was validated by inspecting for the types of traffic typically seen during a workday on a business network. Basic networking knowledge indicates that almost all of the traffic should be over TCP and UDP. This can be verified by dividing the traffic volumes for the sample network based on IP protocol number and looking at the top five protocols in use. The following command and output from the sample data show that there was also some ICMP (protocol 1) traffic, and ESP and IPv4 encapsulation protocols (50 and 4 respectively).

```
$ rwstats sample.rw --fields=protocol --count=5
INPUT: 10985967 Records for 7 Bins and 10985967 Total Records
OUTPUT: Top 5 Bins by Records
pro|   Records|  %Records|   cumul_%|
  6|   7302815| 66.474030| 66.474030|
 17|   3605304| 32.817357| 99.291387|
  1|     72762|  0.662318| 99.953705|
 50|      5079|  0.046232| 99.999936|
  4|         3|  0.000027| 99.999964|
```

Figure 4 is a graphical representation, similar to graphs that some network analysis tools create, of the traffic volume of the top five protocols on the sample network.

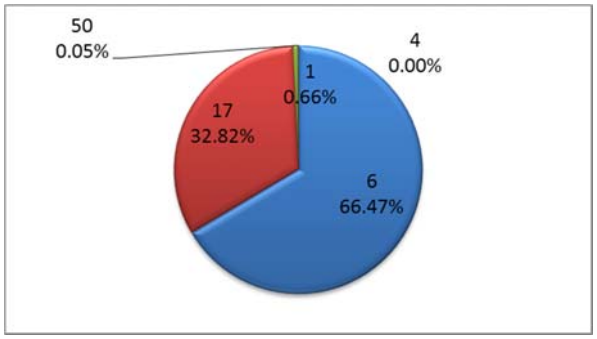

Figure 4: Top Five IP Protocols on the Sample Network

Note: In SiLK, rwstats is the command that groups flows by a specific key or, in this case, protocol, and prints the top values by traffic volume percentage. Other analysis tools may show this graphically, as seen in Figure 4.

The top expected services requested by clients on a typical network are web (ports 80 and 443), DNS (53), and SMTP (25), so verify by sorting outgoing traffic into the top five most common destination ports. The statistics that follow the SiLK command below and are represented graphically in Figure 5 show some traffic to port zero, which is SiLK's notation for traffic on IP protocols other than TCP and UDP.

```
$ rwfilter sample.rw --type=out,outweb --protocol=0- --pass=stdout \
| rwstats --count=5 --fields=dport
INPUT: 5064003 Records for 64477 Bins and 5064003 Total Records
OUTPUT: Top 5 Bins by Records
dPort|   Records|  %Records|   cumul_%|
   80|   2625707| 51.850423| 51.850423|
   53|   1530900| 30.231025| 82.081448|
```

```
  443|    291927|  5.764748| 87.846196|
   25|     13910|  0.274684| 88.120880|
    0|      4000|  0.078989| 88.199869|
```

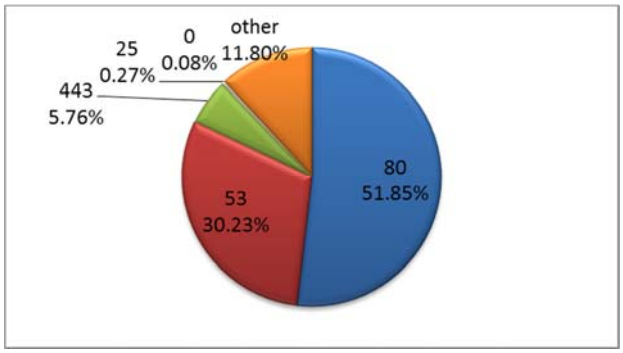

Figure 5: Destination Ports for Outbound Traffic from the Sample Network

The output shows that the expected services are being used. Also, servers on the network are likely to handle the same services.

Next, check the *source* port of outbound traffic. The SiLK command and the output from the sample network follow.

```
$ rwfilter sample.rw --type=out,outweb --protocol=0- --pass=stdout \
| rwstats --count=5 --fields=sport
INPUT: 5064003 Records for 64897 Bins and 5064003 Total Records
OUTPUT: Top 5 Bins by Records
sPort|   Records|  %Records|   cumul_%|
   53|    291693|  5.760127|  5.760127|
   80|    129093|  2.549228|  8.309355|
   25|     85331|  1.685050|  9.994406|
  443|     71429|  1.410524| 11.404930|
    0|      6644|  0.131201| 11.536131|
```

Figure 6 shows the top five source ports in relation to the rest of the outbound traffic (labeled "other" in the graphic).

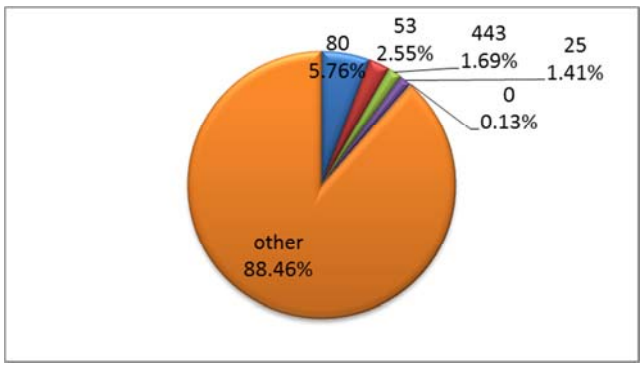

Figure 6: Source Ports for Outbound Traffic from the Sample Network

This second analysis shows that the services in use—DNS, web, and email servers—are as expected. You can safely assume at this point that the available data has been properly sampled and can proceed with the profiling process.

4 Identify the Monitored Address Space

Start network profiling activities by becoming familiar with the network address topology and how well the sensors can see it. Some issues involving monitored address space are whether sensors cover any private address space, what traffic is expected on failover circuits during normal operations, and whether a business unit has connected a system without administrator knowledge. This section explores some of these issues by first finding populated addresses.

The process to identify the monitored address space follows these steps:

1. Identify hosts that have active TCP connections.
2. Identify hosts that generate a nontrivial amount of traffic on protocols other than TCP.
3. Aggregate individual hosts into populated network blocks.
4. Examine additional information to confirm the list of active IP address blocks.

Figure 7 shows steps 1 and 2. Upon completing all four steps, you will have an idea of the active hosts that are seen by the network sensors. The intent at this point is not to identify and name every observed device, but simply to understand sensor coverage and compare it with what is already known about the topology of the network.

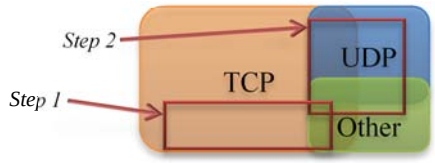

Figure 7: Active Hosts

4.1 TCP Talkers

For most networks, the bulk of traffic happens on TCP (IP protocol 6). Therefore, you can find most active network hosts (also referred to as “talkers”) by looking for legitimate TCP sessions.

Sustained TCP conversations are relatively easy to identify in flow. However, pay careful attention to eliminate scan traffic and “ghost” traffic—attempts to connect to hosts that no longer exist. The purpose of getting rid of the ghost traffic is not because it is inconsequential, but simply because it is not helpful for identifying active internal hosts. Criteria for finding active TCP hosts are listed in Table 5.

Table 5: Active TCP Host Criteria

Criteria	Explanation
Outbound traffic	Outbound traffic is generated from the internal IP block, so no unsolicited traffic or public scanning should be contained in this traffic.
Protocol 6	TCP traffic
More than four packets	To establish a TCP session, two packets must be sent in each direction. Another one or two packets are required to tear down the session. Requiring at least four packets ensures that at least some data are exchanged between the client and the server.
ACK flag set	This eliminates flows consisting entirely of SYN or RST packets, which could be the result of scans or ghosts.

Use the following command to apply these filter criteria to the sample traffic and obtain a list of the source IP addresses actively using TCP.

```
$ rwfilter sample.rw --type=out,outweb                \
--protocol=6 --packets=4- --ack-flag=1 --pass=stdout \
| rwset --sip-file=tcp_talkers.set
$ rwsetcat tcp_talkers.set --count
36
```

The command's output shows that the sample network has 36 active TCP talkers. The addresses of these hosts were placed into a SiLK set file[3] for future reference. Saving the list of IP addresses is an important step to eliminate redundant queries later on.

4.2 Other Talkers

TCP was used to identify active hosts first to eliminate as much scanning and ghost activity as possible; however, many connectionless protocols are used on networks. In addition to finding TCP talkers, it is important to find IP addresses on the network that are actively using other protocols.

You should separate web-browsing traffic (ports 80, 8080, and 443) from all other traffic if the analysis software has the capability to do so. Web-browsing traffic is not normally carried over protocols other than TCP. Leave out this traffic over non-TCP protocols for now.[4] See the following commands as an example.

```
$ rwfilter sample.rw --type=out --protocol=0-5,7-255 --pass=stdout \
| rwset --sip-file=other_talkers.set
$ rwsetcat --count other_talkers.set
25
```

The output of these commands shows that the sample network has 25 active talkers on other protocols.

4.3 Aggregating Hosts

You should now have two lists of active addresses—TCP and non-TCP. Join the lists together and remove any duplicates. The result will be all of the active assets from the sample set, without too much extraneous traffic. If the list is large, consider dividing it into groups of smaller netblocks for separate analysis. The SiLK command for joining the TCP and non-TCP lists follows.

```
$ rwsettool --union tcp_talkers.set other_talkers.set \
--output-path=talkers.set
$ rwsetcat --count talkers.set
40
```

Combining the TCP and "other talkers" lists for the sample network results in 40 active talkers, as shown in the command output above.

---

[3] The SiLK set file is located at http://tools.netsa.cert.org/silk/rwset.html.

[4] In SiLK, "out" traffic does not include traffic to or from ports 80, 8080, or 443; those are included in "outweb" traffic.

If this number seems small given the size of the netblock being analyzed, remember that the sensors have been placed on the external side of perimeter networking devices. It is very likely that client traffic is going through proxies or Network Address Translation (NAT) servers prior to reaching the sensor.

It may be helpful for larger networks to aggregate the IP addresses into netblocks.[5] The following example includes the command and output.

```
$ rwsetcat talkers.set --network-structure=X
  203.0.113.0/27|  9
 203.0.113.32/27|  8
 203.0.113.64/27|  5
 203.0.113.96/27|  1
203.0.113.160/27|  4
203.0.113.192/27| 13
```

If necessary, you can profile the addresses in each block separately.

Once you complete these steps, you will no longer need the TCP talkers and other talkers lists.

4.4 Supplemental Analysis and Validation

Some network monitoring systems may have visualization capabilities that chart the most active hosts or services, plot port usage across time,[6] or view a graph of IP addresses versus ports. If these tools are available, use them to get an overview of the traffic and hosts on the network, but be careful about relying on them for profiling. In general, visualizations are not detailed enough to get an accurate picture of what is going on in a network, but they can help guide the process. Once the process is complete, visualizations become important for reporting and reference material.

Use graphs to supplement the analysis, but do not rely on them as a final authority.

Use the information gathered previously about the network to validate the active hosts. For example, if there are supposed to be 10 servers in the demilitarized zone (DMZ) but the active hosts list has only five entries, there may be either a misconfiguration on the network or a misplaced sensor.

4.5 Anomalies

You should now have a list of active hosts on the network, as seen by the outside world; however, in rare cases, some of this traffic may be merely passing through (transiting) the network. Check for transit traffic by looking for traffic leaving the network that is not from an internal host. Conversely, it may also be worthwhile to check for any traffic heading into the network from an internal host (if inbound traffic is available) and for any traffic heading out of the network to an internal host. The following example shows the command and results of this step.

---

[5] Choose from `--network-structure=[A,B,C,X]` to define /8, /16, /24, or /27 address blocks, respectively.

[6] This is also helpful for finding proxy servers because of their unique port usage pattern.

```
$ rwfilter sample.rw --type=out,outweb --not-sipset=talkers.set \
--print-statistics
Files     1. Read   10985967. Pass          0. Fail    10985967.
$ rwfilter sample.rw --type=in,inweb --sipset=talkers.set \
--print-statistics
Files     1. Read   10985967. Pass          0. Fail    10985967.
$ rwfilter sample.rw --type=out,outweb --dipset=talkers.set \
--print-statistics
Files     1. Read   10985967. Pass          0. Fail    10985967.
```

In the sample network, no transit traffic was found. In fact, the collection system is configured with the correct netblock of internal IP addresses and set to record internal-to-internal communication separately from other communication. Understand how the specific collection system for the network being profiled is configured before moving on with this process.

Asymmetric routing can also create difficulties for interpreting flow traffic and may occur in networks with two or more active internet connections. If sensors do not cover the alternate connections, traffic may not be collected from one direction of the flow. Different flow tools handle asymmetric routing in different ways, so it is important to be aware of how the netflow analysis tool you're using handles asymmetric routing and to adjust the procedure accordingly.

For example, with the SiLK flow analysis suite, unique flows are determined based on protocol, source IP, source port, destination IP, and destination port. SiLK creates a separate flow for each direction of an actual flow. Therefore, the choice of using outbound traffic only for finding active talkers was appropriate because asymmetric routing was not a factor.

5 Catalog Common Services

After identifying the active hosts on the network, continue the profile by inventorying the services that typically constitute the majority of bandwidth usage and business operations, such as web traffic (both client and server) and email. Once you have carved these protocols out of the data set, start working on other services that are likely to be on the network and also visible to instrumentation: Virtual Private Network (VPN), Domain Name Server (DNS), and FTP, and other less used but common protocols. This section includes a template for cataloging other services that may not specifically be listed but make up a large part of the traffic volume on the network.

5.1 Web Servers

Web servers are often the easiest asset to profile and will commonly constitute a large percentage of web traffic. Start by looking for connections into the network destined to a service running on ports 80, 8080, or 443. Also consider whether any other web services are running on the network that would require adding a port to this list, such as streaming services or a port configuration different from the default. For example, web services sometimes reside on ports 81 or 82.

5.1.1 The Process

1. Compile a list of the busiest IP addresses with connections coming from ports 80, 8080, and 443. Filter out flows that do not complete a TCP handshake (the SYN, SYN-ACK, ACK traffic pattern used to initiate TCP connections) by selecting only flows that have an ACK flag set and that are at least four packets long. This will eliminate scans and accidental attempts to connect. Choose only addresses that make up at least 1% of the traffic being profiled (in bytes). These are the busiest web servers. The SiLK command and output from the sample data for this process follow.

```
$ rwfilter sample.rw --type=outweb                          \
--sport=80,443,8080 --protocol=6 --packets=4- --ack-flag=1 \
--pass=stdout                                               \
| rwstats --fields=sip --percentage=1 --bytes
INPUT: 190814 Records for 24 Bins and 15207959195 Total Bytes
OUTPUT: Top 7 bins by Bytes (1% == 152079591)
           sIP|               Bytes|    %Bytes|   cumul_%|
 203.0.113.69|          7888143831| 51.868523| 51.868523|
 203.0.113.28|          3272828647| 21.520499| 73.389022|
203.0.113.198|          2631884565| 17.305968| 90.694990|
203.0.113.194|           652195072|  4.288511| 94.983501|
203.0.113.196|           255865129|  1.682442| 96.665944|
 203.0.113.44|           254700968|  1.674787| 98.340731|
203.0.113.197|           152764557|  1.004504| 99.345235|
```

The result of the above command is a list of seven IP addresses accounting for 99.34% of the web server traffic. Save this list of IP addresses as a file named web_servers, as in the following example. It will be needed several times during the profiling process.

```
$ rwfilter sample.rw --type=outweb                          \
--sport=80,8080,443 --protocol=6 --packets=4- --ack-flag=1 \
--pass=stdout                                               \
| rwstats --fields=sip --percentage=1 --bytes --no-titles  \
| cut -f 1 -d "|"                                          \
| rwsetbuild > web_servers.set
```

2. Next, associate each IP address with the web ports with which it is interacting. Place these addresses and their associated ports into the profile table. To find the associated ports, filter outbound traffic from the web server addresses for source ports 80, 443, and 8080, as in the following example.

```
$ rwfilter sample.rw --type=outweb                          \
--sport=80,443,8080 --protocol=6 --packets=4- --ack-flag=1 \
--sipset=web_servers.set                                    \
--pass=stdout                                               \
| rwuniq --fields=sip,sport
          sIP|sPort|   Records|
203.0.113.198|   80|       483|
203.0.113.197|   80|       473|
203.0.113.196|   80|         5|
203.0.113.194|  443|      2823|
 203.0.113.69|   80|      2567|
 203.0.113.69|  443|       200|
 203.0.113.44|   80|         2|
 203.0.113.44|  443|        83|
 203.0.113.28| 8080|         2|
```

The resulting table should look something like Table 6.

Table 6: Potential Web Servers for the Sample Network

Internal IP	Internal Port	External IP	External Port
203.0.113.198	80, 443	*	*
203.0.113.197	80	*	*
203.0.113.196	80	*	*
203.0.113.194	443	*	*
203.0.113.69	80, 443	*	*
203.0.113.44	80, 443	*	*
203.0.113.28	8080	*	*

This is a list of potential web servers that will have to be validated because some of the traffic to these servers may not really be web server traffic. It is also possible that some of the actual web servers on the network have not been detected in this step but will be detected in other steps in the profiling process. It is important to keep the profile table updated.

5.1.2 How to Validate Findings

Web servers should be easy to validate using a web browser. Manually go to each site, checking the address with both HTTP and HTTPS to verify that the associated ports in the table are correct. If possible, do this from outside the network, as the server may be configured to accept different types of traffic from the internal network, which could conflict with earlier findings. A

nonresponsive server could mean that the server is offline, it has been configured not to accept connections from certain networks, or it has been taken over for malicious purposes. Consider documenting a brief description of the services running on the server for future reference.

Resolve each IP address to a domain name using nslookup (reverse and forward) or some other tool and ensure that the result matches what is expected on the network. If it does, add the domain name as a field in the profile table. If it does not match, it may be an old server that no one realized was still running on the network, or it may be an anomaly. The reverse record may not match if the machine is in a shared hosting environment.

Check the server certificate of the servers associated with port 443. The common name on the certificate should match the server's domain name. If it does not match, there could be a security concern. Also, confirm that the certificate has not expired; this is a common cause of errors received by clients accessing sites over HTTPS that have not been properly maintained.

If it is still unclear which of these addresses are actually web servers, try looking at the external addresses connecting to them. If there are only a few, it could be a web server locked down to a few addresses, or possibly a point-to-point SSL VPN; but if there are many external sources, it probably is an actual web server. Examine the number of distinct external addresses, their address blocks, and the timing of requests to get a feel for whether the traffic is actual HTTP traffic. Some flow analysis tools attempt to label traffic based on content characteristics. This can be very helpful when trying to identify traffic that is not obvious based on header information.

Several of the IP addresses from the sample network listed as potential web servers were validated by using nslookup and by browsing to their addresses. Even though there is no traffic to port 443 on several of the servers, on others the port is open and presents an expired certificate. Also, some of the sites are using mirrors, such that multiple IP addresses are actually assigned to the same website. This is common for servers that receive heavy traffic.

One of the servers on the same network could not be validated using the above simple methods, so we took a closer look at the inbound TCP traffic to that address. Address 203.0.113.28 served only one IP address during the sample time frame as seen in the following example.

```
$ rwfilter sample.rw --type=outweb --sport=80,443,8080 --packets=4- \
--protocol=6 --ack-flag=1 --saddress=203.0.113.28 --pass=stdout     \
| rwuniq --fields=dip
          dIP|   Records|
198.51.100.12|         1|
```

In the example below, the records that resulted from the command show that the traffic from this external IP is composed of one very long flow. The flow is divided into half-hour chunks because that is how SiLK collects flows.

```
$ rwfilter sample.rw --type=outweb                          \
--sport=80,8080,443 --protocol=6 --packets=4- --ack-flag=1 \
--saddress=203.0.113.28                                     \
--pass=stdout                                               \
| rwcut --fields=stime,etime,bytes,flags
                  sTime|                  eTime|     bytes|   flags|
2011/09/28T00:15:32.577|2011/09/28T00:45:32.568|  68169846|   PA   |
2011/09/28T00:45:32.634|2011/09/28T01:15:32.567|  68272876|   PA   |
```

```
2011/09/28T01:15:32.609|2011/09/28T01:45:32.605|  68270589|   PA   |
2011/09/28T01:45:32.643|2011/09/28T02:15:32.640|  68252975|   PA   |
2011/09/28T02:15:32.646|2011/09/28T02:45:32.635|  68334416|   PA   |
2011/09/28T02:45:32.658|2011/09/28T03:15:32.653|  68310047|   PA   |
                    ...|                    ...|       ...|     ...|
2011/09/28T23:45:33.247|2011/09/29T00:15:33.218|  68245623|   PA   |
```

This is not typical traffic to a web server, so we deleted this IP address from the table of web servers. We show how to profile it in a later section.

5.1.3 Anomalies

The following is a list of anomalies that should be considered when determining which IP addresses are web servers.

- client traffic

  Servers normally do not have legitimate client traffic. Expected traffic would be from software updates or an administrator troubleshooting a problem on the server. If there is a web server with a lot of client traffic, it is probably some type of gateway and should go into a different category.

- streaming media services

  Some servers are meant for serving streaming media to clients. These servers have open ports that are different from the HTTP and HTTPS ports. Check for traffic coming into the network using UPD and TCP ports such as 1935, 1755, and 554.

- SSL VPNs

  Long-duration, high-volume connections on port 443 could be SSL VPN connections. More discussion is in Section 5.5 on VPNs.

- one server that sits on many IP addresses

  This situation should become clear upon resolving the IP addresses to DNS names. This is not a security issue, but it should be noted in the profile table.

- many websites that sit on one IP address

  This should also become clear when many IP addresses resolve to the same DNS name with a forward lookup (not necessarily a reverse lookup). Again, note this in the profile table.

- business continuity and server failover

  If the network has backup servers that are either turned off or are handling only small amounts of traffic until they are needed, the servers probably will not be observed in this section. Use the information gathered at the beginning of this exercise to list these servers in the profile table and note that they are backup servers.

- spurious RSTs

  Web servers often send reset packets to close a connection instead of waiting for the standard TCP FIN-ACK. This action is harmless and does not have an effect on profiling the asset.

- mobile devices and embedded web servers

  With today's technology, a surprising number of devices have web servers embedded in them; common devices include VOIP phones, mobile devices, print servers, and copiers.

5.1.4 Results

Sample network results from the validated web servers are in Table 7.

Table 7: Validated Web Servers for the Sample Network

Proto.	Internal Ports	External IPs	External Ports	External Names	Comments
203.0.113.198 – example.org					
6	80, 443	*	*	*	Redirects to st.example.org
203.0.113.197 – kirk.st.example.org					
6	80,443	*	*	*	Expired SSL certificate
203.0.113.196 – spock.example.org					
6	80, 443		*	*	Expired SSL certificate
203.0.113.194 – webmail.st.example.org					
6	443	*	*	*	Reverse DNS is misspelled
203.0.113.69 – www01.st.example.org					
6	80, 443	*	*	*	
203.0.113.44 – vss1.st.example.org					
6	80, 443	*	*	*	VMware View; DNS resolve directs to 203.0.113.198

5.2 Client Web

Client web traffic is usually the most common traffic on any network. For the purposes of this report, client web traffic will include HTTP, HTTPS, Shockwave, and other streaming media. It is important to separate web traffic from services like SSH, VPN, and file sharing because these services are managed very differently than web traffic.

The ports listed in Table 8 are the most common ports for web client traffic. You will find any additional web client protocols when you analyze leftover ports.

Table 8: Services for Normal Client Web Traffic

Protocol	Port
HTTP	80/TCP, 8000/TCP, 8080/TCP
HTTPS	443/TCP
Shockwave	1935/TCP
Microsoft Streaming Media	1755/UDP, 1755/TCP, 554/TCP, 554/UDP
HTTP Live Streaming	Standard HTTP ports

5.2.1 The Process

The process for finding web clients is similar but opposite to the process for finding web servers. Instead of looking for traffic coming from web ports on the internal host, look for traffic going to web ports on an external host, as shown in Figure 8.

Figure 8: Process for Finding Web Clients

1. Start by filtering for legitimate TCP traffic on the ports in Table 8, as shown in the following command. The traffic should be outbound to the ports on IP protocol 6 (TCP) with at least four packets per flow and the ACK flag set.

```
$ rwfilter sample.rw --type=out,outweb  \
--protocol=6 --ack-flag=1 --packets=4-  \
--dport=80,8000,8080,443,1935,1755,554  \
--pass=stdout                           \
| rwset --sip-file=tcp_clients.set
```

2. Add in the UDP traffic to the ports in Table 8 using the following command.

```
$ rwfilter sample.rw --type=out,outweb \
--protocol=17 --dport=1755,554         \
--pass=stdout                          \
| rwset --sip-file=udp_clients.set
```

3. Listing legitimate TCP-only (flows of four packets or more with an ACK flag) and UDP flows separately ignores some scanning traffic. Looking at outbound rather than inbound traffic also

helps with this. After both lists are created, combine them for analysis, as in the following example.

```
$ rwsettool --union tcp_clients.set udp_clients.set \
--output-path=web_clients.set
$ rwsetcat --count web_clients.set
31
```

The output from the above command shows that the sample network has 31 web clients. Look at only the hosts with at least 1% of the client web traffic on the network because most hosts will act as web clients at some point. Even DNS, web, and other servers often have some web traffic from updates or configuration changes; but this traffic is usually minimal. When using a proxy, there should be very few internal IPs that fall into this category because the client traffic will be funneled through these devices. If this is not the case, it may be useful to try profiling the workstations separately from other hosts (separate them based on traffic volume).

4. The bulk of the packets for client web traffic will be inbound. Therefore, filter incoming traffic if possible.

```
$ rwfilter sample.rw --type=in,inweb                            \
--dipset=web_clients.set --sport=80,8080,8000,443,1935,1755,554 \
--pass=stdout                                                   \
| rwstats --fields=dip --bytes --percentage=1
INPUT: 2973484 Records for 31 Bins and 168273056846 Total Bytes
OUTPUT: Top 2 bins by Bytes (1% == 1682730568)
          dIP|               Bytes|    %Bytes|   cumul_%|
 203.0.113.33|        154723380928| 91.947804| 91.947804|
203.0.113.220|         12334655077|  7.330143| 99.277947|
```

The above command results in a list of two IP addresses accounting for 99.27% of the client web traffic. Update the profile to include only the results from filtering incoming traffic.

5. If it is not possible to filter inbound traffic, use the following filter on outbound traffic. This filter checks traffic going from the web clients to external web servers, rather than traffic coming from external web servers to the internal web clients as the previous filter does.

```
$ rwfilter sample.rw --type=out,outweb                          \
--sipset=web_clients.set --dport=80,8080,8000,443,1935,1755,554 \
--pass=stdout                                                   \
| rwstats --fields=sip --percentage=1
INPUT: 2921867 Records for 32 Bins and 2921867 Total Records
OUTPUT: Top 2 bins by Records (1% == 29218)
          sIP|   Records|  %Records|   cumul_%|
 203.0.113.33|   2625228| 89.847621| 89.847621|
203.0.113.220|    274448|  9.392898| 99.240520|
```

6. Next, associate each IP address with the ports and protocols with which it is interacting using the command in the following example, which also shows the output. Place these addresses and their associated ports into the profile table.

```
$ rwfilter sample.rw --type=out,outweb                          \
--sipset=web_clients.set --dport=80,8000,8080,443,1935,1755,554 \
--pass=stdout                                                   \
| rwuniq --fields=sip,dport,protocol
          sIP|dPort|pro|   Records|
203.0.113.220|   80|  6|    235334|
203.0.113.220|  443|  6|     38733|
203.0.113.220|  554|  6|         5|
203.0.113.220| 8080|  6|        95|
203.0.113.220| 1935|  6|       281|
 203.0.113.33| 1935|  6|       128|
 203.0.113.33|  443|  6|    251473|
 203.0.113.33| 8000|  6|         1|
 203.0.113.33| 8080|  6|       895|
 203.0.113.33|   80|  6|   2372731|
```

The sample network has web clients as shown in Table 9, which were added to the full profile.

Table 9: Potential Web Clients for the Sample Network

Internal IP	Internal Port	External IP	External Port	Protocol
203.0.113.220	*	*	80, 443, 554, 1935, 8080	6
203.0.113.33	*	*	80, 443, 1935, 8000, 8080	6

Follow the next section to carefully validate this list of potential web clients.

5.2.2 How to Validate Findings

Web clients are more difficult to validate than web servers. Look for three types of web clients: proxy servers, NAT gateways, and directly connected workstations that do not use a proxy or NAT server. This requires looking at inbound traffic as well as outbound. Some of the addresses in the list might not be web clients at all.

The different types of web clients can be determined based on several characteristics. The first are traffic volume and timing. Directly connected workstations typically have traffic patterns that match the schedule of the organization. As with the sample network, many organizations have no after-hours or weekend traffic, and their daily traffic volume tends to be very small. Web gateways (NAT servers) have a high volume of traffic. Web proxy servers have a medium to high volume of traffic.

For the sample network, the traffic volume for the web clients is as follows.

```
$ rwfilter sample.rw --type=out,outweb                          \
--sipset=web_clients.set --dport=80,8080,8000,443,1935,1755,554 \
--pass=stdout                                                   \
| rwstats --fields=sip --bytes --count=2
INPUT: 3029615 Records for 15228 Bins and 168278493869 Total Bytes
OUTPUT: Top 2 Bins by Bytes
          sIP|               Bytes|    %Bytes|   cumul_%|
 203.0.113.33|       154723380928| 91.944833| 91.944833|
203.0.113.220|        12334655077|  7.329906| 99.274739|
```

Based on traffic volume alone, one can make an educated guess that the top IP address (203.0.113.33) is a web gateway and the other is a web proxy server.

Another characteristic to check is whether the client is running any other services. Client workstations usually do not have any services running. Web gateways are likely to look as if they host services because they are often gateways for the rest of the services on the network. A client that looks as if it is hosting many services could also be a VPN gateway. Web proxies are not as likely to host other services unless they come with services configured by default.

To keep the list size manageable, look for common services such DNS, FTP, SSH, SMTP, HTTP, and HTTPS. The following output shows the command and results for looking for common services.

```
$ rwfilter sample.rw --type=out,outweb                          \
--sipset=web_clients.set --sport=20,21,22,25,53,80,8000,8080,443 \
--pass=stdout                                                   \
| rwuniq --fields=sip,sport
          sIP|sPort|   Records|
203.0.113.220| 8080|         7|
203.0.113.220| 8000|         7|
 203.0.113.33|   20|         1|
 203.0.113.33|  443|       492|
 203.0.113.33|   21|        27|
 203.0.113.33| 8000|        44|
 203.0.113.33| 8080|        58|
 203.0.113.33|   22|         7|
 203.0.113.33|   80|        42|
```

Because 203.0.113.33 makes connections over so many ports, as the example's output shows, there is a good chance that it is actually a multipurpose gateway or a VPN gateway. We removed it from the web clients list and show how to profile it later in this report.

There may be a fair amount of high port traffic, indicating that clients are using various host-based applications or streaming media. Also, most client operating systems use ephemeral ports in a sequential fashion within a certain range. For example, if most clients on the network are Windows 7 machines that use the range 49152 to 65535, look for hosts using ports starting on the lower end to make connections and working their way up to port 65535.

If plotting software is available, it may help to view ephemeral port usage over time. This will allow you to quickly see sequential port usage and what range is being used. This is a method often referred to as "passive fingerprinting."

After we checked traffic volume, services, and port usage of the potential web clients on the sample network, we validated the one client left on the list as an actual web proxy server.

5.2.3 Anomalies

Almost all machines at some point generate a small amount of web traffic, whether they are servers, gateways, or local hosts. Client web traffic coming from machines that serve another purpose can be ignored because it does not need to be part of the profile. If this traffic is of

concern for security reasons, look into the flows in more detail, or check firewall/proxy logs and configurations.

Here is a more detailed description of some of the anomalies mentioned in Section 5.2.2 about validation.

- directly connected workstations hosting services

  Client operating systems come with certain services preinstalled. Sometimes those services are left on by default. So, just because services are hosted at an address does not necessarily indicate that the machine is a server or gateway. Look at the specific service being hosted and the amount of traffic connecting to it.

- servers with client web traffic

  Though servers sometimes use web traffic to get their updates or configuration changes, this traffic is usually minimal and received from a small number of external addresses. If this is the only client web traffic to an address in the list, that address probably should not be listed as a web client.

- other types of traffic over web ports

  Just because there is traffic at the typical web ports does not mean it is HTTP or streaming traffic. Any service can be configured to use those ports. This is often done for application ease of use or to get around firewall policies.

5.2.4 Results

The sample network had one web client that was added to the profile as listed in Table 10.

Table 10: Final Web Clients for the Sample Network

Proto.	Internal Port	External IP	External Port	External Name	Comments
203.0.113.220					
6	*	*	80, 443, 554, 1935, 8080	*	Web gateway

5.3 Email

Email services are provided through several different protocols. The most common protocol for sending mail across the internet is Simple Mail Transfer Protocol (SMTP) on TCP port 25. SMTP traffic is mostly server-to-server traffic. Email clients, desktop clients in particular, do not use SMTP but instead submit emails to the SMTP server at port 587 to be forwarded by the server. To receive messages, clients use Post Office Protocol (POP), Internet Message Access Protocol (IMAP), or web mail.

Email protocols allow for encryption over the standard TCP ports outlined above. However, some additional ports are set aside for encrypted email sessions. These are less commonly used but may still show up on networks. Email ports are listed in Table 11.

Table 11: Email Ports and Protocols

Protocol	Port	Encrypted Session Port
SMTP	25	465
POP3	110	995
IMAP	143	993
MSA	587	None

5.3.1 The Process

1. Using the following command, start by looking for email servers, which will send traffic from the SMTP, POP3, and IMAP ports listed in Table 11. Look for outgoing traffic with a source TCP equal to those ports and an IP protocol of 6. TCP port 587 (MSA) is not used here because it is only used with inbound traffic to servers.

```
$ rwfilter sample.rw --type=out                                          \
--protocol=6 --ack-flag=1 --packets=4- --sport=25,465,110,995,143,993 \
--pass=stdout                                                            \
| rwset --sip-file=smtp_servers.set
```

The results of the above command on the sample network are listed in Table 12.

Table 12: Potential Email Servers for the Sample Network

Internal IP	Internal Port	External IP	External Port
203.0.113.231	25	*	*
203.0.113.195	25	*	*
203.0.113.221	25	*	*
203.0.113.220	25	*	*
203.0.113.222	25	*	*

Because the sensors for the sample network are located on the internet-facing side of the network, no IMAP or POP traffic is seen between internal clients and servers. Only server SMTP traffic is seen by the sensor.

2. Now look for potential email clients. Depending on where the sensors are placed, client traffic may not cross the flow collectors because it only has to go as far as the mail server on the

network. This a good test to see if any clients are sending email directly outside of the network without using standard mail services.

3. Desktop email clients typically submit messages to TCP port 587 and collect them from TCP ports 110 and 143 (or encrypted ports 995 and 993), so filter outbound traffic for these ports using the following command.

```
$ rwfilter sample.rw --type=out                                        \
--protocol=6 --packets=4- --ack-flag=1 --dport=110,143,587,995,993 \
--pass=stdout                                                          \
| rwset --sip-file=email_clients.set
```

The sample network did not have any email clients visible in the day-long sample data set, a strong indicator that its internal hosts are correctly set up to get their (corporate) email through the network's email servers. It could also mean that hosts are getting all their email through web mail.

5.3.2 How to Validate Findings

Once again, it is important to validate the results for the email servers and clients. Potential servers sending mail out of the network to TCP port 25 are almost guaranteed to be servers. This can be thought of as an email server's way of acting as client to other servers on the internet. Filtering for this traffic on the sample network validates two addresses. The following example shows the command results.

```
$ rwfilter sample.rw --type=out                       \
--protocol=6 --packets=4- --ack-flag=1 --dport=25,465 \
--pass=stdout                                         \
| rwuniq --fields=sip
            sIP|   Records|
  203.0.113.222|     10479|
  203.0.113.231|     74852|
```

Servers can also be validated using nslookup for each of the addresses in the list or using the MX record of the domain. Obtain the MX record by typing the following command:

```
nslookup –type=MX example.org
```

The result should be a list of servers, like the one below, that relay email to and from the domain.

```
Non-authoritative answer:
example.org        mail exchanger = 10 checkov.example.org.
example.org        mail exchanger = 20 sulu.example.org.
```

Use this list of mail exchanger servers as a guide to help with validation. Using a regular DNS lookup on each address in the server list can help you validate simply by looking at the name of the server (all externally facing SMTP servers will have a DNS record and associated IP address). Telnetting to each IP address at port 25 may also produce a banner message hinting at the purpose of a machine. These messages sometimes contain error information that is specific to a particular operating system or service running on the remote machine.

As a result of using these methods, all but one of the addresses was verified as an email server—203.0.113.220, which was listed in the previous section as a gateway for web client traffic. It

looks as if this address may also be proxying email traffic, which indicates that it is probably a main gateway for all client traffic.

Email servers almost always come with a default web mail application. This means that these email servers are also basically acting as web servers. Unfortunately, these web servers are not usually discovered when filtering for web servers (as done in Section 5.1) because of their low traffic volume. After the email servers have been validated, test each one for a running web mail application by going to each server's address in a web browser. Try this on both ports 80 and 443.

As is often the case, the sample network was running web applications on two of its email servers. Both were open to connections over HTTP, meaning email login credentials were being sent in plain text.

It is slightly more difficult to validate client email traffic because there is not usually any external information to use in the validation. Try searching the web for a particular client address to see if it shows up in any email headers. Also, try looking at the traffic in more detail to see if it matches expected client traffic.

5.3.3 Anomalies

Email service is probably the most straightforward service discussed in this report. However, there are some anomalies to be aware of.

- nondedicated email servers

  Email servers do not necessarily have to be dedicated machines. They can, and often do, run in parallel with other services, especially web services. So, it is acceptable if an IP address of an email server has already been listed as a web server because it is just one machine running several different services.

- clients acting as servers

  Because email services do not have to be on dedicated servers, any host can run an email server, even a client. If an IP address in a client subnet is listed as a server or an IP address is listed as both a server and a client, this is probably what is happening. These servers will likely have a very low SMTP traffic volume compared to the other servers. Check policies on clients hosting such services and firewall configurations. (On most networks, only certain IPs should be allowed to send and receive SMTP traffic.)

- desktop clients sending SMTP messages

  It is possible for client machines to be configured to send their messages to port 25 (SMTP) rather than port 587 (MSA). Investigate any clients that bypass the internal email infrastructure by sending messages directly to the internet.

5.3.4 Results

The sample network has four SMTP mail servers and no externally facing mail clients, as listed in Table 13.

Table 13: Validated Email Assets for the Sample Network

Proto.	Internal Port	External IP	External Port	External Name	Comments
203.0.113.231 - smtpfw.st.example.org					
6	25	*	*		
6	*	*	25		
203.0.113.195 - sulu.example.org					
6	25	*	*		
6	*	*	25		
203.0.113.221 - omail.example.org					
6	25	*	*		
6	*	*	25		
6	80	*	*		Plain-text email login
203.0.113.222 – imail.example.org					
6	25	*	*		
6	*	*	25		
6	80	*	*		Plain-text email login

5.4 Domain Name System

The Domain Name System (DNS) provides multiple services [Mockapetris 1987]. The most common is the translation between domain names and IP addresses. A network will likely contain both DNS clients and DNS servers. The standard port for DNS is 53, usually carried over UDP. Transactions may also take place over TCP when larger amounts of data need to be transferred, such as when zone transfers occur. However, zone transfers rarely take place across perimeter boundaries. On a network, the servers will be the hosts accepting connections on port 53, and the clients will be those communicating with services on port 53.

Servers support either recursive or iterative query resolution. Recursive servers take the responsibility of looking up the complete answer for the client, recurring, if necessary, from the root (".") to the top-level domain (".com") to the second-level domain ("example.com") and so forth, and only then responding to the client's request. Recursive servers will typically communicate with numerous other servers across the internet. Iterative servers, on the other hand, accept requests but do not do recursive resolution. If an iterative server has the information necessary to answer the request, it gives that information to the client. Otherwise, it responds with an error or a referral to another DNS server. The client is then responsible for querying the next DNS server. This means that you should never see iterative servers issuing DNS requests.

5.4.1 The Process

1. Begin the DNS inventory with a summary of the DNS traffic on the network. Look for the top DNS (port 53) traffic inbound and outbound and the top external addresses receiving the network's DNS traffic. Sort traffic volumes by the number of packets rather than the number of bytes or flows because almost every DNS request or response is contained in a single UDP packet.

The top 10 external DNS servers that were queried by the sample network using the following command are shown in the output below. Their IP addresses have been converted to domain names using SiLK's built-in resolver function.

```
$ rwfilter sample.rw --type=out             \
--dport=53 --protocol=17                    \
--pass=stdout                               \
| rwstats --count=10 --fields=dip --packets \
| rwresolve
INPUT: 1530863 Records for 12559 Bins and 1629508 Total Packets
OUTPUT: Top 10 Bins by Packets
                    dIP|   Packets|  %Packets|   cumul_%|
dns.publicprovider.com|    533435| 32.735955| 32.735955|
             192.0.2.1|    173289| 10.634437| 43.370392|
          provider.net|    113603|  6.971614| 50.342005|
    d.gtld-servers.net|     19974|  1.225769| 51.567774|
    a.gtld-servers.net|     18015|  1.105548| 52.673322|
    c.gtld-servers.net|     16703|  1.025033| 53.698356|
    l.gtld-servers.net|     16620|  1.019940| 54.718295|
    b.gtld-servers.net|     15752|  0.966672| 55.684967|
    f.gtld-servers.net|     15657|  0.960842| 56.645810|
    g.gtld-servers.net|     15378|  0.943720| 57.589530|
```

In the sample network, the highest numbers of resolution requests go to a public, open resolver, and most of the other top destinations are root name servers. Make sure these domains conform to network policy.

2. The next step is to find the assets on the network acting as DNS servers. Look for outbound traffic *from* port 53 over UDP using the following command. Pull only those IPs that are sending at least 1% of the filtered traffic; these are the most likely to be actual DNS servers. You will deal with the rest at the end of the profile.

```
$ rwfilter sample.rw --type=out                                  \
--sport=53 --protocol=17                                         \
--pass=stdout                                                    \
| rwstats --fields=sip --percentage=1 --packets --no-titles      \
| cut -f 1 -d "|"                                                \
| rwsetbuild > dns_servers.set
$ rwsetcat dns_servers.set
203.0.113.50
203.0.113.51
203.0.113.52
```

The result in the above example is a list of DNS servers on the sample network.

3. Next, find DNS clients by looking at outbound traffic *to* port 53 over UDP using the following command. Again, pull only the clients making up at least 1% of the DNS client traffic. If there is a long list of DNS clients making up less than 1% of the total DNS client traffic, it is likely that the hosts on the network are not configured to go to the local DNS server and/or are not making their requests through the gateway.

```
$ rwfilter sample.rw --type=out                                  \
--dport=53 --protocol=17                                         \
--pass=stdout                                                    \
| rwstats --fields=sip --percentage=1 --packets --no-titles      \
| cut -f 1 -d "|"                                                \
| rwsetbuild > dns_clients.set
$ rwsetcat dns_clients.set
203.0.113.222
203.0.113.220
203.0.113.33
203.0.113.51
203.0.113.52
```

You now have a list of DNS clients, as shown in the results of the above example, and a list of DNS servers. It may be that one or more IP addresses are in both lists; these are the recursive name servers. Recursive name servers act both as DNS clients and servers as illustrated in Figure 9 because they receive and issue requests.

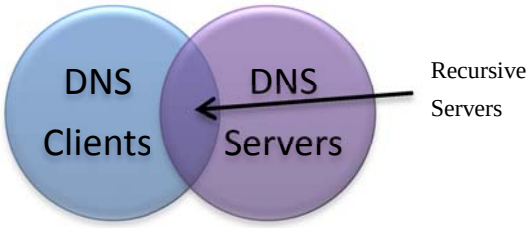

Figure 9: Recursive DNS Servers

4. Identify recursive servers by looking at the intersection of the DNS client and server lists, as in the following example.

```
$ rwsettool --intersect dns_clients.set dns_servers.set \
--output-path=recursive.set
$ rwsetcat recursive.set
203.0.113.51
203.0.113.52
```

5. The last step in finding DNS assets is to identify any potential authoritative name servers. Authoritative servers are iterative, so find the difference between the list of DNS servers and recursive DNS servers to get the list of iterative servers. The following example shows the command and results.

```
$ rwsettool --difference dns_servers.set recursive.set \
--output-path=iterative.set
$ rwsetcat iterative.set
203.0.113.50
```

Just because a server is iterative does not necessarily mean it is authoritative. This will need to be validated in the next section. The sample network has one potential authoritative server, as shown in Table 14.

Table 14: Potential DNS Assets for the Sample Network

Internal IP	Internal Port	External IP	External Port	Comments
203.0.113.33	*	*	53	Client
203.0.113.50	53	*	*	Iterative & Authoritative
203.0.113.51	53	*	*	Recursive
	*	*	53	
203.0.113.52	53	*	*	Recursive
	*	*	53	
203.0.113.220	*	*	53	Client
203.0.113.222	*	*	53	Client

5.4.2 How to Validate Findings

It generally is not necessary to validate the DNS client list. Using the following command, check that the hosts listed as DNS clients create at least 1% of the total DNS client traffic. Assets with less than 1% of the traffic will likely be profiled as web clients or other types of servers.

```
$ rwfilter sample.rw --type=out     \
--protocol=17 --dport=53            \
--pass=stdout                       \
| rwstats --fields=sip --percent=1 --packets
INPUT: 1530662 Records for 13 Bins and 1629113 Total Packets
OUTPUT: Top 5 bins by Packets (1% == 16291)
            sIP|   Packets|  %Packets|   cumul_%|
  203.0.113.33|    550990| 33.821472| 33.821472|
  203.0.113.51|    460194| 28.248133| 62.069605|
  203.0.113.52|    381179| 23.397947| 85.467552|
 203.0.113.222|    120051|  7.369102| 92.836654|
 203.0.113.220|    106669|  6.547673| 99.384328|
```

The results from the above example show that the DNS client list for the sample network is accurate; however, remember that addresses 203.0.113.33 and 203.0.113.220 were profiled in the web client section as a gateway and a proxy server, respectively. Both addresses will be removed from the DNS list because they are simply proxying requests from other hosts instead of acting as clients themselves. Address 203.0.113.222 has already been profiled as an email server (from Section 5.3).

To validate the servers, resolve each of the addresses in the list to a domain name using one of the tools discussed in previous sections (nslookup, dig, robtex.com, rwresolve). Often, the servers have names beginning with “ns” or “dns,” indicating that they are name servers. The results of using nslookup on the potential DNS server addresses .50, .51, and .52 are domain names that start with ns, ns1, and ns2, respectively.

Also use nslookup to view what servers are being advertised to the internet as name servers.

```
$ nslookup -type=ns example.org
...
Non-authoritative answer:
example.org        nameserver = ns2.example.org.
example.org        nameserver = ns1.example.org.

Authoritative answers can be found from:
example.org        nameserver = ns.example.org.
```

It is possible that not all of the DNS servers on the network will show up in an nslookup query, but the ones that do can be validated.

5.4.3 Anomalies

Below are some anomalies you may find while examining DNS flows.

- port 53 to 53 traffic

  Older implementations of DNS used both source port and destination port of 53 for server-to-server requests. A number of attacks are based on the ability of the attacker to inject fake responses [US-CERT 2008]. Randomizing request source port numbers is one way to make this more difficult and is standard practice; however, port-53-to-53 traffic still exists.

- assets or hosts other than the ones listed that make or respond to DNS requests

  Sometimes assests on a network make their own requests to DNS servers rather than go through the local DNS server. The asset typically has a very low traffic volume to port 53

and only makes requests to one specific server. This behavior is necessary when there is no local DNS server available. However, if the network is set up so that clients have access to a local DNS server, this behavior may indicate malicious acitivity or possibly a policy violation.

Unauthorized servers, on the other hand, should be of concern. If you see DNS responses from hosts in a network other than the network's DNS servers, it could be the result of malcious activity, a policy violation, or misconfiguration at the firewall or host.

- unexpected destination servers

  Although these servers are not necessarily an anomaly, consider looking at a list of destination servers to make sure the DNS traffic is going where expected. If clients are configured to query specific external servers for security or speed, check that the traffic is going only to those servers. Also, check the destination servers against blacklists if necessary.

- multiple servers or clients at one IP address

  It is possible for one physical server to act as both a recursive and an authoritative server. For example, one IP address could have a recursive server for processing queries from internal hosts and an authoritative server for external requests. In these cases, it may be impossible to tell from netflow data alone whether the server is recursive or iterative and which clients it is servicing. Simply note that it is a DNS server.

5.4.4 Results

We used the above process to identify three DNS servers on the sample network. Two of the DNS servers are recursive and one is iterative, as shown in Table 15.

Table 15: Final DNS Assets for the Sample Network

Proto.	Internal Port	External IP	External Port	External Name	Comments
203.0.113.50 – ns.st.example.org					
17	53	*	*		Iterative, Authoritative
203.0.113.51 – ns1.st.example.org					
6,17	53	*	*		Recursive
6,17	*	*	53		
203.0.113.52 – ns2.st.example.org					
6,17	53	*	*		Recursive
6,17	*	*	53		

5.5 Virtual Private Networks

Virtual Private Network (VPN) technology varies, and servers can be easy or difficult to identify within netflow traffic depending on the technology used. For the purposes of this report, VPNs are defined as tunnels over public networks, which allow remote offices and users to connect to the main office network as if they are physically in the home office. VPNs are sometimes referred to as VPN tunnels because once a VPN connection is established, any type of traffic can be carried over that connection; it is impossible to tell at a glance what protocol is being carried because the packet is always encapsulated within a VPN wrapper. Often, VPNs use a cryptographic protocol such as SSL or IPsec to ensure the privacy of traffic. In addition, it is possible to layer tunneling protocols inside each other, which increases the difficulty of monitoring.

Typically, VPNs have an initial setup phase and a tunneling phase. This is somewhat similar to FTP traffic, which uses a control channel and a data channel. The initial setup phase is meant to allow VPNs to be initialized even when the destination is behind Network Address Translation (NAT) hardware. Because the initial setup uses TCP and UDP ports, the NAT box is able to translate ports to IP addresses. The tunneling phase uses other IP protocols, such as ESP or GRE, which usually do not have any ports associated with them. These two phases make VPN traffic fairly distinct from other netflow traffic.

VPNs can be created using a number of different protocols based on the needs of the organization and the vendor being used. As shown in Table 16, this report includes the ports and protocol numbers of the VPN technologies PPTP, L2TP, SSL, and IKEv2, but this report will not discuss how each one specifically works. Many vendors may have their own specific implementation that uses other ports or protocols. For example, Cisco is a popular VPN implementation that can use any pre-defined TCP port for IPSEC over VPN[7]. The default is 10000. If you know that the network being profiled uses a different protocol or set of ports, include those here as well.

Table 16: VPN Technologies

	Initial Setup		Tunnel	
	Protocol	**Ports**	**Protocol**	**Ports**
PPTP	6	1723	47	N/A
L2TP	17	500, 1701, 4500	50	N/A
SSL	6	443	6	443
IKEv2	17	500	50, 51	N/A

VPNs can be broken into two types: remote access and point-to-point. Remote access VPNs allow remote users to connect to the internal network. Requests from and responses to these external sources are wrapped in a VPN header that will always be addressed directly to (or from) the VPN gateway rather than the internal resource the user actually needs. Connections to a remote access VPN are from many different addresses (often ISPs) and usually last for the length of time the user needs the connection—about a few hours.

[7] http://www.cisco.com/en/US/products/hw/vpndevc/ps2030/products_configuration_example09186a0080645722.shtml

For point-to-point VPNs, the connection is always set up between the same two points (often a remote office and main office). There can be multiple point-to-point connections to one VPN gateway, but they will be long-term connections, and the external addresses will rarely change.

Once a VPN connection terminates inside of the network, the traffic from that connection will look like all other internal network traffic. For example, if a client that is VPNed into the network starts to browse the web, this activity will show up in the regular web client traffic that was profiled in Section 5.2.

5.5.1 The Process

1. Start by finding the assets on the network using the protocols 47, 50, and 51 using the following command. Look only at outbound traffic.

```
$ rwfilter sample.rw --type=out   \
--protocol=47,50,51               \
--pass=stdout                     \
| rwuniq --fields=sip --no-titles \
| cut -f 1 -d "|"                 \
| rwsetbuild > vpn.set
$ rwsetcat vpn.set
203.0.113.33
203.0.113.35
203.0.113.178
```

In most cases, the results from this query will be the correct VPN gateways on the network.

2. Next, look for the ports being used by the assets you found in the preceding step. Filtering outbound traffic from each address for a source port equal to any of those listed in Table 16 will show which ports the VPN is using. This filter is shown in the following example. Also note the protocol and destination port associated with each source port.

```
$ rwfilter sample.rw --type=out                                  \
--saddress=203.0.113.33 --sport=500,4500,1701,1723,10000         \
--pass=stdout                                                    \
| rwstats --fields=sport,protocol,dport --bytes --percentage=1
INPUT: 822 Records for 46 Bins and 1661747093 Total Bytes
OUTPUT: Top 10 bins by Bytes (1% == 16617470)
sPort|pro|dPort|               Bytes|    %Bytes|   cumul_%|
 4500| 17| 4500|           738166761| 44.421125| 44.421125|
 4500| 17| 4476|           292362688| 17.593693| 62.014819|
 4500| 17|43648|           220907925| 13.293715| 75.308534|
 4500| 17|31780|            68069706|  4.096274| 79.404807|
 4500| 17| 1603|            64275393|  3.867941| 83.272748|
 4500| 17|31281|            48482198|  2.917544| 86.190292|
 4500| 17| 1082|            47065498|  2.832290| 89.022582|
 4500| 17|36348|            39271744|  2.363280| 91.385862|
 4500| 17| 1079|            37549575|  2.259644| 93.645507|
 4500| 17| 1663|            29137025|  1.753397| 95.398904|
```

The output in the above example shows that the VPN for the sample network uses port 4500 for the majority of its VPN connections, meaning it is likely using L2TP. Repeat this filter for each VPN asset.

3. Now determine whether each VPN is remote access or point-to-point. Looking at the number of VPN connections may not help in determining this because some sites may have multiple remote offices connecting into the VPN gateway. Instead, look at the duration of the flows. In SiLK, the simplest way to do this is to look at the unique destination IP addresses, as shown in the following example. By default, SiLK will list the number of flow records associated with that address for the given time frame. When SiLK collects flows longer than 30 minutes, it cuts the flows into 30-minute records, so 48 consecutive records actually constitute a 24-hour long flow.

```
$ rwfilter sample.rw --type=out             \
--saddress=203.0.113.33 --protocol=47,50,51 \
--pass=stdout                               \
| rwuniq --fields=dip --no-titles           \
| sort -nrt '|' -k 2
            dIP|   Records|
   74.94.205.65|         48|
   75.144.2.201|         48|
   74.94.207.85|         48|
    75.144.0.93|         48|
...                     ...
  75.145.32.145|         48|
...                      ...
   72.61.11.246|          2|
 68.246.188.197|          1|
 174.147.23.130|          1|
108.117.128.105|          1|
[…55 total…]
```

In the above example, the Unix “sort” command was used to sort the output by the second column in descending order. Almost all the VPN connections for 203.0.113.33 last all day. Based on the length of the traffic flows, the sample network has about 49 site-to-site connections and is only acting as remote access for a handful of external connections.

VPNs using SSL are much harder to detect with netflow because they use the standard HTTPS port and protocol and, therefore, look like web servers. If all of the potential web servers from Section 5.1 were validated, none would be SSL-based VPNs. If there were some that were not validated or were questionable, they are possibly SSL VPNs. If there are very long, high-volume connections to port 443, there is reason to suspect that the server could actually be a VPN gateway.

The resulting VPNs for the sample network are listed in Table 17.

Table 17: Potential VPN Gateways for the Sample Network

Internal IP	Protocol	Internal Port	External IP	External Port	Comments
203.0.113.33	50 (17)	0 (4500)	*	0 (*)	50 S2S, remote access
203.0.113.35	50	0 (500)	*	0 (500)	1 S2S, remote access
203.0.113.178	50	0 (10000, 4500)	*	0 (10000, *)	1 S2S, remote access

5.5.2 How to Validate Findings

Assets using IP protocols 47, 50, or 51 are very likely to be VPNs. To verify VPN gateways, look up the domain names of the assets in the list using the following command.

```
nslookup <IP_Address>
```

Looking up the three potential VPNs for the sample network resulted in domain names for two of the addresses. Both include “vpn” in their name.

For assets that cannot be validated using domain name lookup, try taking a quick look at traffic patterns. VPN connections typically start with a brief (a few seconds) connection setup using one of the ports listed under the “Initial Setup” header in Table 16. The actual VPN session takes place over a longer period of time over one of the IP protocols listed under the “Tunnel” header in Table 16. Also, VPN sessions usually need to be renewed after a period of inactivity (usually 30 minutes), or a certain period of active use.

Because address 203.0.113.35 from the sample network could not be validated using nslookup, we looked more closely at the traffic going to and from that address. The command and results are shown in the following example. The flows are sorted by time.

```
$ rwfilter sample.rw --type=all          \
--any-address=203.0.113.35 --aport=0,500 \
--pass=stdout                            \
| rwsort --fields=sTime                   \
| rwcut --fields=protocol,sip,sport,dip,dport,stime,etime
pro|            sIP|sPort|             dIP|dPort|  sTime|  eTime|
 50|   192.0.2.170|    0|     203.0.113.35|    0|  00:00|  00:25|
 17|  203.0.113.35|  500|      192.0.2.170|  500|  00:20|  00:20|
 17|   192.0.2.170|  500|     203.0.113.35|  500|  00:20|  00:20|
 50|  203.0.113.35|    0|      192.0.2.170|    0|  00:26|  00:53|
 50|   192.0.2.170|    0|     203.0.113.35|    0|  00:30|  00:53|
 50|  203.0.113.35|    0|      192.0.2.170|    0|  00:56|  01:25|
 50|   192.0.2.170|    0|     203.0.113.35|    0|  01:00|  01:25|
 17|  203.0.113.35|  500|      192.0.2.170|  500|  01:20|  01:20|
 17|   192.0.2.170|  500|     203.0.113.35|  500|  01:20|  01:20|
 50|  203.0.113.35|    0|      192.0.2.170|    0|  01:26|  01:53|
 ...              ...                ...                   ...
```

This traffic shows the pattern described above: short negotiations over UDP and a long VPN session over protocol 50 that is renewed every hour.

5.5.3 Anomalies

Some anomalies associated with VPNs are listed below.

- VPN colocated with other services

  Just as with other services, VPN services can be located on the same piece of hardware (and the same IP address) as another service. It is common to find VPN gateways colocated with web gateways or proxy servers, which happened with 203.0.113.33 on the sample network.

- VPN gateway as a concentrator and site-to-site

  There is no reason to have separate hardware for these two types of VPN tunnels. If the network has both types, they will likely be located on the same machine.

- VPNs over nonstandard ports

  There are a wide variety of VPN solutions from various vendors, all using different ports, and sometimes different protocols, to establish connections. Also, anyone can choose a

different port to use, as long as the other end of the connection knows which port to use as well. These nonstandard ports often have to be manually configured to be allowed through the firewall.

- varying timeouts

  VPNs commonly use three timeouts: the timeout for establishing a connection, the inactive timeout, and the active timeout. Different vendors may choose to implement different default times, and the timeouts can usually be configured manually at the client or concentrator. Depending on how many types of VPN software are on the network, and whether clients are allowed to configure their own timeouts, varying timeouts may be seen across VPNs. For example, the U.S. Government requires active timeouts to be set at 30 minutes for any of its networks [OMB 2006]. Regardless of the timeout length, there will still be a regular pattern within each VPN flow.

5.5.4 Results

One high-volume VPN server (203.0.113.33) was found in the sample network, along with two lower volume VPN servers.

Table 18: Validated VPN Assets for the Sample Network

Proto.	Internal Port	External IP	External Port	External Name	Comments
203.0.113.33 - webvpn.example.org					
50	0	*	0		50 S2S, remote access
17	4500	*	*		
203.0.113.35					
50	0	*	0		1 S2S, remote access
17	500	*	500		
203.0.113.178 – vpn.st.example.org					
17	4500	*	*		
17	10000	*	10000		
50	0	*	0		1 S2S, remote access

5.6 Remote Services

This section discusses how to profile three remote service protocols: Telnet, SSH, and FTP. Telnet is an older protocol running on TCP port 23 that should no longer be in use today because of its lack of security [Postel 1983]. The Secure Shell (SSH) protocol provides secure remote login capabilities over TCP at port 22 [Lonvick 2006]. It often provides security services for FTP (called SFTP).

Although FTP is not used as much today as it was 10 years ago, it is still used to transfer large files between entities, to back up hosts to central (or external) servers, and to upload configuration files to servers [Postel 1985]. An FTP connection starts with a control connection from the client connecting to the server over TCP at port 21. Then a separate data connection is established over TCP at a different port. If the mode is set to active, the server initiates the data connection from its port 20 to an ephemeral port on the client. Otherwise, the mode is set to passive, and the client initiates the data connection from an ephemeral port to an ephemeral port on the server.

5.6.1 The Process

1. Start looking for FTP by checking for control connections between FTP clients and servers. A secondary goal will be to find whether the corresponding data connection is active or passive.

To find remote file servers, filter for outbound connections from ports 21, 22, and 23 individually, as the following example shows. Looking at each service individually prevents a high-traffic volume for one service from overshadowing traffic volume for another service. Gather the source addresses. As in the previous sections, look for hosts making up at least 1% of the remote file service traffic.

```
$ rwfilter sample.rw --type=out                              \
--protocol=6 --packets=4- --ack-flag=1 --sport=21            \
--pass=stdout                                                \
| rwstats --fields=sip --percentage=1 --bytes --no-titles    \
| cut -f 1 -d "|"                                           \
| rwsetbuild > ftp_servers.set
$ rwsetcat ftp_servers.set
203.0.113.69
203.0.113.71

$ rwfilter sample.rw --type=out                              \
--protocol=6 --packets=4- --ack-flag=1 --sport=22            \
--pass=stdout                                                \
| rwstats --fields=sip --percentage=1 --bytes --no-titles    \
| cut -f 1 -d "|"                                           \
| rwsetbuild > ssh_servers.set
$ rwsetcat ssh_servers.set
203.0.113.222
203.0.113.36
203.0.113.199
```

```
$ rwfilter sample.rw --type=out                              \
--protocol=6 --packets=4- --ack-flag=1 --sport=23            \
--pass=stdout                                                \
| rwstats --fields=sip --percentage=1 --bytes --no-titles    \
| cut -f 1 -d "|"                                           \
| rwsetbuild > telnet_servers.set
$ rwsetcat telnet_servers.set
[empty]
```

As the results in this example show, the sample network has two potential FTP servers, three potential SSH servers, and no Telnet servers.

2. To find clients on the network using these services, filter in the same way as you did to find servers in the previous step, but look at destination ports instead of source ports, as shown in the following example.

```
$ rwfilter sample.rw --type=out                              \
--protocol=6 --packets=4- --ack-flag=1 --dport=21            \
--pass=stdout                                                \
| rwstats --fields=sip --percentage=1 --bytes --no-titles    \
| cut -f 1 -d "|"                                           \
| rwsetbuild > ftp_clients.set
$ rwsetcat ftp_clients.set
203.0.113.220
203.0.113.33
203.0.113.199

$ rwfilter sample.rw --type=out                              \
--protocol=6 --packets=4- --ack-flag=1 --dport=22            \
--pass=stdout                                                \
| rwstats --fields=sip --percentage=1 --bytes --no-titles    \
| cut -f 1 -d "|"                                           \
| rwsetbuild > ssh_clients.set
$ rwsetcat ssh_clients.set
203.0.113.33

$ rwfilter sample.rw --type=out                              \
--protocol=6 --packets=4- --ack-flag=1 --dport=23            \
--pass=stdout                                                \
| rwstats --fields=sip --percentage=1 --bytes --no-titles    \
| cut -f 1 -d "|"                                           \
| rwsetbuild > telnet_clients.set
$ rwsetcat telnet_clients.set
203.0.113.33
```

The source addresses shown in this example are potential clients and are summarized in Table 19.

Table 19: Potential Remote Assets for the Sample Network

Internal IP	Internal Port	External IP	External Port	Comments
203.0.113.222	22	*	*	SSH/SFTP server
203.0.113.36	22	*	*	SSH/SFTP server
203.0.113.69	21	*	*	FTP server
203.0.113.71	21	*	*	FTP server
203.0.113.199	22	*	*	SSH/SFTP server
203.0.113.220	*	*	21	FTP client
203.0.113.33	*	*	21	FTP, SSH, Telnet client
203.0.113.199	*	192.168.5.138[8]	21	FTP client

Addresses 203.0.113.33 and 203.0.113.220 were removed from the sample network because they were already listed in the profile as gateway devices. Addresses 203.0.113.222 and 203.0.113.69 were also removed because we found earlier that the first is an email server and the second is a web server. However, their remote file service activities are noted in the profile.

3. Use the following command to determine whether the FTP assets are making active or passive connections by looking for connections that the servers initiate from TCP port 20. These will be active connections from the FTP servers.

```
$ rwfilter sample.rw --type=out                              \
--sipset=ftp_servers.set --sport=20 --flags-initial=S/SAFR \
--pass=stdout                                                \
| rwuniq --fields=sip
          sIP|   Records|
```

4. For clients, use the following command to look at traffic outbound to port 20. These connections will have been initiated by the server.

```
$ rwfilter sample.rw --type=out     \
--dport=20 --sipset=ftp_clients.set \
--pass=stdout                       \
| rwuniq --fields=sip
           sIP|   Records|
```

The results from the above example confirm that the servers and clients on the sample network consistently make passive connections.

5.6.2 How to Validate Findings

Start validating servers by resolving their addresses to domain names. The following example shows how we used `nslookup` to determine that address 203.0.113.71 resolves to ftp1.example.org in the sample network.

```
$ nslookup 203.0.113.71
Non-authoritative answer:
71.113.0.203.in-addr.arpa  name = ftp1.example.org.
```

[8] Actual address removed.

Next, log in to each server using a file transfer program or command-line utility. If using the command line, remember to try both SFTP and SSH for port 22 connections. If a prompt for a username or password comes up as in the following example, it is a valid server.

```
$ ssh 203.0.113.71
Password:
```

Try verifying FTP clients by looking for high-port-to-high-port traffic initiated by the client. If this traffic is preceded by a connection to port 21, it is probably FTP. To verify that it is FTP, look at the traffic in more detail. Remote access data connections consist of small packets (commands and responses) on the command channel and large packets (data transfers) over the data channel.

To look at the traffic in more detail, first find the external addresses with which the host is communicating. The following example shows the command and results.

```
$ rwfilter sample.rw --type=out                                    \
--saddress=203.0.113.199 --dport=21 --protocol=6 --packets=4- \
--pass=stdout                                                      \
| rwuniq --fields=dip
            dIP|    Records|
   192.168.5.138|         12|
```

Then look for traffic associated with that host. Sort the results by start time. The command and sorted results are in the following example.

```
$ rwfilter sample.rw --type=all                       \
--any-address=192.168.5.138 --protocol=6 --packets=4- \
--pass=stdout                                         \
| rwsort --fields=stime                               \
| rwcut --fields=sip,sport,dip,dport,stime,etime
           sIP|sPort|           dIP|dPort|       sTime|       eTime|
 203.0.113.199|   21| 192.168.5.138|49995|22:11:09.855|22:11:10.122|
 192.168.5.138|49750| 203.0.113.199|   21|22:11:56.398|22:11:56.691|
 203.0.113.199|   21| 192.168.5.138|49750|22:11:56.402|22:11:56.691|
 192.168.5.138|49752| 203.0.113.199|   21|22:12:37.163|22:12:37.419|
 203.0.113.199|   21| 192.168.5.138|49752|22:12:37.165|22:12:37.419|
 192.168.5.138|49753| 203.0.113.199|   21|22:12:37.581|22:12:37.869|
 203.0.113.199|   21| 192.168.5.138|49753|22:12:37.583|22:12:37.869|
 192.168.5.138|49754| 203.0.113.199|   21|22:12:45.473|22:12:45.721|
 203.0.113.199|   21| 192.168.5.138|49754|22:12:45.475|22:12:45.721|
 192.168.5.138|49758| 203.0.113.199|32879|22:12:50.581|22:12:50.869|
 203.0.113.199|32879| 192.168.5.138|49758|22:12:50.880|22:12:50.902|
 192.168.5.138|49758| 203.0.113.199|32879|22:16:37.581|22:16:37.869|
 203.0.113.199|32879| 192.168.5.138|49758|22:16:37.892|22:16:37.922|
 192.168.5.138|49758| 203.0.113.199|32879|22:26:37.581|22:26:37.869|
 203.0.113.199|32879| 192.168.5.138|49758|22:26:37.583|22:26:37.869|
 192.168.5.138|49755| 203.0.113.199|   21|22:31:07.124|22:31:07.425|
 203.0.113.199|   21| 192.168.5.138|49755|22:31:07.126|22:31:07.425|
 192.168.5.138|49995| 203.0.113.199|   21|22:31:09.853|22:31:10.122|
```

These results show a data channel established over port 21 and high-port-to-high-port traffic, indicating a passive mode. Active-mode FTP sessions will have a similar pattern, except instead

of high-port-to-high-port traffic, the data connection will have client-high-port-to-server-port-20 traffic.

5.6.3 Anomalies

Some of the anomalies for remote file services are listed below.

- servers acting as clients

  FTP servers can also be FTP clients. This is seen in the case of address 203.0.113.199 on the sample network.

- no response from the server (timeout)

  A timeout does not necessarily mean it is not a remote file access server. Some firewalls may be configured to allow this type of traffic only if it is from specific addresses. In fact, this is likely the reason that some of the servers on the sample network did not allow a connection; they may only serve traffic from their own network.

- control channel only traffic (for FTP)

  Sometimes a control channel will stay open without an accompanying data channel if keep-alive packets are being used. These will show in traffic as small packets at regular intervals. Also, brute force attempts against an FTP server will show up as control-channel-only traffic.

5.6.4 Results

Three servers on the sample network were left after validation—one FTP server and two SSH servers, which are listed in Table 20.

Table 20: Validated Remote File Services Assets for the Sample Network

Proto.	Internal Port	External IP	External Port	External Name	Comments
203.0.113.36					
6	22	*	*	*	SSH server
203.0.113.71 – ftp1.st.example.org					
6	21	*	*	*	FTP server
203.0.113.199					
6	22	*	*	*	SSH server

5.7 Other Services

Depending on the business's needs, a network may be running other services vital to the organization that have not yet been profiled. Or, there may have been a large amount of another type of traffic during an initial look at the top services on the network. The following steps can be used to generally profile other services.

Profiling a specific service depends on understanding how that service works. Knowledge of the protocol, ports, and architecture (client-server versus peer-to-peer) are all helpful to gain a general understanding of the typical packet flow for that service. For example, the Network Time Protocol (NTP) is carried over UDP port 123 and has a client-server architecture. It typically uses small packets of around 120 bytes, but the packet size can vary depending on the options used. The timing of the packets is fairly regular, but not necessarily an exact pattern.

The Requests for Comments (RFCs) from the IETF[9] are a good resource for information on a specific protocol, but they are often too detail-oriented for the purpose of getting a basic understanding of a protocol. Vendor websites and public help forums often have clear, in-depth explanations of a particular service or protocol.

5.7.1 The Process

In general, the process for profiling a specific service on a network starts with filtering by port number and protocol. If the service has a client-server architecture, profile the clients separately from the servers. The goal is to find out which hosts on the internal network are using which ports or protocols.

1. In Section 3, the sample data set was validated in part by looking at the most common services in use. Do the same process for other services, but leave out services that have already been profiled. Sort the results by the most common port in use for both clients and servers. An example of the command to validate other services and the sorted results follow.

```
$ rwfilter sample.rw --type=out       \
--sport=1-19,24,26-52,54-499,501-1023 \
--pass=stdout                         \
|rwstats --fields=sport --percentage=1
INPUT: 2550 Records for 378 Bins and 2550 Total Records
OUTPUT: Top 6 bins by Records (1% == 25)
sPort|   Records|  %Records|   cumul_%|
  123|       446| 17.490196| 39.333333|
  445|       187|  7.333333| 46.666667|
    3|        60|  2.352941| 49.019608|
    1|        51|  2.000000| 51.019608|
    2|        38|  1.490196| 52.509804|
```

2. These results show several services being "served" from the internal network that have not yet been profiled. Changing "source port" to "destination port" will show which services are requested from external servers. The following example shows this change to "destination port" and the results of that command.

[9] http://www.ietf.org/rfc.html

```
$ rwfilter sample.rw --type=out         \
--dport=1-19,24,26-52,54-499,501-1023   \
--pass=stdout                           \
| rwstats --fields=dport --percentage=1
INPUT: 4034 Records for 77 Bins and 4034 Total Records
OUTPUT: Top 5 bins by Records (1% == 40)
dPort|   Records|  %Records|   cumul_%|
  123|      2965| 73.500248| 73.500248|
  771|       229|  5.676748| 79.176996|
  778|       180|  4.462072| 83.639068|
  843|        80|  1.983143| 85.622211|
   81|        59|  1.462568| 87.084779|
```

The above results include several client services that also need to be profiled.

3. Look at the most used protocols other than TCP, UDP, GRE, and ESP (used in previous sections) to see if there is any significant traffic on alternative protocols. The following example shows the command and results for looking at these protocols.

```
$ rwfilter sample.rw --type=out           \
--protocol=1-5,7-16,18-46,48-49,52-       \
--pass=stdout                             \
|rwstats --fields=protocol --percentage=1
INPUT: 2003 Records for 1 Bin and 2003 Total Records
OUTPUT: Top 1 bins by Records (1% == 20)
pro|   Records|  %Records|   cumul_%|
  1|      2003|100.000000|100.000000|
```

The above ICMP (protocol 1) traffic should be examined further.

4. Get servers by filtering for outbound traffic from the (source) port used by servers for the specific service. For example, port 123 (Network Time Protocol, or NTP) is one of the services that needs to be profiled for the sample network. NTP uses the UDP IP protocol. The following example shows the filter and results.

```
$ rwfilter sample.rw --type=out \
--sport=123 --protocol=17       \
--pass=stdout                   \
| rwstats --fields=sip --percentage=1 --bytes
INPUT: 446 Records for 4 Bins and 51908 Total Bytes
OUTPUT: Top 3 bins by Bytes (1% == 519)
          sIP|               Bytes|    %Bytes|   cumul_%|
203.0.113.199|               24852| 47.877013| 47.877013|
 203.0.113.36|               19684| 37.920937| 85.797950|
203.0.113.222|                7296| 14.055637| 99.853587|
```

Each of these internal IP addresses has already been cataloged as a different type of server, but their NTP usage will be noted in the final profile. Do this for each of the other services found above, as well as for clients (using the destination port instead of the source port) and other protocols (leave out the port altogether). If the resulting assets are already listed in the profile, simply note their other activities as a separate line.

Here are some points to remember while profiling other common services:

- For some services, servers also initiate client requests.

  DNS and SMTP both have servers initiating client requests as part of the protocol. Peer-to-peer hosts will look like both servers and clients.
- List only assets making up a certain percentage of the traffic volume.

  For networks using gateways or proxy/NAT servers, choose a minimum percentage between 1 and 5 (in bytes). Otherwise, try to profile directly connected workstations separately from the rest of the machines based on their low byte volumes.
- Keep good records.

  Record the internal address, internal and external ports, protocol, and any specific external addresses the host is talking to, as well as any DNS names associated with the hosts. Also, record any protocol-specific aspects, such as active or passive for FTP clients. Remember to include notes on security or configuration concerns.

5.7.2 How to Validate Findings

Some services are more difficult to validate than others, but this section presents some general ways to validate the findings.

Start by comparing the list of new addresses with the lists from the services already profiled. Remove or correct hosts that are already listed in previous sections. A good practice is to categorize a host based on the service that makes up most of its traffic. Look for VPNs and gateways, which will show up in many of the services profiled.

Servers hosted within the network are often easier to validate than clients because the validation can rely on external sources. For servers, use the following methods to validate:

- looking up domain name

  Externally facing servers usually have a domain name associated with their IP address. Often, this name hints at what kind of service is operating on the machine. For example, the sample network's name servers each started with ns, ns1, and ns2.
- accessing the service

  Try to access the service in whatever way is appropriate. For web servers, navigate to the server with a browser. For DNS servers, send a DNS request directed at that server. If the response comes back as expected, the server is validated. Even if the server responds with an error, it can still be validated; for example, a potential web server is validated when it responds with, "The page you requested is unavailable."
- telnetting to the address

  Attempting to telnet to the address in question at the port on which the service is offered may return an error that gives clues as to the machine's purpose.
- responding to other servers

  Servers for some services such as DNS and SMTP communicate with each other as part of the protocol. This can confirm findings if it is appropriate for the service being profiled.

Validating clients can be more difficult but often is not necessary. The most important question should be whether that client behavior is appropriate on the network. If client traffic must be

validated, study the traffic pattern, looking specifically at the timing and size of the packets. Do not try to connect to unknown destination services to validate client traffic.

5.7.3 Anomalies

The following are some items to be aware of while profiling a service.

- historical behavior

  Old or unpatched hosts can have different behavior than is typically expected from a certain service.

- multiple services running on a single box

  Servers can be used to host several services in parallel with each other, either at one IP address or at different IP addresses, by using a different port for each service.

- general protocol anomalies

  This could be something suspicious, like tunneling, or something more legitimate, like a new application reusing an outdated protocol.

- a service running across multiple machines

  This may not mean that the service operates at multiple IP addresses. It can indicate load balancing or use of virtual machines.

- unconventional devices

  Many non-PC devices are now running services on their own hardware or connecting to services without using a PC as a conduit. For example, photocopiers are often able to email documents.

5.7.4 Results

Only one address we found using the process in this section had not already been listed in the profile. It is shown in Table 21.

Table 21: Assets for Other Services

Proto.	Internal Port	External IP	External Port	External Name	Comments
203.0.113.20					
6	*	*	123	ntp.*	NTP Client

6 Catalog Remaining Active Assets

By this point, most common services will have been cataloged, plus any services that were profiled based on top-used services in the network. The list of assets in the profile hopefully covers almost all of the traffic. Compare the list of hosts in the current profile with the list of active assets found at the beginning of this report using the following command.

```
$ rwsettool --difference talkers.set profiled_assets.set \
--output-path=not_profiled.set
$ rwsetcat not_profiled.set --count
10
```

This result shows that for the sample network, there are 10 hosts that have not yet been profiled. Before profiling any remaining hosts, expand the time frame for the sample data set to see if there are any other active hosts that were not represented in the smaller data set. After finding the hosts that are left, profile each one individually.

6.1 The Process

The expanded data set should be at least one month's worth of data to get accurate results. If the query time is too great, it is possible to separate out inbound traffic and only look at outbound traffic over that duration. However, this is likely to produce a very large file, so the benefits of doing such a separation might be outweighed.

Finding the additional talkers from the expanded data set is the same process as finding the talkers in Section 4 but excludes the addresses already in the profile in order to speed up the query. The following examples show the commands and results used to find additional talkers.

```
$ rwfilter --start-date=2011/10/01:00 --end-date=2011/10/31:23  \
--type=out,outweb                                               \
--not-sipset=talkers.set --protocol=6 --packets=4- --ack-flag=1 \
--pass=stdout                                                   \
| rwset --sip-file=tcp_talkers.set
$ rwsetcat tcp_talkers.set --count
6

$ rwfilter --start-date=2011/10/01:00 --end-date=2011/10/31:23 \
--type=out                                                     \
--not-sipset=talkers.set --protocol=0-5,7-                     \
--pass=stdout                                                  \
| rwset --sip-file=other_talkers.set
$ rwsetcat other_talkers.set --count
3

$ rwsettool --union tcp_talkers.set other_talkers.set \
--output-path=expanded_talkers.set
$ rwsetcat expanded_talkers.set --count
6
```

Combine the list of these new active assets with the active assets not already profiled. After we combined them, the sample network had 16 leftover assets. Profile leftover assets by looking at what kind of traffic the assets are generating. For each address, look at which ports make up the majority of the traffic, as the following command and results show.

```
$ rwfilter --start-date=2011/10/01:00 --end-date=2011/10/31:23 \
--type=out,outweb                                              \
--saddress=203.0.113.22                                        \
--pass=stdout                                                  \
| rwstats --fields=dport,sport,protocol --count=5 --bytes
INPUT: 730 Records for 7 Bins and 64220 Total Bytes
OUTPUT: Top 5 Bins by Bytes
dPort| sPort|pro|               Bytes|    %Bytes|   cumul_%|
32986|  3389|  6|               18772| 29.230769| 29.230769|
32988|  3389|  6|               18468| 28.757396| 57.988166|
32987|  3389|  6|               10564| 16.449704| 74.437870|
32989|  3389|  6|                8056| 12.544379| 86.982249|
32982|  3389|  6|                2812|  4.378698| 91.360947|
```

The machine in the sample network is using Remote Desktop Protocol because it is being connected through port 3389. We created a “miscellaneous” section at the end of the profile to hold these assets with uncommon protocols. If the machine has (or is connecting to) multiple services that each make up at least 10% of the total traffic, list each of those services, but make a note of which service is the most used. Table 22 does this by listing the most used service first.

Before adding these findings to the profile, try to validate them. For clients, try resolving the names of the external addresses to which they connect. For servers, try resolving the address of the server itself. Use any of the other validation methods from Section 5.7.2 to help with this. Many sites have information on ports and their corresponding protocols.

Throughout the process of cataloging unfamiliar and low-traffic assets, you may spend a great deal of time researching ports and protocols or examining traffic in detail. This is a very valuable part of the profiling process because it allows you to get an accurate picture of what the asset is really doing on the network.

6.2 Example Findings

While profiling the leftovers, you may find all kinds of random and interesting traffic to or from hosts on the network. Remember to note these findings in the profile, even if they are difficult to verify or seem incorrect. This information can always be looked into in more detail during a further investigation. If it was difficult to verify an asset, note that in the profile also.

Here are some examples of traffic that can be found in the leftovers.

- services already profiled

  When the common services were profiled in Section 5, the assets included were limited to those comprising at least 1% of the traffic for that service. That was done because assets making up less than that percentage were assumed to be using or providing that service for secondary purposes. However, sometimes even the assets making up such a small percentage of traffic are actually using or providing the service as a primary purpose. This is particularly

likely when two or three very heavily used assets skew the percentages, as was the case with the web servers on the sample network.

- common services over encrypted ports or legacy ports

  Some services may use a separate port for encrypted connections or may still be using a port that has since been changed. This is simply a matter of including the port in the filter.

- other well-known services

  There are far too many well-known services in use today to list in this report. Protocols such as NTP, RDP, LDAP, SMB, and sqlnet may show up on the network. At some point, verify that these services are appropriate for the network. For example, NetBIOS, LDAP, and SMB traffic should not be crossing the network perimeter.

- routers

  Because you included protocols other than TCP and UDP using the process in this section, some routing traffic may be visible. It should be fairly easy to tell which router is the internet gateway by the specific protocols used.

- uncommon or uncategorized ports

  Sometimes an asset will generate traffic for a seemingly random service. For example, in the sample network, the only traffic found to and from one of the assets consisted of updates for a mobile GPS based on the registered port number used. This particular asset was not noted in the profile because the overall volume was insignificant.

  Do not spend too much time on the assets in this category. Rather than examining traffic details and researching port numbers, simply list the ports and addresses in the profile.

6.3 Results

All of the assets found in the talkers section (4) are now profiled. For the sample network, we added many of the assets classified in this section to other sections in the final profile (see Appendix A). Specifically, we listed many in the web server and VPN sections. Table 22 shows the assets we found using the process in this section before we added them to the final table.

Table 22: Final Assets from Leftovers in the Sample Network

Proto.	Internal Port	External IP	External Port	External Name	Comments
203.0.113.22					
6	3389	*	*	*	Remote Desktop
203.0.113.24					
6	*	*	80	*	
203.0.113.170					
6	22	65.104.x.x	*	[Network Ops/Support]	
203.0.113.75 – www.example.org					
6	*	*	443		
6	80, 443	*	*		
203.0.113.53					
6	*	*	80, 443		
203.0.113.46					
6	*	*	80, 443		
203.0.113.74 – uhura.st.example.org					
6	80, 443	*	*	*	Web server
203.0.113.72 – www01.sales.example.org					
6	80	*	*	*	Web server
203.0.113.29 – example.org					
6	80	*	*	*	Web server
6	*	*	80	*	
17	*	4.2.2.1	53	softdns.org	DNS
203.0.113.28					
6	8080	198.51.100.12	62571	[data center]	Web server
6	80,443	*	*	*	Web client
203.0.113.27					
6	80,443	*	*	*	Web server
203.0.113.23 - picard.st.example.org					
6	80	*	*	*	Web server
203.0.113.77					
6	*	*	80		Web client
203.0.113.183					
6	*	*	80,443		Web client
203.0.113.193					
6	*	*	80, 443		Web client
203.0.113.226					
6	*	*	80		Web client

7 Maintain the Profile

Networks, especially large ones, can change quickly and often. The new profile you created using the process in this report may not be accurate six months from now. Much of what was done throughout this report can be automated. Particular netflow analysis software may allow for scheduling filters to run weekly or monthly or at least allow for saving any custom filters created during the process. Filters for command-line tools like SiLK can be rolled into a script and then run as part of a cron job. It may be wise to have the tool send an email alert when it finds something that is not already in the profile or when one of the assets in the profile has not been active for a long time.

Automated tools can only do so much. You will need to consistently validate potential assets for a particular service before adding them to the profile. This may seem daunting, but the more consistently you do it, the quicker it will become. Also, consistently updating and validating new assets will keep communication lines open between you and the people who administer the network. Most importantly, this will help you stay informed as assets are removed, added, or updated.

Update the profile at least once a month. One way to do this is to run through the whole profiling process again (automated or not), find top talkers on the network, find potential assets for each service, validate those assets, expand the data set, and profile the leftovers. Only the assets that change will need to be validated. Any assets that have been added, removed, or changed should stand out and can be further investigated.

Another way to update the profile is to look at trends over time. Use the current profile as a baseline and take a daily snapshot of the assets (the services they are running and the external addresses with which they are communicating) as well as the list of active assets on the network. This is called “trending.” Use trending to visualize when changes are made and whether they are persistent changes that should be added into the profile. Many analysis packages come with trending capabilities. Trending scripts for SiLK can be downloaded that show changes over time using graphs. This is probably the best option for updating the profile because it takes less time and requires little effort. However, changes in trends still need to be validated before being added to the profile.

8 Conclusion

Having an accurate list of assets and a network map can be a powerful tool when it comes time to makes changes in the network, buy new equipment, respond to an incident, or conduct a security assessment. For the profile to have even more of an impact, consider adding more data points. You can list the machine administrator, update schedule, intended purpose, certificate registrar if applicable, and anything else pertaining to administration or security. For instance, findings can be correlated with an asset management database.

Profiling is also the first step in anomaly detection. You may have discovered during this process that there was at least one machine doing something for which it was not intended. For example, a web server may also be used to send and receive email, or an email server may violate policy with regular telnet traffic. Go through the profile and make a list of security issues like these by comparing the profile with what *should* be on the network. Use this list to guide a security audit or stress to administration the importance of network profiling.

The profiling process has certain limitations, such as basing services on port number, which is not always accurate because the standard is not enforced. Also, it relies on sensor placement to get accurate results. As long as you are aware of these limitations, you can use judgment and knowledge of networking to overcome them.

Use these results to report on what changes need to be made within the network and why. You might also suggest organizational policies or procedures that will help keep records up to date and prevent machines from being used for unintended purposes. You may even be able to find inefficiencies in the network that can be corrected to save resources, such as combining two separate low-volume email and web servers onto the same machine. Whatever the original goal of profiling the network was, you can be sure to find other benefits from this process.

Appendix A Sample Network Profile

Table 23 is the final list of sample network assets found and validated during the profiling process.

Table 23: Final Sample Network Profile

Proto.	Internal Port	External IP	External Port	External Name	Comments
Web Servers					
203.0.113.198 – example.org					
6	80, 443	*	*	*	
203.0.113.197 – kirk.st.example.org					
6	80,443	*	*	*	
6	*	*	80		Web browsing, or possibly distributed hosting/mirror
203.0.113.196 – spock.example.org					
6	80, 443		*	*	Expired SSL certificate
203.0.113.194 – pghwebmail.st.example.org					
6	443	*	*		Reverse DNS is misspelled
6	*	*	80		
203.0.113.75 – www.example.org					
6	*	*	443		
6	80, 443	*	*		
203.0.113.74 – uhura.st.example.org					
6	80, 443	*	*	*	Low volume
203.0.113.72 – www01.sales.example.com					
6	80	*	*	*	Low volume
203.0.113.69 – www01.st.example.org					
6	80, 443	*	*		
6	21	*	*		FTP server
203.0.113.44 – vss1.st.example.org					
6	80, 443	*	*		VMware View; DNS resolve directs to 203.0.113.198: unable to get to port 80 from external network
17	4172	*	50002		VMWare protocol
203.0.113.29 – example.org					
6	80	*	*		Low volume
6	*	*	80		Web client
17	*	4.2.2.1	53	softdns.org	DNS
203.0.113.28					
6	8080	198.51.100.12	*	[Data Center]	
6	*	*	80, 443		
203.0.113.27					
6	80,443	*	*	*	Low volume

Proto.	Internal Port	External IP	External Port	External Name	Comments
203.0.113.23 - picard.st.example.org					
6	*	*	80, 443		
6	80	*	*		Low-volume web server
Web Clients					
203.0.113.220					
6	*	*	*		Web gateway
203.0.113.53					
6	*	*	80, 443		
203.0.113.46					
6	*	*	80, 443		
203.0.113.24					
6	*	*	80, 443		
203.0.113.77					
6	*	*	80		
203.0.113.183					
6	*	*	80,443		
203.0.113.193					
6	*	*	80, 443		
203.0.113.226					
6	*	*	80		
Email					
203.0.113.231 - smtpfw.st.example.org					
6	25	*	*		
6	*	*	25		
203.0.113.200 - smtp-apps.example.org					
6	*	*	25		
6	80	*	*		
203.0.113.195 - sulu.example.org					
6	25	*	*		
6	*	*	25		
6	80	*	*		
6	*	*	80		
203.0.113.221 - omail.example.org					
6	25	*	*		
6	*	*	25		
6	80	*	*		
203.0.113.222 – imail.example.org					
6	25	*	*		
6	*	*	25		
6	80	*	*		
6, 17	*	*	53	*	DNS client
6	22	*	*	*	Remote file access

Proto.	Internal Port	External IP	External Port	External Name	Comments
DNS Servers					
203.0.113.50 – ns.st.example.org					
17	53	*	*	*	Iterative, authoritative
203.0.113.51 – ns1.st.example.org					
6,17	53	*	53	*	Recursive
203.0.113.52 – ns2.st.example.org					
6,17	53	*	53	*	Recursive
VPN Gateways					
203.0.113.33 - webvpn.pittsburgh.pa.gov					
50	0	*	0		50 S2S, remote access
17	4500	*	*		
6	443, 8080	*	*		DNS resolve directs to 203.0.113.198; self-signed SSL
6	*	*	*		A little bit of everything
203.0.113.35					
50	0	*	0		1 S2S, remote access
17	500	*	500		
203.0.113.178 – cop-vpn.st.example.org					
17	4500	*	*		
17	10000, 500	*	10000, 500		
50	0	*	0		1 S2S, remote access
6	80,443	*	*		
Remote Access					
203.0.113.36					
6	22	*	*	*	SFTP server
6	*	*	6000		
6	*	*	80, 443		
6	Eph	*	Eph		
203.0.113.71 – ftp1.example.org					
6	21	*	*	*	
6	*	*	80, 443		
203.0.113.170					
6	22	65.104.x.x	*	[Network Ops/Support]	
203.0.113.199					
6	22	*	*		
17	123	*	*		
Miscellaneous					
203.0.113.20					
17	*	*	123	ntp.*	NTP client
203.0.113.22					
6	*	75.150.30.4	5721		Remote Desktop

Proto.	Internal Port	External IP	External Port	External Name	Comments
6	*	*	80		
203.0.113.30					
47	0		0		Router
1	0		2048		
6	*	*	80, 443	*	Web client
17	123	*	123		

Appendix B Scripts

These scripts follow the steps provided in the body of the report. Edit them with the appropriate values for dates, addresses, and so on. Use the Unix "script" command to save text output to a file.

Remember that the scripts will not provide final profile results. They will simply return a list of potential addresses corresponding to each service and, in some cases, a few other details about the traffic being filtered. It is the profiler's job to further investigate and validate the output using the process described in the body of the report.

Section 3 Script

This script selects a sample data set and prints out the top protocols, top requested services, and top services being provided.

```
#! /bin/bash
echo Initial dataset:
rwfilter --type=out,outweb --start-date=2011/09/28:00 --end-
date=2011/09/28:23 --protocol=0- --pass=sample.rw
rwfileinfo sample.rw

echo -e "\nTop protocols:"
rwstats sample.rw --fields=protocol --count=5

echo -e "\nTop services being requested:"
rwfilter sample.rw --type=out,outweb --protocol=0- --pass=stdout|rwstats --
count=5 --fields=dport

echo -e "\nTop services being provided:"
rwfilter sample.rw --type=out,outweb --protocol=0- --pass=stdout|rwstats --
count=5 --fields=sport
```

Section 4 Script

This script finds the active addresses in the sample.rw data set and turns them into a SiLK IPset.

```
#! /bin/bash
echo Number of TCP talkers:
rwfilter sample.rw --type=out,outweb --protocol=6 --packets=4- --ack-flag=1
--pass=stdout|rwset --sip-file=tcp-talkers.set
rwsetcat tcp-talkers.set --count

echo -e "\nNumber of talkers on other protocols:"
rwfilter sample.rw --type=out --protocol=0-5,7- --pass=stdout|rwset --sip-
file=other-talkers.set
rwsetcat other-talkers.set --count

rwsettool --union tcp-talkers.set other-talkers.set --output-
path=talkers.set
```

```
rm tcp-talkers.set
rm other-talkers.set

echo
rwsetcat talkers.set --network-structure

echo -e "\nClass C network blocks:"
rwsetcat talkers.set --network-structure=C

echo -e "\nTransit traffic:"
rwfilter sample.rw --type=out,outweb --not-sipset=talkers.set --
pass=stdout|rwtotal --sip-first-8 --summation --skip-zeroes --no-titles |
cut -f 2 -d "|"
rwfilter sample.rw --type=out,outweb --dipset=talkers.set --
pass=stdout|rwtotal --sip-first-8 --summation --skip-zeroes --no-titles |
cut -f 2 -d "|"
```

Section 5 Script

This script finds potential assets for each service in Section 5.

```
#! /bin/bash
echo "######## Web Servers ########"
rwfilter sample.rw --type=outweb --sport=80,443,8080 --protocol=6 --
packets=4- --ack-flag=1 --pass=stdout|rwstats --fields=sip --percentage=1 --
bytes --no-titles|cut -f 1 -d "|"|rwsetbuild > web_servers.set
echo Potential Web Servers:
rwfilter sample.rw --type=outweb --sport=80,443,8080 --protocol=6 --
packets=4- --ack-flag=1 --sipset=web_servers.set --pass=stdout|rwuniq --
fields=sip,sport --bytes --sort-output

echo -e  "\n######## Web Clients ########"
rwfilter sample.rw --type=out,outweb --protocol=6 --ack-flag=1 --packets=4-
--dport=80,8000,8080,443,1935,1755,554 --pass=stdout|rwset --sip-
file=tcpclients.set
rwfilter sample.rw --type=out,outweb --protocol=17 --dport=1755,554 --
pass=stdout|rwset --sip-file=udpclients.set
rwsettool --union tcpclients.set udpclients.set --output-path=webclients.set
rm tcpclients.set
rm udpclients.set
rwfilter sample.rw --type=out,outweb --sipset=webclients.set --
dport=80,8080,8000,443,1935,1755,554 --pass=stdout|rwstats --fields=sip --
percentage=1 --no-titles|cut -f 1 -d "|" |rwsetbuild > web_clients.set
rm webclients.set

echo Potential Web Clients:
rwfilter sample.rw --type=out,outweb --sipset=web_clients.set --
dport=80,8080,8000,443,1935,1755,554 --pass=stdout|rwuniq --
fields=sip,dport,protocol --bytes --sort-output

echo -e "\n"Web Client traffic volume:
```

```
rwfilter sample.rw --type=out,outweb --sipset=web_clients.set --
dport=80,8080,8000,443,1935,1755,554 --pass=stdout|rwuniq --fields=sip --
bytes

echo -e "\n######## Email ########"
echo Potential SMTP servers
rwfilter sample.rw --type=out --sport=25,465,110,995,143,993 --protocol=6 --
packets=4- --ack-flag=1 --pass=stdout|rwset --sip-file=smtpservers.set
rwfilter sample.rw --type=out --sport=25,465,110,995,143,993 --
sipset=smtpservers.set --protocol=6 --packets=4- --ack-flag=1 --
pass=stdout|rwuniq --fields=sip --bytes --sort-output

echo -e "\n"Potential SMTP clients:
rwfilter sample.rw --type=out --dport=110,143,587,993,995 --protocol=6 --
ack-flag=1 --packets=4- --pass=stdout|rwset --sip-file=email_clients.set
rwfilter sample.rw --type=out --dport=110,143,587,993,995 --protocol=6 --
ack-flag=1 --packets=4- --sipset=email_clients.set --pass=stdout|rwuniq --
fields=sip,dport --bytes --sort-output

echo -e "\n######## DNS ########"
echo DNS Servers:
rwfilter sample.rw --type=out --sport=53 --protocol=17 --pass=stdout|rwstats
--fields=sip --percentage=1 --packets --no-titles|cut -f 1 -d "|"|
rwsetbuild > dns_servers.set
rwsetcat dns_servers.set

echo -e "\n"DNS Clients:
rwfilter sample.rw --type=out --dport=53 --protocol=17 --pass=stdout|rwstats
--fields=sip --percentage=1 --packets --no-titles|cut -f 1 -d "|"|rwsetbuild
> dns_clients.set
rwsetcat dns_clients.set

echo -e "\n"Recursive DNS Servers:
rwsettool --intersect dns_clients.set dns_servers.set --output-
path=dns_recursive.set
rwsetcat dns_recursive.set

echo -e "\n"Iterative DNS Servers:
rwsettool --difference dns_servers.set dns_recursive.set --output-
path=dns_iterative.set
rwsetcat dns_iterative.set

echo -e "\n"DNS Client traffic:
rwfilter sample.rw --type=out --sipset=dns_clients.set --dport=53 --
pass=stdout|rwstats --fields=sip --percentage=1 --bytes

echo -e "\n######## VPN ########"
echo Potential VPNs:
rwfilter sample.rw --type=out --protocol=47,50,51 --pass=stdout|rwuniq --
fields=sip --no-titles|cut -f 1 -d "|" |rwsetbuild > vpn.set
rwfilter sample.rw --type=out --sipset=vpn.set --pass=stdout|rwuniq --
fields=sip,protocol --bytes --sort-output

echo -e "\n######## Remote File Services ########"
```

```
echo Potential FTP Servers:
rwfilter sample.rw --type=out --protocol=6 --packets=4- --ack-flag=1 --
sport=21 --pass=stdout|rwstats --fields=sip --percentage=1 --bytes --no-
titles|cut -f 1 -d "|"|rwsetbuild > ftpservers.set
rwsetcat ftpservers.set

rwfilter sample.rw --type=out --protocol=6 --packets=4- --ack-flag=1 --
sport=22 --pass=stdout|rwstats --fields=sip --percentage=1 --bytes --no-
titles|cut -f 1 -d "|"|rwsetbuild>ssh_servers.set
echo -e "\nPotential SSH Servers"
rwsetcat ssh_servers.set

echo -e "\n"Potential Telnet Servers:
rwfilter sample.rw --type=out --protocol=6 --packets=4- --ack-flag=1 --
sport=23 --pass=stdout|rwstats --fields=sip --percentage=1 --bytes --no-
titles|cut -f 1 -d "|"|rwsetbuild > telnet_servers.set
rwsetcat telnet_servers.set

echo -e "\n"Potential FTP Clients:
rwfilter sample.rw --type=out --protocol=6 --packets=4- --ack-flag=1 --
dport=21 --pass=stdout|rwstats --fields=sip --percentage=1 --bytes --no-
titles|cut -f 1 -d "|"|rwsetbuild > ftp_clients.set
rwsetcat ftp_clients.set

rwfilter sample.rw --type=out --protocol=6 --packets=4- --ack-flag=1 --
dport=22 --pass=stdout|rwstats --fields=sip --percentage=1 --bytes --no-
titles|cut -f 1 -d "|"|rwsetbuild > ssh_clients.set
echo
echo Potentail SSH Clients:
rwsetcat ssh_clients.set

rwfilter sample.rw --type=out --protocol=6 --packets=4- --ack-flag=1 --
dport=23 --pass=stdout|rwstats --fields=sip --percentage=1 --bytes --no-
titles|cut -f 1 -d "|"|rwsetbuild > telnet_clients.set
echo
echo Potential Telnet Clients:
rwsetcat telnet_clients.set

echo
echo FTP Servers making active connections:
rwfilter sample.rw --type=out --sipset=ftpservers.set --sport=20 --flags-
initial=S/SAFR --pass=stdout|rwuniq --fields=sip

echo
echo FTP Clients making active connections:
rwfilter sample.rw --type=out --dport=20 --sipset=ftp_clients.set --
pass=stdout|rwuniq --fields=sip

echo -e "\n######## Leftovers ########"
echo Servers:
rwfilter sample.rw --type=out --sport=1-19,24,26-52,54-499,501-1023 --
pass=stdout|rwstats --fields=sport --percentage=1

echo -e "\nClients:"
```

```
rwfilter sample.rw --type=out --dport=1-19,24,26-52,54-499,501-1023 --
pass=stdout| rwstats --fields=dport --percentage=1
```

Section 6 Script

This script uses an extended data sample to find any remaining active hosts. Profile these hosts using the methods in Section 5.7.

```
#! /bin/bash
echo Number of unprofiled active hosts before expansion:
mv talkers.set talkers.txt
rwsettool --union *.set --output-path=profiled_assets.set
mv talkers.txt talkers.set
rwsettool --difference talkers.set profiled_assets.set --output-
path=not_profiled.set
rwsetcat not_profiled.set --count

sdate=2011/10/01:00
edate=2011/10/31:23
rwfilter --start-date=$sdate --end-date=$edate --type=out,outweb --not-
sipset=talkers.set --protocol=6 --packets=4- --ack-flag=1 --
pass=stdout|rwset --sip-file=tcp-talkers.set

rwfilter --start-date=$sdate --end-date=$edate --type=out --not-
sipset=talkers.set --protocol=0-5,7- --pass=stdout|rwset --sip-file=other-
talkers.set

rwsettool --union tcp-talkers.set other-talkers.set not_profiled.set --
output-path=new_unprofiled.set
echo Number of unprofiled hosts after expansion:
rwsetcat new_unprofiled.set --count

rm tcp-talkers.set
rm other-talkers.set
rm not_profiled.set

echo -e "\n"Profile the hosts below using Chapter 6
rwsetcat new_unprofiled.set
```

References

URLs are valid as of the publication date of this document.

[Lonvick 2006]
Lonvick, C. & Ylonen, T. *The Secure Shell (SSH) Transport Layer Protocol* (RFC 4253). Network Working Group, Internet Engineering Task Force, January 2006. http://www.ietf.org/rfc/rfc4253.txt

[Mockapetris 1987]
Mockapetris, P. *Domain Names – Implementation and Specification* (RFC 1035). Network Working Group, Internet Engineering Task Force, November 1987. http://www.ietf.org/rfc/rfc1035.txt

[OMB 2006]
Executive Office of the President. *Memorandum for the Heads of Departments and Agencies* (M-06-16). Executive Office of the President, Office of Management and Budget, June 2006. www.whitehouse.gov/omb/memoranda/fy2006/m06-16.pdf

[Postel 1983]
Postel, J. & Reynolds, J. *Telnet Protocol Specification* (RFC 854). Network Working Group, Internet Engineering Task Force, May, 1983. http://www.ietf.org/rfc/rfc854.txt

[Postel 1985]
Postel, J. & Reynolds, J. *File Transfer Protocol (FTP)* (RFC 959). Network Working Group, Internet Engineering Task Force, October 1985. http://www.ietf.org/rfc/rfc959.txt

[US-CERT 2008]
US-CERT. *Multiple DNS Implementations Vulnerable to Cache Poisoning* (VU#800113). US-CERT, 2008. http://www.kb.cert.org/vuls/id/800113

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
(Leave Blank)	August 2012	Final

4. TITLE AND SUBTITLE	5. FUNDING NUMBERS
Network Profiling Using Flow	FA8721-05-C-0003

6. AUTHOR(S)

Austin Whisnant, Sid Faber

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)	8. PERFORMING ORGANIZATION REPORT NUMBER
Software Engineering Institute Carnegie Mellon University Pittsburgh, PA 15213	CMU/SEI-2012-TR-006

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSORING/MONITORING AGENCY REPORT NUMBER
HQ ESC/XPK 5 Eglin Street Hanscom AFB, MA 01731-2116	ESC-TR-2012-006

11. SUPPLEMENTARY NOTES

12A DISTRIBUTION/AVAILABILITY STATEMENT	12B DISTRIBUTION CODE
Unclassified/Unlimited, DTIC, NTIS	

13. ABSTRACT (MAXIMUM 200 WORDS)

This report provides a step-by-step guide for profiling—discovering public-facing assets on a network—using network flow (netflow) data. Netflow data can be used for forensic purposes, for finding malicious activity, and for determining appropriate prioritization settings. The goal of this report is to create a profile to see a potential attacker's view of an external network.

Readers will learn how to choose a data set, find the top assets and services with the most traffic on the network, and profile several services. A case study provides an example of the profiling process. The underlying concepts of using netflow data are presented so that readers can apply the approach to other cases. A reader using this report to profile a network can expect to end with a list of public-facing assets and the ports on which each is communicating and may also learn other pertinent information, such as external IP addresses, to which the asset is connecting. This report also provides ideas for using, maintaining, and reporting on findings. The appendices include an example profile and scripts for running the commands in the report. The scripts are a summary only and cannot replace reading and understanding this report.

14. SUBJECT TERMS	15. NUMBER OF PAGES
network, profiling, inventory, servers, services, flow, netflow, traffic, situational awareness	75

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL