
Chapter 6. Enumerating Target

Enumerating target is a process that is used to find and collect information about ports, operating systems, and services available on the target machines. This process is usually done after we have discovered that the target machines are available. In penetration testing practice, this task is conducted at the time of the discovery process.

In this chapter, we will discuss the following topics related to the target enumeration process:

- A brief background concept describing port scanning and various port scanning types supported by the port scanning tools
- The tools that can be used to carry out network scanning task
- The tools that can be used to do SMB enumeration on the Windows environment
- The tools that can be used to do SNMP enumeration
- The tool that can be used to enumerate the IPsec VPN server

The goal of performing the enumeration process is to collect information about the services available on the target systems. Later on, we will use this information to identify vulnerabilities that exist on these services.

Introducing port scanning

In its simplest definition, port scanning can be defined as a method used to determine the state of the **Transmission Control Protocol** (**TCP**) and **User Datagram Protocol** (**UDP**) ports on the target machines. An open port may mean that there is a network service listening on the port and the service is accessible, whereas a closed port means that there is no network service listening on that port.

After getting the port's state, an attacker will then check the version of the software used by the network service and find out the vulnerability of that version of software. For example, suppose that server A has web server software Version 1.0. A few days ago, there was a security advisory released. The advisory gave information about the vulnerability in web server software Version 1.0. If an attacker finds out about server A's web server and is able to get the version information, the attacker can use this information to attack the server. This is just a simple example of what an attacker can do after getting information about the services available on the machine.

Before we dig into the world of port scanning, let us discuss a little bit of the TCP/IP protocol theory.

Understanding the TCP/IP protocol

In the TCP/IP protocol suite, there are dozens of different protocols, but the most important ones are TCP and IP. IP provides addressing, datagram routing, and other functions for connecting one machine to another, while TCP is responsible for managing connections and provides reliable data transport between processes on two machines. The IP is located in the network layer (layer 3) in the **Open Systems Interconnection** (**OSI**) model, whereas TCP is located in the transport layer (layer 4) of OSI.

Besides TCP, the other key protocol in the transport layer is UDP. You may ask what the differences between these two protocols are.

In brief, TCP has the following characteristics:

- **This is a connection-oriented protocol**: Before TCP can be used for sending data, the client and the server that want to communicate must establish a TCP connection using a three-way handshake mechanism as follows:

    1. The client initiates the connection by sending a packet containing a SYN (synchronize) flag to the server. The client also sends the **initial sequence number** (**ISN**) in the **Sequence number** field of the SYN segment. This ISN is chosen randomly.
    2. The server replies with its own SYN segment containing its ISN. The server acknowledges the client's SYN by sending an ACK (acknowledgment) flag containing the client's ISN + 1 value.
    3. The client acknowledges the server by sending an ACK flag containing the server ISN + 1. At this point, the client and the server can exchange data.
    4. To terminate the connection, the TCP must follow the given mechanism:

        1. The client sends a packet containing a FIN (finish) flag set.
        2. The server sends an ACK (acknowledgment) packet to inform the client that the server has received the FIN packet.
        3. After the application server is ready to close, the server sends a FIN packet.
        4. The client then sends the ACK packet to acknowledge receiving the server's FIN packet. In a normal case, each side (client or server) can terminate its end of communication independently by sending the FIN packet.

- **This is a reliable protocol**: TCP uses a sequence number and acknowledgment to identify packet data. The receiver sends an acknowledgment when it has received the packet. When a packet is lost, TCP will automatically retransmit it if it hasn't received any acknowledgment from the

receiver. If the packets arrive out of order, TCP will reorder them before submitting it to the application.

Applications that need to transfer files or important data use TCP, such as **Hypertext Transport Protocol** (**HTTP**) and **File Transfer Protocol** (**FTP**).

UDP has characteristics opposite to TCP, which are stated as follows:

- This is a connectionless protocol. To send data, the client and the server don't need to establish a UDP connection first.
- It will do its best to send a packet to the destination, but if a packet is lost, UDP will not automatically resend it. It is up to the application to retransmit the packet.

Applications that can bear the loss of some packets, such as video streaming and other multimedia applications, use UDP. The other well-known applications that use UDP are **Domain Name System** (**DNS**), **Dynamic Host Configuration Protocol** (**DHCP**), and **Simple Network Management Protocol** (**SNMP**).

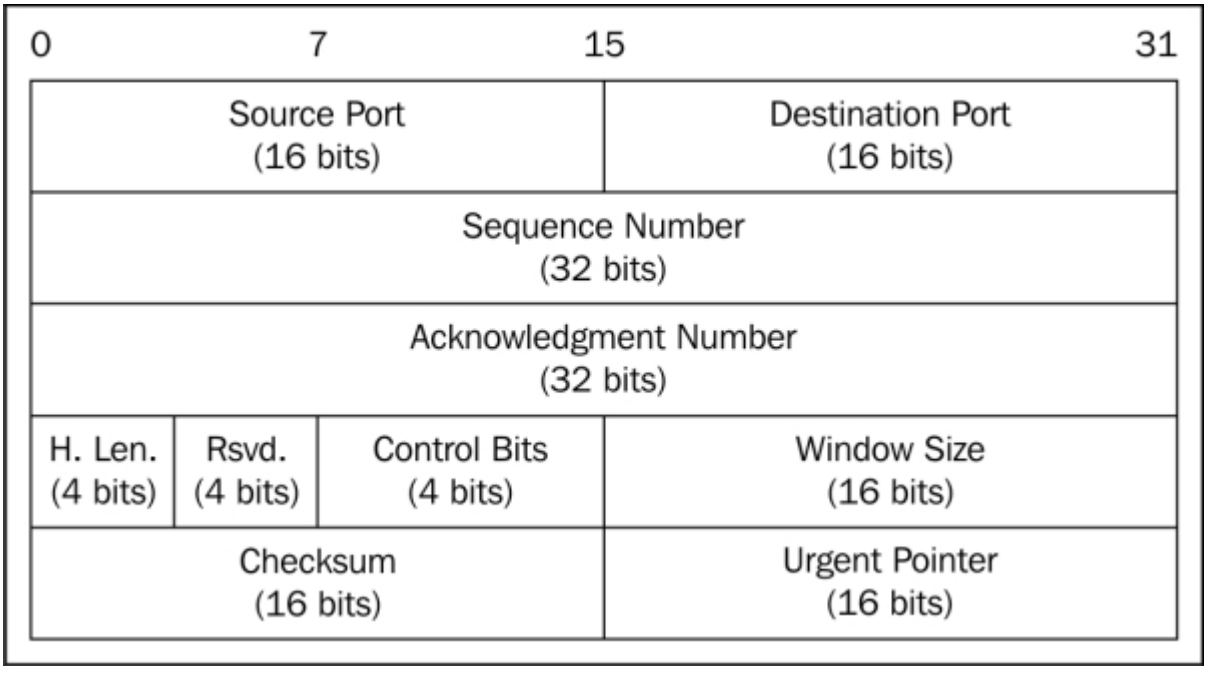
For applications to be able to communicate correctly, the transport layer uses addressing called ports. A software process listens on a particular port number on the server side, and the client machine sends data to that server port to be processed by the server application. The port numbers have a 16-bit address, and it can range from 0 to 65,535. To avoid a chaotic usage of port numbers, there are universal agreements on the port numbers' ranges as follows:

- **Well-known port numbers** (0 to 1023): Port numbers in this range are reserved port numbers and are usually used by the server processes that are run by a system administrator or privileged user. The examples of the port numbers used by an application server are SSH (`port 22`), HTTP (`port 80`), HTTPS (`port 443`), and so on.
- **Registered port numbers** (1024 to 49151): Users can send a request to the **Internet Assigned Number Authority** (**IANA**) to reserve one of these port numbers for their client-server application.
- **Private or dynamic port numbers** (49152 to 65535): Anyone can use port numbers in this range without registering themselves to IANA.

After discussing the differences between TCP and UDP in brief, let us describe the TCP and UDP message format.

Understanding the TCP and UDP message format

The TCP message is called a segment. A TCP segment consists of a header and a data section. The TCP header is often 20 bytes long (without TCP options). It can be described using the following figure:



Following is a brief description of each field:

- The **Source Port** and the **Destination Port** have a length of 16 bits each. The source port is the port on the sending machine that transmits the packet, while the destination port is the port on the target machine that receives the packet.
- The **Sequence Number (32 bits)**, in normal transmission, is the sequence number of the first byte of data of this segment.
- The **Acknowledgment Number (32 bits)** contains the sequence number from the sender increased by one.
- **H.Len. (4 bits)** is the size of the TCP header in 32-bit words.
- **Rsvd**. is reserved for future use. It is a 4-bit field and must be zero.

- The **Control Bits** (control flags) contains eight 1-bit flags. In the original specification (RFC 793; the RFC can be downloaded from http://www.ietf.org/rfc/rfc793.txt), the TCP only has six flags as follows:
    - **SYN**: This flag synchronizes the sequence numbers. This bit is used during session establishment.
    - **ACK**: This flag indicates that the **Acknowledgment** field in the TCP header is significant. If a packet contains this flag, it means that it is an acknowledgement to the previously received packet.
    - **RST**: This flag resets the connection.
    - **FIN**: This flag indicates that the party has no more data to send. It is used to tear down a connection gracefully.
    - **PSH**: This flag indicates that the buffered data should be pushed immediately to the application rather than waiting for more data.
    - **URG**: This flag indicates that the **Urgent Pointer** field in the TCP header is significant. The urgent pointer refers to important data sequence numbers.
- Later on, the RFC 3168 (the RFC can be downloaded from http://www.ietf.org/rfc/rfc3168.txt) added two more extended flags as follows:
    - **Congestion Window Reduced** (**CWR**): This is used by the data sender to inform the data receiver that the queue of outstanding packets to be sent has been reduced due to network congestion
    - **Explicit Connection Notification-Echo** (**ECN-Echo**): This indicates that the network connection is experiencing congestion
- **Window Size (16 bits)** specifies the number of bytes the receiver is willing to accept.
- **Checksum (16 bits)** is used for error checking of the TCP header and data.

The flags can be set independent of each other.

> **Note**
>
> To get more information on TCP, consult RFC 793 and RFC 3168.

When performing a port scanning on the TCP port by using a SYN packet to the target machine, an attacker might face the following behaviors:

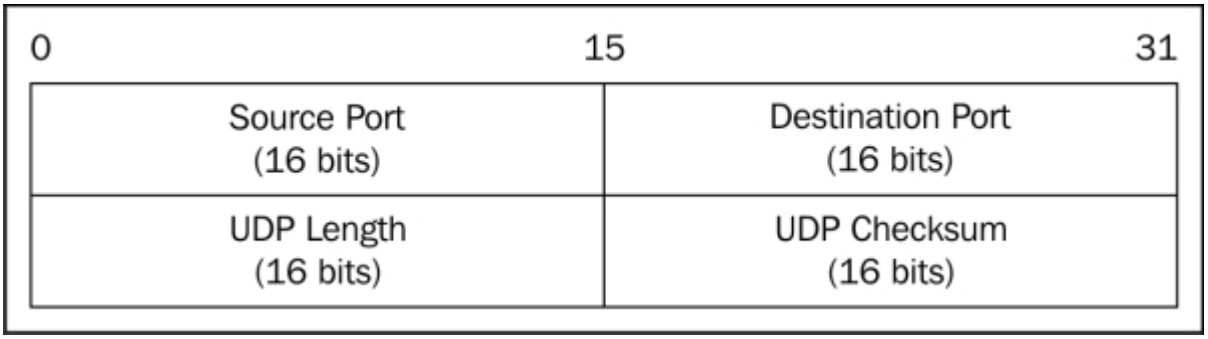
- The target machine responds with the SYN+ACK packet. If we receive this packet, we know that the port is open. This behavior is defined in the TCP specification (RFC 793), which states that the SYN packet must be responded with the SYN+ACK packet if the port is open without considering the SYN packet payload.
- The target machine sends back a packet with the RST and ACK bit set. This means that the port is closed.
- The target machine sends an ICMP message such as `ICMP Port Unreachable`, which means that the port is not accessible to us most likely because it is blocked by the firewall.
- The target machine sends nothing back to us. It may indicate that there is no network service listening on that port or that the firewall is blocking our SYN packet silently.

From a pentester's point of view, interesting behavior is when the port is open because this means that there is a service available on that port that can be tested further.

If you conduct a port scanning attack, you should understand the various TCP behaviors listed in order to be able to attack more effectively.

When scanning for UDP ports, you will see different behaviors, as will be explained later on.

Before we go to see various UDP behaviors, let's see the UDP header format first as shown in the following figure:



The following is a brief explanation of each field in the UDP header depicted in the preceding figure:

- Just like the TCP header, the UDP header also has the **Source Port** and the **Destination Port**, each of which has 16-bits length. The source port

is the port on the sending machine that transmits the packet, while the destination port is the port on the target machine that receives the packet.

- **UDP Length** is the length of the UDP header.
- **UDP Checksum (16 bits)** is used for error checking of the UDP header and data.

Note that there are no Sequence Number, Acknowledgement Number, and Control Bits fields in the UDP header.

During a port scanning activity to the UDP port on the target machine, an attacker might face the following behaviors:

- The target machine responds with a UDP packet. If we receive this packet, we know that the port is open.
- The target machine sends an ICMP message such as `ICMP Port Unreachable`. It can be concluded that the port is closed. However, if the message sent is not an ICMP unreachable message, it means that the port is filtered by the firewall.
- The target machine sends nothing back to us. This may indicate one of the following situations:
    - The port is closed
    - The inbound UDP packet is blocked
    - The response is blocked

UDP port scanning is less reliable when compared to TCP port scanning because sometimes, the UDP port is open but the service listening on that port is looking for a specific UDP payload. Thus, the the service will not send any replies.

Now that we have briefly described the port scanning theory, let's put this into practice. In the following sections, we will look at several tools that can be used to help us perform network scanning.

For the practical scenarios in this chapter, we will utilize a Metasploitable virtual machine, as explained in Chapter 1, *Beginning with Kali Linux*, as our target machine. It has an IP address of `192.168.56.103`, while our attacking machine has an IP address of `192.168.56.102`.