BRNO UNIVERSITY OF TECHNOLOGY
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF INFORMATION TECHNOLOGY
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

NETWORK MONITORING
MONITOROVÁNÍ SÍTĚ

ISA TERM PROJECT
SEMESTRÁLNÍ PROJEKT Z ISA

AUTHOR **ANNA KHAITOVICH**
AUTOR PRÁCE

SUPERVISOR **Ing. JAN PLUSKAL**
VEDOUCÍ PRÁCE

BRNO 2017

Abstract

The purpose of this project was to get familiar with various network scanning methods and algorithms, implement a communication model that uses those algorithms to scan a network for active hosts and hosts with open ports and test this model. Isamon software could be used as a simple network scanning and mointoring tool.

First, network address and broadcast address of an examined network should be calculated using submask. Those addresses are not going to be scanned, they are needed to get the range of addresses that will be. Then, for active hosts scanning, I've used ICMP echo scan. For port scanning, I've used TCP connect scanning for TCP ports and UDP ICMP port unreachable scanning for UDP ports. There is also a possibility to use TCP SYN scanning for TCP ports (it is faster and harder to detect), but was not used in Isamon because failed to be implemented in time and may be delivered with later patches. Also, ARP scanning could be used for scanning active hosts instead for ICMP echo scanning (it works on lower level, so is less dependent on network configuration), but was not for similar reasons.

Isamon was not desinged to be a realiable, fast network scanning tool (Nmap does this job a lot better), but to create an implementation of scanning algorithms described above. Isamon is not guaranteed to output 100% correct results, however, may be extended with patches that improve both speed and reliability of it with solutions described in this document.

Keywords

Network scanning, network monitoring, TCP scanning, UDP scanning, ICMP scanning, active hosts scanning, TCP connect scan, UDP ICMP port unreachable scan, ICMP echo scan, port scanning

Reference

KHAITOVICH, Anna. *Network monitoring.* Brno, 2017. ISA term project. Brno University of Technology, Faculty of Information Technology. Supervisor Ing. Jan Pluskal

Network monitoring

Declaration

Hereby I declare that this project was prepared as an original author's work under the supervision of Ing. Jan Pluskal. All the relevant information sources, which were used during preparation of this thesis, are properly cited and included in the list of references.

. . . .
Anna Khaitovich
November 20, 2017

Contents

Listings

Chapter 1

Introduction

The purpose of creating Isamon was to examine and try different implementations of host and ports scans. When I was looking for proper scanning algoorithms that could be used in this project I was using Nmap as inspiration.

Nmap uses several different scanning methods, such as:

- TCP connect() scanning
- TCP SYN scanning (used in Isamon)
- TCP FIN scanning
- Fragmentation scanning
- TCP reverse ident scanning
- UDP ICMP port unreachable scanning (used in Isamon)
- UDP recvfrom() and write() scanning
- ICMP echo scanning (used in Isamon)

Study of ICMP echo scan and an attempt to use this algorithm for active hosts scanning showed that it is relatively reliable, but only if the correct round trip time is set by the user and the scanned network's configuration is not blocking ICMP messages. ARP scanning is proved to be more accurate, but was not implemented in Isamon because of time limits.

Study of TCP SYN scan showed it is really fast, since it does not establishe a complete 3 way TCP handshake, like in TCP connect() scanning. Instead it just sends a SYN packet to host and looks for the SYN/ACK sent from host in response. Results are almost as accurate as result of TCP connect, but the scan is much faster. There is also a possibility to use TCP FIN scanning, which could be profitable only to use against *NIX machines.

For UDP port scanning, the UDP ICMP port unreachable scanning was proved to be the best solution. The only disadvantage of this method is that uses raw sockets, thus requires root access. Non-root users could use UDP recvfrom() and write() scanning, but it is much less reliable, than UDP ICMP port scanning.

More detailed description of most of these scan methods could be found in appropriate chapters of this document.

Chapter 2

Usage

```
isamon [-h] [-i <interface>] [-t] [-u] [-p <port>] [-w <ms>] -n <net_address/mask>

  -h --help -- output help
  -i --interface <interface> -- scan using specific interface
  -n --network <net_address/mask> -- IP address with a submask that define scanning
    range
  -t --tcp -- use TCP scan
  -u --udp -- use UDP scan
  -p --port <port> -- port that will be scanned. If not provided, all ports from
    range 1-65535 will be scanned
  -w --wait <ms> -- maximum RTT
```

Listing 2.1: Options usage

Chapter 3

Utilities

3.1 Scan range calculation

Network IP adress is given in XXX.XXX.XXX.XXX/YY format, where XXX.XXX.XXX.XXX is IPv4 address and YY is a subnet mask. Isamon implements and algorithm that transforms YY into a binary form subnet mask. For example, /24 would be transformed into $(4294967040)_{10} = (11111111111111111111111100000000)_2$ - a 32-bit value with 24 highest bits set to 1.

Then the range of addresses that will be scanned is constructed as the following:

```
network = XXXX.XXX.XXX.XXX & subnet
broadcast = XXX.XXX.XXX.XXX | ~subnet
first IP = network + 1
last IP =  broadcast − 1
```

Listing 3.1: Scan range calculation

3.2 Local IP calculation

Calculation of the local IP of the machine is needed to fill in the source IP field in IPv4 packets that are sent during scans. Unfortunatelly, a `gethostbyname` call would return 127.0.0.1, while the *external* IP adress is needed. So it is done by setting up a connection with Google DNS server (8.8.8.8) via a UDP socket, then getting the address to which the socket is bound as the local address.

3.3 Checksum calculation

Checksum calculation is also needed to fill in an IPv4 checksum field. IPv4 uses the checksum to detect corruption of packet headers. The TCP protocol includes an extra checksum that protects the packet „payload“ as well as the header. This is in addition to the header-checksum of IP.

The algorithm for the TCP and IPv4 checksums is identical. The data is processed a word at a time. The words are added using ones-complement arithmetic, with the location holding the checksum set to be zeros. Once this chain of additions is complete, the result is negated in ones-complement by taking the binary not, and the result is stored in the right spot. If this operation is repeated then the result of the checksum should be the binary all-ones.

Source: [7]

Chapter 4

Scanning methods used in Isamon

4.1 ICMP echo scanning

Isamon outputs every active host's IP address to standard output stream. ICMP echo scanning is used to determine what hosts in a network are up. This technique is generally used for this purpose (Unix `ping`, `traceroute` utilies). An IP packet containig ICMP echo request message is to source IP address. Then, if ICMP echo reply with type 0 (OK) is recieved, the host that has sent it is concidered active. If, after a certain timeout (specified by user with -w or –wait option), no such response has been recieved, host is considered to be down.

However, even if a host didn't return ICMP echo reply with type OK, Isamon will still scan it's port(s). This is because ICMP echo scanning could be ineffective, if a scanned network is set up with a firewall, that blocks ICMP messages. Possible solution would be using ARP scanning instead, but it doesn't work if used outside of scanned host's subnet.

4.2 TCP SYN scanning

This technique is often referred to as „half-open" scanning, because no full TCP connection is established. Isamon sends a SYN packet and waits for the response. According to RFC 793, if a port is listening and ready so set up a connection, a SYN/ACK should be sent. So if a SYN/ACK is recieved, Isamon outputs the port as open and closes socket. Then a RST is immidiately sent by kernel to tear down the connection. If no SYN/ACK was recieved within a user-defined timeout, the port is considered closed.

Source: [6]

4.3 UDP ICMP port unreachable scanning

UDP scanning is significantly more difficult than scanning methods described above. UDP protocol is not reliable, open UDP ports don't have to send an acknowledgement in response of any communication attempts, and closed ports aren't required to send any error packets. Fortunately, most hosts do send an ICMP_PORT_UNREACH error when you send a packet to a closed UDP port. Thus you can find out if a port is not open, and by exclusion determine which ports which are. But neither UDP packets, nor the ICMP errors are guaranteed to arrive. Possible solution to this problem is to implement retransmission of packets that appear to be lost.

Source: [6]

Chapter 5

Conclusion

When creating this project, several port scanning methods were studied and compared in both speed and reliability. Nmap software was used as inspiration.

Isamon could be improved in several ways. As mentioned in Chapter 4.1, ARP scan could be implemented along with ICMP echo scan, but was not because of time limits. Instead of TCP SYN scan, a more reliable TCP connect() scanning could be used. But this method is a lot slower, since it establishes complete 3-way handshake and could not respect user-defined timeouts. UDP ICMP port unreachable scan was proved to be fast, but completely unreliable. Possible solution could be implementing packet retransmition. This solution was also not implemented because of the time limit and could be delievered with later patches.

Bibliography

1 RFC 950 [online]. Available at: https://tools.ietf.org/html/rfc950

2 RFC 792 - Internet Control Message Protocol [online]. Available at: https://tools.ietf.org/html/rfc792

3 RFC 768 [online]. Available at: https://www.ietf.org/rfc/rfc768

4 RFC 793 - Transmission Control Protocol [online]. Available at: https://tools.ietf.org/html/rfc793

5 Google Public DNS [online]. Available at: https://developers.google.com/speed/public-dns/

6 Nmap documentation [online]. Available at: https://nmap.org/nmap_doc.html

7 The TCP/IP Checksum [online]. Available at: https://locklessinc.com/articles/tcp_checksum/

8 Tcp syn portscan code in C with Linux sockets [online]. Availale at: http://www.binarytides.com/tcp-syn-portscan-in-c-with-linux-sockets/