Click Here

UFW stands for “Uncomplicated Firewall” is the default firewall tool for Debian based operating systems. It is an alternative program to iptables that simplifies the process of configuring and managing the firewall. Generally, iptables is a very advanced tool with powerful functionality, but it’s syntax is very complex and difficult for beginners. While UFW provides user-friendly syntax that makes it easier to manage your firewall. UFW also provides a user-friendly front-end for managing iptables firewall rules. UFW may be the appropriate solution for you are looking to secure your network easily. In this tutorial, we will show you how to install and use UFW firewall on Linux. Requirements A server running Ubuntu operating system. A root password is configured on your system. Install UFW By default, UFW is installed in most Ubuntu operating systems. If not installed, you can install it by running the following command: apt-get install ufw -y Once the installation is completed, you can check the status of UFW firewall with the following command: ufw status You should see the following output: Status: inactive You can check all the options available with UFW using the following command: ufw --help You should see the following screen: Configure UFW Default Policies By default, UFW is configured to allow all outgoing connections and deny all incoming connections. Before started with your UFW firewall, set your UFW firewall rules back to the defaults. You can set the default policy with the following commands: ufw default deny incoming ufw default allow outgoing Now, anyone trying to access your server can not be able to connect your server. Next, you will need to activate the UFW firewall to apply the changes. Before activating the UFW firewall, it is recommended to allow incoming SSH connection to access your server from a remote location. You can allow incoming SSH connection with the following command: ufw allow ssh Or ufw allow 22 If your SSH is configured to listen on a different port (2222)then run the following command: ufw allow 2222 Now, you can enable the UFW firewall using the following command: ufw enable Allow incoming Connections There are several ways to allow incoming connections like, allow specific port, allow specific port range, allow specific IP address, allow specific subnet and allow specific network interface. Allow Specific Port By default UFW is configured to deny all incoming connections. So you will need to allow specific port as per your need. For example, to allow incoming HTTP connections run the following command: ufw allow http You can also specify port number instead service name as shown below: ufw allow 80/tcp To allow incoming connection on UDP port 21 run the following command: ufw allow 21/udp You can see a sample output of the above commands in the following screen: You can also check the status of UFW firewall with the following command: ufw status You should see the following screen: Allow Specific Port Range Instead of allowing single ports UFW also allows you to specify multiple port numbers in a single command. For example, you can allow incoming connections from ports 8000 to 9000 with the following command: ufw allow 8000:9000/tcp To allow UDP port range 8000 to 9000, run the following command: ufw allow 8000:9000/udp Allow Specific IP Address UFW allows you to access all ports from a specific IP address. For example, if you want to allow all incoming connections from the IP address 192.168.0.100 run the following command: ufw allow from 192.168.0.100 You can also allow access to specific port (8080) from the specific IP address (192.168.0.101). ufw allow from 192.168.0.101 to any port 8080 Allow Specific Subnet You can also allow incoming connections from a range of IP addresses. For example, to allow all incoming connections from the IP address 192.168.0.1 to 192.168.0.254 run the following command: ufw allow from 192.168.0.0/24 You can also specify the destination port 8089 that the subnet 192.168.0.0/24 is allowed to connect to as shown below: ufw allow from 192.168.0.0/24 to any port 8089 Allow Specific Network Interface If you want to add firewall rules that only apply to a specific network interface (eth0). For example, allow all incoming connections to the network interface eth0 run the following command: ufw allow in on eth0 You can allow access on a specific port 9000 only to specific interface eth0 by running the following command: ufw allow in on eth0 to any port 9000 Deny incoming Connections You can deny all incoming connections on port 80, 443 and 25 with the following command: ufw deny 80/tcp ufw deny 443/tcp ufw deny 25/tcp To deny all incoming connections from specific subnet, run the following command: ufw deny from 192.168.1.0/24 To deny all incoming connections from specific IP address, run the following command: ufw deny from 192.168.0.102 List Application Profile By default, all application profile saved in the /etc/ufw/applications.d directory. You can list all application profile with the following command: ufw app list You should see the following output: Available applications: Apache Apache Full Apache Secure CUPS Nginx Full Nginx HTTP Nginx HTTPS OpenSSH You can also see the information about a specific application with the following command: ufw app info Apache You should see the following output: Profile: Apache Title: Web Server Description: Apache v2 is the next generation of the omnipresent Apache web server. Port: 80/tcp You can see a sample output of above commands in the following screen: Delete UFW Rules There are two ways you can delete UFW rules, with the actual rule and with rule numbers. The first method to delete a rule by specifying the actual rule. For example, to delete a rule for port 80 and 443 run the following command: ufw delete allow 80 ufw delete allow 443 The second method to delete a rule by specifying rule numbers. First, you will need to find the number of the rule you want to delete. You can list all the rules with numbers as shown below: ufw status numbered You should see the following output: Status: active To Action From -- ------ ---- [1] 22 ALLOW IN Anywhere [2] 21/udp ALLOW IN Anywhere [3] 8000:9000/tcp ALLOW IN Anywhere [4] Anywhere on wlan0 ALLOW IN Anywhere (v6) [5] 22 (v6) ALLOW IN Anywhere (v6) [6] 21/udp (v6) ALLOW IN Anywhere (v6) [7] 8000:9000/tcp (v6) ALLOW IN Anywhere (v6) [8] Anywhere (v6) on wlan0 ALLOW IN Anywhere (v6) Now, delete the rule number 6 with the following command: ufw delete 6 To delete the rule number 3, run the following command: ufw delete 3 You can see the sample output of all commands in the following screen: Enable UFW Logging Enabling UFW logging is very useful to troubleshoot your firewall rules. You can enable logging with the following command: ufw logging on You can also disable the UFW logging any time with the following command: ufw logging off By default, UFW logs are stored in the file /var/log/ufw.log. You can see it with the following command: tail -f /var/log/ufw.log Reset & Disable UFW If you want to revert all of your changes that you have made with UFW, run the following command: ufw reset The above command will disable UFW and delete all active rules. In some cases, you want to disable and deactivate all the rules. You can do it with the following command: ufw disable Conclusion In the above tutorial, we learned how to install, configure and working with UFW firewall. I hope you have now enough knowledge to secure your system with the UFW. Network and Server administration is one of the key areas in which Linux is actually preferred to any other operating system. Hence most data center admins are well versed with the Linux command line. There can be scenarios when certain ports on a server, which were required to be closed, are open and causing unexpected traffic on the server. Today, we will learn how to find and close an open port in Linux. Finding Open Ports in Linux For finding the open ports, we will make use of the ss command, which is preinstalled in most common Linux distributions and it is now the replacement for the previously very popular netstat command. Let’s run the ss command with the following syntax, to get all listening TCP sockets: $ ss -tl Here, the 't' stands for TCP and the 'l' stands for Listening sockets. Find Listening TCP Sockets Similarly to get all listening UDP ports, run: $ ss -ul Find Listening UDP Sockets Observe the output. In the ‘State’ column for UDP, all the ports have state UNCONN, i.e., unconnected. Since we only need the ports which are actively listening, we pipe the output and filter it with the grep command. $ ss -ul | grep LISTEN We can also combine the TCP and UDP output together. $ ss -tul | grep LISTEN Find Listening Ports in Linux Another addition that can be done here is: the argument '-n'. In the output above, the command is resolving the name of the service associated with a port: Eg. HTTP, IPP, etc. With the argument '-n' it just shows the port number which is listening. $ ss -tuln | grep LISTEN Find Open Ports in Linux Closing Open Ports in Linux The manual way to close an open port in Linux is quite tedious and programmatic. Hence, we will use the easier approach: to close the processes which are listening on the port. We need to call ss with another argument, '-p' to list the process which is using each port (run the command as a sudo user). $ sudo ss -tulnp | grep LISTEN Find Process with Port Number As shown above, the last column lists the ‘users’, i.e., the processes which use the port number. Now you can close the port by terminating the process using it. For example, to close port 80, we have to stop the process ‘Apache2’. $ sudo service apache2 stop OR $ sudo systemctl stop apache2 Close Open Ports in Linux Thus, we have successfully closed port 80 and no process is listening on it any longer. Conclusion In this article, we learned how to find and close an open port in Linux. Many ports like 22 (SSH), 21 (Telnet), etc. should be kept closed at most times, as these are the ports from which cyber attacks can arise. Other ports should also be closed when the process of using them is no longer required. Thanks for reading and let us know your thoughts in the comments below! So you are dealing with a critical server where you have to maintain security at any cost. And closing ports to block unwanted traffic is the first step you'd take.Find open ports in LinuxIn this tutorial, I am going to use the ss command to find open ports. You can use the -l option with the ss command to get listening ports. But to be more specific, I'm going with -lt to get listening TCP ports:ss -tlSimilarly, if you want to have a list of both TCP and UDP in the listening state, you can use the given command:ss -tulAnd to get the listening port of each service, you can use -n and for more fine-tuned results, you can always use the grep command:ss -tuln | grep LISTENEnough of finding open ports, let's jump to how you can close them.Close open ports in LinuxTo close the port, first, you will need to stop the service and to find the service name, you can use the same ss command with -p option:sudo ss -tulnp | grep LISTENAs you can see, the NGINX is utilizing port number 80. So let's stop it using the given command:sudo systemctl stop nginxAs it will enable itself on every boot and you can alter this behavior using the given command:sudo systemctl disable nginxFor better results, I would recommend changing firewall rules.Here, I'm going to block port no 80 (used by NGINX) in UFW (which is pre-installed in Ubuntu).First, let's check the status of UFW:sudo ufw statusAnd if it shows inactive, you can use the given command to enable it:sudo ufw enableNow, you just have to pair the deny option with the port number:sudo ufw deny 80And here's the end result:No sign of NGINX! Wrapping UpThis was my take on how you can find and close open ports in Linux. I hope you will find this helpful.And if you have any queries, let me know in the comments. In this tutorial we’ll learn how to check for listening ports, using numerous tools, along with managing ports by allowing or disallowing incoming and outgoing connections. Ports in Ubuntu What is a port? In simple words: a door to a program running in your operating system. Or: application-specific or process-specific software construct used as a numeric identifier of a particular connection between two applications. Port numbers is a 16-bit unsigned integer that range from 0 to 65535. Applications listen for ports to achieve a successful communication from the outside. When dealing with a well-known distribution as Ubuntu, there are multiple tricks and features that check for, close or open ports. So as an alert Linux user, it’s imperative to be aware of probe for open ports in your system, which ones are open by default, closing open ports and allowing exceptions. If not, securities holes and system’s vulnerabilities would be the least of your problems, not to mention bandwidth and resource consuming connections. Check common ports: Port Name Port Number/Protocol Alias ftp 21/tcp // -ssh 22/tcp // -smtp 25/tcp mail domain 53/tcp nameserver domain 53/udp nameserver http 80/tcp www www-http -https 443/tcp // pop3 110/tcp pop-3 How to Inspect Listening Ports Obviously, before even starting to open or close ports, it’s necessary to be familiar with which ports are open in your system. To do that, we can use various built-in command line utilities or installed. Fire up your machine, open the terminal, then type the next command to list running services and which ports are used. less /etc/services ftp 21/tcp fsp 21/udp fspd ssh 22/tcp # SSH Remote Login Protocol telnet 23/tcp smtp 25/tcp mail time 37/tcp timserver time 37/udp timserver whois 43/tcp nicname When you want to quit less, hit the q key. Suppose we want to check for specific ports (80, 443 and 22). Easily use the grep tool like so: grep -we 80 -we 22 -we 443 /etc/services ssh 22/tcp # SSH Remote Login Protocol http 80/tcp www # WorldWideWeb HTTP https 443/tcp # http protocol over TLS/SSL https 443/udp # HTTP/3 The next command is using netstat, which is a well known utility that can be used to inspect listening ports and socket information. There are multiple variations of using this tool, but we will be ok by the next (tcp/udp): netstat -intu We can also be using the ss tool, which is very similar to netstat. ss -intu lsof is a command line utility for listing open files, but can be used to check what process and tool is listening on a specific port; or by using the protocol. lsof -i :80 lsof -i udp Now to one of the major tools in the port scanning field: Nmap. We can also use this tool in a variety of ways, but we will see only the following example: nmap localhost Note that the utilities above will only display the port if a service or a process is actually listening for incoming connections, (if port is in use). But keep in mind, this does not mean that the listening service is open to the internet, since our firewall could be blocking incoming connections for certain ports. How to Open Ports Generally to open or close ports on Ubuntu we use ufw command (Uncomplicated Firewall); which is a frontend for iptables. Before starting to manage our ports, we have to check the ufw statues by running the next command: sudo ufw status verbose Status: inactive Enable your firewall as so: sudo ufw enable Firewall is active and enabled on system startup Run the first command: sudo ufw status verbose Status: active Logging: on (low) Default: deny (incoming), allow (outgoing), disabled (routed) New profiles: skip As you can see, our firewall is denying incoming connections. So if we want to add an exception (22 tcp), we should run the below mentioned command. sudo ufw allow 22/tcp Rule added Rule added (v6) Check if our firewall is indeed making an exception for port 22: sudo ufw status verbose Status: active Logging: on (low) Default: deny (incoming), allow (outgoing), disabled (routed) New profiles: skip To Action From -- ------ ---- 22/tcp ALLOW IN Anywhere 22/tcp (v6) ALLOW IN Anywhere (v6) In the even we want to allow a specific IP address to connect to port 22: sudo ufw allow from 40.200.14.5 to any port 22 Rule added We could use the same previous command for a subnet of IP addresses: sudo ufw allow from 40.200.14.0/24 to any port 22 Rule added We checked managing incoming connections; for allowing outgoing connections we use the out option: sudo ufw allow out 22/tcp Rule added Rule added (v6) Let’s check the iptables examples for opening ports. First let’s make an exception for incoming connections to port 80: sudo iptables -I INPUT -p tcp --dport 80 -j ACCEPT This second command for making an exception for outgoing connections to port 80: sudo iptables -I OUTPUT -p tcp --sport 80 -j ACCEPT How to Close Ports Like opening ports, there are numerous commands for closing ports. Upon discovery of an open port that should be closed. The easiest way as before is using ufw. Let’s start by blocking port 22: sudo ufw deny 22 Rule added Rule added (v6) We could use the reject keyword instead: sudo ufw reject 22 Note that reject and deny options achieve similar results, the key difference though, is that reject informs back the sender that their connection was rejected via error packet. As before, in case we want to block outgoing call, we use the following command: sudo ufw deny out 22 Rule added Rule added (v6) Let’s try iptables to block incoming connections: sudo iptables -A INPUT -p tcp --dport 80 -j DROP Just like ufw, DROP keyword is used to block connections without sending back an error packet. sudo iptables -A INPUT -p tcp --dport 80 -j REJECT The REJECT option, for informing the sender that their request has been rejected. When trying to block users from using a specific port (outgoing), we use the following: sudo iptables -A OUTPUT -p tcp --sport 80 -j DROP Bonus: In case that you missed around with your firewall (adding and dropping a whole lot of exceptions). Do not worry; just run the command mentioned below to start over. sudo ufw reset Resetting all rules to installed defaults. Proceed with operation (y|n)? y Backing up 'user.rules' to '/etc/ufw/user.rules.20220503_211753' Backing up 'before.rules' to '/etc/ufw/before.rules.20220503_211753' Backing up 'after.rules' to '/etc/ufw/after.rules.20220503_211753' Backing up 'user6.rules' to '/etc/ufw/user6.rules.20220503_211753' Backing up 'before6.rules' to '/etc/ufw/before6.rules.20220503_211753' Backing up 'after6.rules' to '/etc/ufw/after6.rules.20220503_211753' Conclusion In this how to article we learned about checking which ports are used by our services and programs, we did that by testing different utilities and command lines. We also learned about managing ports with ufw and iptables; we’ve managed that by being acquainted with executing allow exceptions, deny/reject exceptions and how to run them for different scenarios. On a Linux system, closing a particular port might help increase security or free up system resources. This short tutorial will walk you through closing a port using IPtables, FirewallD, UFW, or stopping the service. If you are looking for a tutorial to open a port in Linux, please check our tutorial: How to Open a Specific Port on Linux First, let’s see which ports are open. You can use one of these commands: sudo netstat -tuln Below is a sample output in which underlined in red are the ports which are opened. (22 and 53) sudo ss -tuln Below is a sample output in which underlined in red are the ports which are opened. (22 and 53) The lsof command is mostly not preinstalled on Linux systems. You need to manually install this command to use. If you are using a Debian-based system like Ubuntu, use the command: sudo apt-get install lsof to install. For Centos/Redhat: sudo yum install lsof and for Fedora: sudo dnf install lsof sudo lsof -i -P -n Below is a sample output in which underlined in red are the ports which are opened. (22 and 53) These commands will list all the open ports on your system. For Ubuntu and other Debian-based systems, UFW is a straightforward firewall management application. sudo apt-get install ufw sudo ufw enable Replace with the port number, you want to close. sudo ufw deny Example screenshot: sudo ufw status A more sophisticated tool for controlling firewall rules is IPtables. To install IPtables use the command: sudo apt-get install iptables if it was not installed. Replace with the port number, you want to close. sudo iptables -A INPUT -p tcp --dport -j DROP sudo iptables -A INPUT -p udp --dport -j DROP Example screenshot: On Debian-based systems: sudo sh -c "iptables-save > /etc/iptables/rules.v4" sudo service iptables save Firewalld is the default firewall tool for CentOS and Fedora. sudo yum install firewalld Replace with the port number, you want to close. sudo firewall-cmd --permanent --remove-port=/tcp sudo firewall-cmd --permanent --remove-port=/udp Example screenshot: sudo firewall-cmd --reload sudo firewall-cmd --list-all Sometimes, a port is open because a service is using it. Stopping the service will close the port. In that case, blocking the service will be the best option rather than closing the port. Use netstat, ss, or lsof to find which service is using the port. sudo netstat -tuln | grep OR sudo ss -tuln | grep OR sudo lsof -i : For Example: Replace with the actual port number. Replace with the name of the service. sudo systemctl stop For example: This prevents the service from starting at boot. sudo systemctl disable On Linux, there are multiple methods to close a port: UFW, IPtables, Firewalld, or terminating a service. Select the approach that best suits your needs. For simplicity, UFW is a great choice. For more control, IPtables is ideal. Firewalld is perfect for Red Hat-based distributions, and stopping a service directly is effective when dealing with a specific application. Always ensure you correctly identify the service using the port before stopping it. Experience unparalleled speed and reliability. Optimize your digital presence for maximum impact. Related Blogs: How to Open a Specific Port on Linux How to Find the Process Which Uses the Port in Linux and Windows? Managing network ports is a fundamental aspect of securing a Linux server. In Ubuntu, the Uncomplicated Firewall (UFW) provides a user-friendly way to configure your firewall settings. This tutorial walks you through the process of opening and closing ports on your Ubuntu system using UFW, with a step-by-step approach that ranges from basic to advanced configurations.PrerequisitesAn Ubuntu server or desktopAccess to a user account with sudo privilegesUFW installed (usually installed by default on Ubuntu)Basic Commands to Manage Ports with UFWBefore making any changes, it’s good practice to check the status of your UFW firewall. Use the following command:sudo ufw status verbose If UFW is inactive, you can enable it with the command:sudo ufw enable To open a port, you need to allow traffic through it. For example, allowing traffic through port 22 (SSH) can be done with the command:sudo ufw allow 22 If you wish to close the port, you can deny traffic through it like so:sudo ufw deny 22 Allowing Specific Port Ranges and ProtocolsTo allow a range of ports, for instance, ports 3000-3005, use:sudo ufw allow 3000:3005/tcp If you want to specify the protocol for a single port, you can add it at the end of your command:sudo ufw allow 53/udp Using UFW with IPv6By default, UFW is configured to support IPv6. To allow a port for both IPv4 and IPv6, the commands are the same:sudo ufw allow 80 If you specifically want to deny a port for IPv6, you can directly specify this:sudo ufw deny 80 from any to any proto tcpAdvanced UFW ConfigurationFor more advanced scenarios, you might want to specify a particular IP address for the rule. For example, to allow access to port 22 only from the IP 192.168.1.100, you can do:sudo ufw allow from 192.168.1.100 to any port 22 To allow traffic to a specific network interface, such as eth0, you could specify:sudo ufw allow in on eth0 to any port 80 Managing UFW RulesYou can list all the current UFW rules with the command:sudo ufw status numbered To remove a rule by its number, you can do the following. For example, to remove the rule number 2:sudo ufw delete 2 Remember to reload the firewall for changes to take effect:sudo ufw reload Configuration with Configuration FilesFor those who prefer to manually edit configuration files, UFW keeps its main configuration in /etc/default/ufw. To apply rule sets consistently across reboots, you can edit the /etc/ufw/before.rules and /etc/ufw/after.rules files.Be cautious when manually editing configuration files, as incorrect settings can lock you out of your system.Logging and MonitoringUFW provides an easy way to enable or disable logging:sudo ufw logging on sudo ufw logging off To monitor live firewall logs, you can use the command:sudo tail -f /var/log/ufw.log Using UFW with Application ProfilesUFW supports application profiles. If an application includes a UFW profile, you can allow or deny it using its profile name. To check available profiles, use:sudo ufw app list To allow an application called ‘Apache,’ do the following:sudo ufw allow 'Apache' Troubleshooting and TipsIf you encounter issues, you can reset UFW to its default settings by using:sudo ufw reset To ensure you don’t get locked out during configuration, UFW includes the ‘ufw limit’ command, which you can use to limit connections to a service and help prevent brute-force attacks:sudo ufw limit 22As a tip, when configuring your firewall, always make sure you have another means of accessing your server, such as physical access or an out-of-band management interface.ConclusionIn this guide, we’ve seen how to manage ports in Ubuntu’s Uncomplicated Firewall, from the basics of opening and closing ports to more advanced examples. Knowing how to properly manage your firewall is key to maintaining server security and ensuring authorized traffic flows properly. Ubuntu is a popular Linux distribution known for its user-friendly interface and robust security features. However, like any operating system, Ubuntu may have open ports that, if left unattended, can pose security risks. Open ports can potentially expose your system to unauthorized access and attacks. To enhance the security of your Ubuntu system, it’s essential to know how to close open ports. In this article, we’ll guide you through the process of identifying open ports and securing your system. Why Close Open Ports? Open ports are network access points that allow data to enter or leave your computer. While some ports are necessary for system services and applications to function correctly, open ports that you don’t actively use can be security vulnerabilities. Cyber attackers may exploit these open ports to gain unauthorized access to your system, potentially compromising your data and privacy. Here’s how to identify and close open ports in Ubuntu: 1. Identifying Open Ports Before closing open ports, you need to identify which ports are currently open on your system. You can use various tools and commands for this purpose. One of the most commonly used commands is `netstat`. Open your terminal and type: This command will display a list of open ports, both for TCP and UDP connections. You can also use the `ss` command, as mentioned in the previous response: The command “ss -tl” is used to display a list of listening TCP sockets on a Linux system. It shows information about TCP connections that the system is currently listening for. The “ss” command is a replacement for the older “netstat” command and provides detailed socket statistics. Here’s a breakdown of the command: – “ss”: This is the command to invoke the socket statistics utility in Linux. – “-t”: This option is used to display TCP sockets. – “-l”: This option is used to show only listening sockets. See the commands of the UFW firewall on Ubuntu Linux to list the open ports that are allowed or denied to be accessed from outside the network. UFW stands for uncomplicated firewall. The goal of UFW is to provide an uncomplicated command line-based frontend for the very powerful, but not exactly easy-to-configure IPtables. UFW supports both IPv4 and IPv6. It is quite easy to install because it is included in the package sources – at least if you are using an Ubuntu or Debian distribution. Also, those who are not familiar with the command line can use the graphical user interface known as GUFW, also available to install using the default repository of Ubuntu. It makes us easily manage inbound and outbound traffic using firewall rules. If you don’t want to know what are the ports already opened in your system firewall i.e UFW, simply open the command terminal and use the given command: sudo ufw status verbose Note: Ports with the “Allow in” action are opened whereas the ones with “Deny in” are closed. If there is any rule you have added to open some port in UFW and now want to delete that rule; for that, you can simply use the added rule’s serial number and a ‘delete’ parameter. List all ports with their serial number sudo ufw status numbered Now, the rule that you want to delete- note down its serial number and use it with the given command: For example, if you want to delete a rule specified with serial number 8, then the command will be: sudo ufw delete 8 Also, for those who want to know the command to open any new port using the UFW command line, then here is the syntax: sudo ufw allow port-number for example, if you want to open port 443, the above syntax will be: sudo ufw allow 443 In case, instead of deleting rules you just want to close the access of some particular port on your system from the outside network, then the syntax is: sudo ufw deny port-number Example: sudo ufw deny 443 Well, there is an easy way to manage rules in the UFW Firewall by installing a graphical user interface for it. If you are on Ubuntu then here is the command for that: sudo apt install gufw Other Articles: How to Open or close ports in AlmaLinux 8 or Rocky FirewallInstall and Configure UFW Firewall on Debian 11How To Install SFTPGo on Ubuntu 22.04 LTS Open ports are one of the main vulnerabilities of the server. If you don't manage them, you risk leaving the "door open" for intruders. In this article, we'll look at how to close ports on Linux using iptables, ufw, and see how to check which ports are already open. You will also learn how to close port 80, block all ports except the necessary ones, and how this relates to the security of servers and HTTP clients. Cloud Servers from $5 / mo Intel Xeon Gold 6254 3.1 GHz CPU, SLA 99,9%, 100 Mbps channel How do I view the open ports on the server? Before closing ports, you need to find out which ones are currently open and which services are using them. or: Flag values: t — TCP u — UDP l — LISTEN ports only n — show numeric port values How do I close/open the port via iptables? To close a port (for example, port 80), you can use iptables. This rule will block incoming connections to port 80 over TCP. For verification purposes: To remove this rule: If you previously closed the port and want to allow connections again: How to close or open a port via UFW UFW (Uncomplicated Firewall) is a simple tool for configuring the firewall in Linux, especially convenient in Ubuntu. To close a port, for example 80, use the command: This command will block incoming connections to the specified default TCP port. You can check the current rules like this: If you want to cancel one of the rules, find its number in the list and delete it.: To reopen port 80 for incoming connections: This way, you can easily control access to any port: close unnecessary ones and open the necessary ones as needed, ensuring server security without complex iptables rules. Script for configuring ports For port management, you can use automated configuration tools and scripts that will allow you to block all connections except those allowed. echo "Disabling ufw if it has already been enabled..." sudo ufw disable echo "Reset all rules..." sudo ufw reset echo "I'm setting the default policy: prohibit incoming, allow outgoing..." sudo ufw default deny incoming sudo ufw default allow outgoing echo "I allow HTTPS (port 443)..." sudo ufw allow 443/tcp echo "Enabling ufw..." sudo ufw enable echo "Current ufw status:" sudo ufw status verbose How to disable the service that opens the port Sometimes ports are open because a certain service is running (for example, Apache or Nginx). If you are not using it, disable it. Disabling Apache: Disabling Nginx: Safety tips Close all unused ports. Allow only the specific ports you need (for example, 22 and 443). Use fail2ban to protect against password brute force. Configure the firewall (UFW, firewalld, or iptables) immediately after installing the server. Port management is an important part of protecting a Linux server. Now you know how to close ports in Linux, view open ports, block port 80, and allow only the necessary connections. 33145 North Miami, FL 2520 Coral Way apt 2-135 +1 302 425-97-76 700 300 ITGLOBAL.COM CORP 33145 North Miami, FL 2520 Coral Way apt 2-135 +1 302 425-97-76 700 300 ITGLOBAL.COM CORP 700 300