Understanding Ports and Protocols: A Beginner-Friendly Explanation

Hello readers! In this Medium post, I'd like to provide a clear and simple explanation of ports. If you're a beginner, this is one of the most essential topics to grasp, as ports play a critical role in enabling communication between devices.

To put it simply, you can think of ports as windows or doors in your house. Just as you use them to look outside, step out, or let someone in, devices use ports to send and receive data. Interestingly, there are over 65,000 ports available, offering a wide range of communication channels for different types of connections.

Now, let's clarify what protocols are. Protocols are sets of rules that devices follow to communicate with each other. Each protocol has its own characteristics, advantages, and limitations—much like how different languages function in human conversation.

It's important to note that not all 65,000 ports are the same. Just like a window and a door serve different purposes, ports differ based on the protocols they operate with.

In the context of ethical hacking, scanning for open or active ports is a common practice. Tools like Nmap help identify which services are running on a target device, making ports a crucial aspect of reconnaissance and security analysis.

Basically the ports are categorised in to 3 types:

The 3 Types of Ports by their **Port Numbers**

Ports are divided into 3 main categories:

A. Well-known Ports (0-1023)

These are reserved for major services (like "VIP apartments"):

- **Port 80** → HTTP (Web traffic)
- **Port 443** → HTTPS (Secure web traffic)
- **Port 22** → SSH (Secure remote login)
- **Port 21** → FTP (File transfers)
- **Port 53** → DNS (Domain name resolution)

🔹 **Fun fact**: Only admin/root can use these ports (they're privileged).

B. Registered Ports (1024-49151)

Used by apps that you install (like games, chat apps):

- **Port 3306** → MySQL database
- **Port 3389** → Remote Desktop (RDP)
- **Port 8080** → Alternate HTTP port

C. Dynamic/Private Ports (49152-65535)

These are temporary ports used by your computer when it connects to services (like a "random apartment for a guest").

- Example: When you browse the web, your PC picks a random port (e.g., 54321) to receive data back from a website.

3. TCP vs. UDP Ports

Both TCP and UDP use ports, but they work differently:

Feature	TCP (Transmission Control Protocol)	UDP (User Datagram Protocol)
Connection	Connection-oriented (handshake)	Connectionless (no handshake)
Reliability	Guarantees delivery (resends lost packets)	No guarantees (packets may be lost)
Speed	Slower (more checks)	Faster (no error recovery)
Use Cases	Web browsing (HTTP/HTTPS), Email (SMTP), File transfers (FTP)	Video calls (Zoom), Gaming, DNS queries

Key Takeaway:

- **TCP Port 80 (HTTP) ≠ UDP Port 80** (they're separate).
- Example:
    - **TCP 53** → Used for DNS zone transfers (reliable).
    - **UDP 53** → Used for quick DNS lookups (faster).

Ports = "Doors" for network traffic.
0-1023 = VIP ports (HTTP, SSH, FTP).
1024-49151 = App ports (MySQL, RDP).
49152+ = Temporary ports (used by your PC).
TCP = Reliable (web, email), UDP = Fast (gaming, streaming).

Common Ports and Associated Vulnerabilities/Attack Vectors:

- **HTTP/HTTPS (80/443):** Web server vulnerabilities, misconfigurations, directory traversal, SQL injection, XSS.
- **SSH (22):** Brute-force, weak keys, old ciphers.
- **FTP (21/20):** Anonymous access, weak credentials, vulnerable versions.
- **SMB/NetBIOS (139/445):** EternalBlue, unauthenticated shares, weak authentication.
- **DNS (53 TCP/UDP):** Zone transfers, cache poisoning.
- **Telnet (23):** Cleartext credentials, outdated.
- **RDP (3389):** Brute-force, NTLM relay.
- **SNMP (161/162 UDP):** Default community strings, information disclosure.
- **SMTP/POP3/IMAP (25/110/143):** Open relays, brute-force.
- **Database Ports (e.g., MySQL 3306, PostgreSQL 5432, MSSQL 1433, Oracle 1521):** SQL injection, weak credentials, unauthenticated access.
- **RPC/DCOM (135):** Windows service vulnerabilities.
- **NFS (2049):** Misconfigurations allowing unauthorized access.

Common Port Vulnerabilities for Pentesting

Knowing which ports are commonly exploited and what typical vulnerabilities they expose is crucial for effective pentesting.

Here's a breakdown by common port, as highlighted by your search results:

- **Port 20, 21 (FTP - File Transfer Protocol):**
    - **Vulnerabilities:** Plaintext transmission (credentials and data), anonymous access allowed, weak credentials, old vulnerable FTP server versions (e.g., vsftpd backdoor, ProFTPD vulnerabilities), directory traversal.
    - **Pentesting Focus:** Check for anonymous login, brute-force weak credentials, version enumeration for known exploits.
- **Port 22 (SSH - Secure Shell):**
    - **Vulnerabilities:** Brute-force attacks (especially against root or common usernames), weak SSH keys, outdated SSH versions with known vulnerabilities, insecure configurations (e.g., password authentication allowed when only key-based should be).

    - **Pentesting Focus:** Brute-force, identify allowed authentication methods, check for weak ciphers, look for public keys on exposed systems.

- **Port 23 (Telnet):**
    - **Vulnerabilities: Extremely insecure!** Plaintext transmission of credentials and data, usually replaced by SSH. Brute-force, weak default credentials.
    - **Pentesting Focus:** If found, it's a high-priority target for credential stuffing/brute-force as any traffic is visible.
- **Port 25 (SMTP - Simple Mail Transfer Protocol):**
    - **Vulnerabilities:** Open mail relays (allows anyone to send mail through the server, used for spam), user enumeration (VRFY, EXPN, RCPT commands), spoofing.
    - **Pentesting Focus:** Test for open relay, enumerate valid users.
- **Port 53 (DNS - Domain Name System TCP/UDP):**
    - **Vulnerabilities:** Zone transfers (if unauthenticated, can reveal internal network structure), DNS cache poisoning, DDoS amplification attacks.
    - **Pentesting Focus:** Attempt zone transfers, identify DNS server version.
- **Port 80 (HTTP) & 443 (HTTPS) (Web Servers):**
    - **Vulnerabilities: Vast and diverse.** SQL Injection, Cross-Site Scripting (XSS), Cross-Site Request Forgery (CSRF), Directory Traversal, Server-Side Request Forgery (SSRF), insecure deserialization, file upload vulnerabilities, misconfigurations (default pages, sensitive files exposed), outdated web server software (Apache, Nginx, IIS).
    - **Pentesting Focus:** This is a huge area! Enumerate web server version, content, check for common web app vulnerabilities, use tools like Burp Suite.
- **Port 135 (RPC - Remote Procedure Call):**
    - **Vulnerabilities:** Windows-specific. Can be used for service enumeration (e.g., with Impacket's rpcdump.py) and is a stepping stone for various Windows exploitation techniques.
- **Ports 137, 139 (NetBIOS) & 445 (SMB - Server Message Block):**
    - **Vulnerabilities:** Critical for Windows environments. Unauthenticated file shares, weak SMB authentication, **EternalBlue (SMBv1 vulnerability leading to RCE, e.g., WannaCry)**, NTLM relay attacks, SMB enumeration (user lists, share drives).
    - **Pentesting Focus:** Check for anonymous access to shares, enumerate shares, check for SMBv1 enabled, attempt NTLM relay.
- **Port 161, 162 (SNMP - Simple Network Management Protocol UDP):**

    - **Vulnerabilities:** Default community strings ("public," "private") allowing unauthenticated access to system information (OS, network interfaces, running processes, user accounts).
    - **Pentesting Focus:** Attempt to read sensitive information using common community strings.
- **Port 389 (LDAP) & 636 (LDAPS):**
    - **Vulnerabilities:** Anonymous bind allowed, insecure configurations, injection attacks.
    - **Pentesting Focus:** Attempt anonymous bind, enumerate users/groups if allowed.
- **Port 3306 (MySQL), 5432 (PostgreSQL), 1433 (MSSQL), 1521 (Oracle):**
    - **Vulnerabilities:** Weak credentials, SQL injection, unauthenticated access, default configurations, outdated database versions.
    - **Pentesting Focus:** Brute-force credentials, test for SQL injection on web apps, check for default databases/users.
- **Port 3389 (RDP - Remote Desktop Protocol):**
    - **Vulnerabilities:** Brute-force attacks (common), weak credentials, **BlueKeep (CVE-2019-0708 RCE vulnerability)**, RDP hijacking.
    - **Pentesting Focus:** Brute-force RDP, check for known RDP vulnerabilities if applicable.
- **Port 8080, 8443 (Alternative HTTP/HTTPS):**
    - **Vulnerabilities:** Often used for administrative interfaces or development servers, which might have default credentials, misconfigurations, or known vulnerabilities for specific applications (e.g., Apache Tomcat, Jenkins, JBoss, GlassFish).
    - **Pentesting Focus:** Treat them like ports 80/443 but with an added focus on identifying the specific application.
- **Port 2049 (NFS - Network File System):**
    - **Vulnerabilities:** Misconfigurations allowing unauthorized read/write access to network shares.
    - **Pentesting Focus:** Check for insecure exports, attempt to mount shares.

In the field of **ethical hacking**, it is important to understand that there are various types of **protocols** such as **SSH, Telnet, NFS, SMB**, and many others. Correspondingly, there are also a variety of **tools available** to interact with and analyze these specific protocols effectively.

Understanding **why these protocols differ from one another**—whether due to their **unique characteristics** or the **distinct ways they facilitate communication between devices**—can be incredibly valuable in the field of ethical hacking.Below mentioned services are very much help now I'll explain why we need to learn all of this

Once you've learned about **ports**, the next step in ethical hacking is to **scan and identify the services running on those ports**. For example, in many **TryHackMe exercises**, you'll often need to work with these discovered services. You might wonder, *how do we actually use this information?*

Think of it like unlocking your phone with a password. Similarly, once you identify a service—for instance, **SMB (Server Message Block)**—you can use the appropriate tool, such as **smbclient**, to interact with it. You provide the **target IP address** and attempt a connection. If the service prompts for authentication and you **know or crack the correct credentials**, you gain access to that service or potentially the underlying system. It's that straightforward in concept.

However, in practice, there are challenges—**firewalls** being one of the most significant. A firewall can **block port scanning attempts** or **filter out specific traffic**, making it harder to detect or access services. This is where your problem-solving skills come into play. While I'll continue focusing on ports in this discussion, it's important to note that **there are always creative ways to bypass restrictions**—though **ethically** and **legally**. The goal is not to break systems, but to **understand how they work** and **test their security responsibly**.

Phase 2: Service-Specific Deep Dives & Vulnerability Hunting

For each service type, your goal is to understand:

1. **How it works (Protocol Specification):** Beyond just "it's for file transfer."
2. **Common Default Configurations:** What's the "out-of-the-box" setup?
3. **Common Vulnerabilities (CVEs, Misconfigurations):** What weaknesses are often found?
4. **Enumeration Techniques:** How to extract maximum information.
5. **Exploitation Methods:** How to leverage findings.

A. Web Services (HTTP/HTTPS - Ports 80, 443, 8080, etc.):

- **Web Server Enumeration:**
    - **Banner Grabbing:** HTTP headers (Server, X-Powered-By).
    - **nikto / dirb / gobuster / ffuf:** Understanding these content discovery tools.
    - **Manual Directory Brute-Forcing:** Common web roots, backup files (.bak, .old), robots.txt, .git folders.
    - **Virtual Hosts:** Identifying multiple websites on one IP.
- **SSL/TLS Configuration:**
    - **sslscan / testssl.sh:** Identifying weak ciphers, old TLS versions (SSLv3, TLS 1.0/1.1), vulnerable renegotiation.
    - **Certificate Analysis:** Expiry, common name, Subject Alternative Names (SANs).
- **Web Frameworks/CMS Detection:** Identifying WordPress, Joomla, Apache Struts, etc., and their associated vulnerabilities.

- **API Enumeration:** Identifying REST, SOAP, GraphQL endpoints.
- **Web Application Firewall (WAF) Detection & Bypasses:** Identifying WAFs and understanding basic evasion techniques.

B. File Transfer Services (FTP - 21, SMB - 445/139, NFS - 2049):

- **FTP:**
    - **Anonymous Access:** Checking for anonymous:anonymous.
    - **Writable Directories:** Identifying directories that can be written to by anonymous or low-priv users.
    - **Outdated Versions:** Known CVEs for specific FTP daemons (e.g., vsFTPd backdoor).
- **SMB (Server Message Block / CIFS): CRITICAL for Windows networks.**
    - **Anonymous/Null Session Access:** Checking for unauthenticated share enumeration.
    - **Shared Drives/Folders:** Listing accessible shares (smbclient -L, enum4linux, crackmapexec).
    - **Permissions:** Identifying writable shares for file upload or code execution.
    - **Vulnerable SMB Versions:** EternalBlue (MS17-010), SMBGhost (CVE-2020-0796).
    - **Pass-the-Hash/Pass-the-Ticket Implications:** How SMB allows NTLM/Kerberos authentication.
- **NFS (Network File System):**
    - **Exported Shares:** Listing available shares (showmount -e).
    - **no_root_squash:** If root on the client can act as root on the server (huge vulnerability).
    - **Writable Shares:** Similar to SMB.

C. Remote Access Services (SSH - 22, RDP - 3389, Telnet - 23):

- **SSH:**
    - **Banner Grabbing:** SSH version string (e.g., SSH-2.0-OpenSSH_7.6p1).
    - **Weak Cipher Suites:** Identifying outdated or insecure key exchange algorithms.
    - **Brute-Force/Password Spraying:** Effective credential attacks.
    - **Public Key Authentication:** Checking for authorized keys.
    - **User Enumeration:** Techniques to determine valid usernames.
    - **Subsystem Exploitation:** Understanding SSH forwarding, SOCKS proxy.
- **RDP:**
    - **NLA (Network Level Authentication):** How it affects brute-forcing.
    - **Credential Guessing/Brute-Force.**

    - **Vulnerabilities:** BlueKeep (CVE-2019-0708) and other critical RDP vulnerabilities.
- **Telnet:** Generally insecure (cleartext). Prioritize exploitation if found.

D. Directory Services (LDAP - 389/636, DNS - 53):

- **LDAP (Active Directory): CRITICAL for enterprise networks.**
    - **Anonymous Binds:** Checking if anonymous queries are allowed.
    - **Enumerating Users, Groups, Computers, OUs:** Using ldapsearch, ldp.exe, ADExplorer, BloodHound.
    - **SPN (Service Principal Name) Enumeration:** For Kerberoasting.
    - **Kerberos Pre-Authentication Issues:** For AS-REP Roasting.
    - **Active Directory Certificate Services (AD CS) Misconfigurations:** If exposed.
- **DNS:**
    - **Zone Transfers:** Attempting unauthorized zone transfers (dig axfr).
    - **Subdomain Enumeration:** Using passive and active techniques (Brute-force, wordlists, permutations, search engine dorking).
    - **DNS Cache Poisoning/Spoofing:** Understanding DNS attacks.
    - **DNSSEC:** Understanding its role in securing DNS.

E. Database Services (MySQL - 3306, PostgreSQL - 5432, MSSQL - 1433, MongoDB - 27017, Redis - 6379):

- **Banner Grabbing:** Identifying database version.
- **Default/Weak Credentials.**
- **Common Vulnerabilities:** Specific CVEs for versions.
- **SQL Injection (if web-facing):** This is more web app, but database access is the goal.
- **Command Execution:** SQL injection often leads to OS command execution.
- **NoSQL Specifics:** JSON parsing issues, query injection for NoSQL databases.

F. Other Common Services:

- **SNMP (161/162):**
    - **Default Community Strings (public/private):** Huge information disclosure.
    - **Network Mapping:** Getting hostnames, network interfaces, running services, even device configurations.
- **SMTP (25, 465, 587):**
    - **Open Relays:** If the server forwards mail for anyone.
    - **User Enumeration:** VRFY, EXPN commands.

    - **Header Analysis:** Email spoofing.
- **POP3/IMAP (110, 143, 993, 995):**
    - **Credential Brute-Force.**
    - **Cleartext Authentication.**
- **RMI/JMX (Java Remote Method Invocation / Java Management Extensions):** Common in enterprise Java applications, often misconfigured.

In **ethical hacking**, the first step often involves **enumeration**, and **port scanning** plays a critical role in this phase. Using tools like **Nmap**, we scan a target device by providing its **IP address**, which reveals a list of **active services** running on that system. (Of course, this assumes that no **firewall** is actively blocking or filtering your scan.)

Once we identify the services running on the device, the next logical step is to **research each service**—especially to look for known **misconfigurations or vulnerabilities**. A quick search on **Google** or public vulnerability databases can provide valuable insights that may be used later during the **exploitation phase** of an engagement.
