https://wiki.micylou.com/index.php/Linux

Linux

Linux

Source:	Query
Language:	English
Topic:	Linux
SubTopic:	Query
Last Edit By:	DochyJP
LastEdit:	2021-04-14
Document type:	Table
Status:	Active
Access:	free

All semantic queries defined by the semantic variable [[Topic::DataSource]] or [[SubTopic::Linux]]

Here below are all documents about Linux present in this WIKI.

Result in a limited list ▼

Section	Source	Language	Topic	Subtopic	DocumentType	Status	Access
Enabling ActiveSync on Plesk on Linux	DataSource	English	Linux	Plesk	Documentation	Active	free
How To Secure Apache with Let's Encrypt on Ubuntu?	DataSource	English	Linux	Apache	User Guide	Active	free
How to Check if Your Linux Server is Under DDoS Attack	DataSource	English	Linux	Security	Documentation	Active	free
Install Citrix Workspace on Ubuntu	DataSource	English	Linux	Citrix	Documentation	Active	free
Install/upgrade/Downgrade Composer	DataSource	English	Mediawiki	Linux	Documentation	Active	free
Linux commands - Cheat Sheet	DataSource	English	Linux	Linux Commands	User Guide	Active	free
Linux fail2ban	DataSource	English	Fail2Ban	Linux	Documentation	Active	Private
Mediawiki backup script	DataSource	English	Mediawiki	Linux	Documentation	Active	Micylou restricted
Virtualbox Copy/Paste not working	DataSource	English	Linux	VirtualBox	Documentation	Active	free
WIKI - KMS - Crontabs	DataSource	English	Mediawiki	Linux	Documentation	Active	free
Section	**Source**	**Language**	**Topic**	**Subtopic**	**DocumentType**	**Status**	**Access**
Enabling ActiveSync on Plesk on Linux	DataSource	English	Linux	Plesk	Documentation	Active	free
How To Secure Apache with Let's Encrypt on Ubuntu?	DataSource	English	Linux	Apache	User Guide	Active	free

How to Check if Your Linux Server is Under DDoS Attack	DataSource	English	Linux	Security	Documentation	Active	free
Install Citrix Workspace on Ubuntu	DataSource	English	Linux	Citrix	Documentation	Active	free
Install/upgrade/Downgrade Composer	DataSource	English	Mediawiki	Linux	Documentation	Active	free
Linux commands - Cheat Sheet	DataSource	English	Linux	Linux Commands	User Guide	Active	free
Linux fail2ban	DataSource	English	Fail2Ban	Linux	Documentation	Active	Private
Mediawiki backup script	DataSource	English	Mediawiki	Linux	Documentation	Active	Micylou restricted
Virtualbox Copy/Paste not working	DataSource	English	Linux	VirtualBox	Documentation	Active	free
WIKI - KMS - Crontabs	DataSource	English	Mediawiki	Linux	Documentation	Active	free

Contents

Enabling ActiveSync on Plesk on Linux

Linux

Source:	DataSource
Language:	English
Topic:	Linux
SubTopic:	Plesk
Last Edit By:	DochyJP
LastEdit:	2021-04-14
Document type:	Documentation
Status:	Active
Access:	free

Download this page as PDF

I have activated ActiveSync on my Ubuntu server and it works pretty well. Now I want to share my experience and maybe get some new suggestions. Thanks to everyone who helped to achieve it, I learned a lot from many posts.

- Choose a domain and an email address for administration. It should be a domain with SSL (there are even free SSL providers); this is not necessary but better.
- Edit in the file /etc/psa-webmail/horde/horde/conf.php these lines:

```
$conf['auth']['admins'] = array('your-admin-email-address');
$conf['activesync']['enabled'] = true;
$conf['activesync']['version'] = '14.1';
```

- In Plesk Panel go to your main domain Subscription > Websites & Domains > Web Server Settings. Activate Process PHP by nginx and put following code into both: Additional directives for HTTP and Additional directives for HTTPS:

```
# Enable ActiveSync
<Directory /usr/share/psa-horde>
```

```
 Order allow,deny
 Allow from all
</Directory>
RewriteEngine On
RewriteRule ^/Microsoft-Server-ActiveSync /usr/share/psa-horde/rpc.php
[L,QSA]
Alias /autodiscover/autodiscover.xml /usr/share/psa-horde/rpc.php
Alias /Autodiscover/Autodiscover.xml /usr/share/psa-horde/rpc.php
Alias /AutoDiscover/AutoDiscover.xml /usr/share/psa-horde/rpc.php
# End Enable ActiveSync
```

- Now in Plesk go to Websites & Domains > PHP Settings and set max_execution_time to custom value 3000 (it should be more than the value of $conf['activesync']['ping']['heartbeatmax'] in your horde.conf)

If you do this, Plesk automatically sets proxy_read_timeout 3000; in your nginx.conf, which is important.

- Go to Tools & Settings > Mail Server Settings and set Maximum number of connections for a user per IP address to a higher value, e.g. 40.

I had to raise this value to prevent an error in mail.err: courier-imapd: Maximum connection limit reached for ::1 (Horde makes many local connections to IMAP and it stops to authenticate when the limit is met, your mail client can't login and asks for a password.)

If you use Courier you can alternatively edit the file /etc/courier-imap/imapd:

```
MAXPERIP=40
```

- Reload Apache and Courier IMAP services (Plesk Panel > Tools & Settings > Services Management)
- Optional, but needed for some clients (e.g. Windows 8.1 Mail app): Go to your Webmail and log in with your admin account. Go to Settings (the gear icon) > Administration > Permissions. Add following tree:

All Permissions > Horde > ActiveSync > Provisioning

In Provisioning, set Allow for All Authenticated Users

That's it!
Try to connect with a client (i.e. Outlook 2013). Use these values: username: full email address (eg. contact@mydomain.com)
server: your chosen main domain (e.g. server.com)
domain: the domain of your email address (e.g. mydomain.com) <-- This is necessary for some clients (e.g. Windows 8.1 Mail app).

There are more steps needed to make Autodiscover work, but it is not so important for me.
For more info, here is the official Horde Wiki.

How To Secure Apache with Let's Encrypt on Ubuntu?

Linux

Source:	DataSource
Language:	English
Topic:	Linux
SubTopic:	Apache
Last Edit By:	DochyJP
LastEdit:	2021-04-14
Document type:	User Guide
Status:	Active
Access:	free

Download this page as PDF

Introduction

Let’s Encrypt is a Certificate Authority (CA) that provides an easy way to obtain and install free TLS/SSL certificates, thereby enabling encrypted HTTPS on web servers. It simplifies the process by providing a software client, Certbot, that attempts to automate most (if not all) of the required steps. Currently, the entire process of obtaining and installing a certificate is fully automated on both Apache and Nginx.

We will use Certbot to obtain a free SSL certificate for Apache on Ubuntu 18.04 and set up your certificate to renew automatically.

This tutorial will use a separate Apache virtual host file instead of the default configuration file. We recommend creating new Apache virtual host files for each domain because it helps to avoid common mistakes and maintains the default files as a fallback configuration.

Prerequisites

To follow this tutorial, you will need:

- One Ubuntu 18.04 server set up by following this initial server setup for Ubuntu 18.04 tutorial, including a sudo non-root user and a firewall.
- A fully registered domain name. This tutorial will use your_domain as an example throughout.
- Both of the following DNS records set up for your server.

An A record with your_domain pointing to your server’s public IP address. An A record with www.your_domain pointing to your server’s public IP address. Apache installed. Be sure that you

have a virtual host file for your domain. This tutorial will use /etc/apache2/sites-available/your_domain.conf as an example.

Step 1 — Installing Certbot

The first step to using Let's Encrypt to obtain an SSL certificate is to install the Certbot software on your server.

Certbot is in very active development, so the Certbot packages provided by Ubuntu tend to be outdated. However, the Certbot developers maintain a Ubuntu software repository with up-to-date versions, so we'll use that repository instead.

First, add the repository:

```
sudo add-apt-repository ppa:certbot/certbot
```

You'll need to press ENTER to accept.

Install Certbot's Apache package with apt:

```
sudo apt install python-certbot-apache
```

Certbot is now ready to use, but in order for it to configure SSL for Apache, we need to verify some of Apache's configuration.

Step 2 — Set Up the SSL Certificate

Certbot needs to be able to find the correct virtual host in your Apache configuration for it to automatically configure SSL. Specifically, it does this by looking for a ServerName directive that matches the domain you request a certificate for.

If you followed the virtual host set up step in the Apache installation tutorial, you should have a VirtualHost block for your domain at /etc/apache2/sites-available/your_domain.com.conf with the ServerName directive already set appropriately.

To check, open the virtual host file for your domain using nano or your favorite text editor:

```
sudo nano /etc/apache2/sites-available/your_domain.conf
```

Find the existing ServerName line. It should look like this:

```
...
ServerName your_domain;
...
```

If it does, exit your editor and move on to the next step.

If it doesn't, update it to match. Then save the file, quit your editor, and verify the syntax of your configuration edits:

```
sudo apache2ctl configtest
```

If you get an error, reopen the virtual host file and check for any typos or missing characters. Once

your configuration file's syntax is correct, reload Apache to load the new configuration:

```
sudo systemctl reload apache2
```

Certbot can now find the correct VirtualHost block and update it.

Next, let's update the firewall to allow HTTPS traffic.

Step 3 — Allowing HTTPS Through the Firewall

If you have the ufw firewall enabled, as recommended by the prerequisite guides, you'll need to adjust the settings to allow for HTTPS traffic. Luckily, Apache registers a few profiles with ufw upon installation.

You can see the current setting by typing:

```
sudo ufw status
```

It will probably look like this, meaning that only HTTP traffic is allowed to the web server:

```
Output
Status: active

To                         Action      From
--                         ------      ----
OpenSSH                    ALLOW       Anywhere
Apache                     ALLOW       Anywhere
OpenSSH (v6)               ALLOW       Anywhere (v6)
Apache (v6)                ALLOW       Anywhere (v6)
```

To additionally let in HTTPS traffic, allow the Apache Full profile and delete the redundant Apache profile allowance:

```
sudo ufw allow 'Apache Full'
sudo ufw delete allow 'Apache'
```

Your status should now look like this:

```
sudo ufw status
```

```
Output
Status: active

To                         Action      From
--                         ------      ----
OpenSSH                    ALLOW       Anywhere
Apache Full                ALLOW       Anywhere
OpenSSH (v6)               ALLOW       Anywhere (v6)
Apache Full (v6)           ALLOW       Anywhere (v6)
```

Next, let's run Certbot and fetch our certificates.

Step 4 — Obtaining an SSL Certificate

Certbot provides a variety of ways to obtain SSL certificates through plugins. The Apache plugin will take care of reconfiguring Apache and reloading the config whenever necessary. To use this plugin, type the following:

```
sudo certbot --apache -d your_domain -d www.your_domain
```

This runs certbot with the --apache plugin, using -d to specify the names you'd like the certificate to be valid for.

If this is your first time running certbot, you will be prompted to enter an email address and agree to the terms of service. After doing so, certbot will communicate with the Let's Encrypt server, then run a challenge to verify that you control the domain you're requesting a certificate for.

If that's successful, certbot will ask how you'd like to configure your HTTPS settings:

```
Output
Please choose whether or not to redirect HTTP traffic to HTTPS, removing HTTP
access.
-------------------------------------------------------------------------------
--
1: No redirect - Make no further changes to the webserver configuration.
2: Redirect - Make all requests redirect to secure HTTPS access. Choose this
for
new sites, or if you're confident your site works on HTTPS. You can undo this
change by editing your web server's configuration.
-------------------------------------------------------------------------------
--
Select the appropriate number [1-2] then [enter] (press 'c' to cancel):
```

Select your choice then hit ENTER. The configuration will be updated, and Apache will reload to pick up the new settings. certbot will wrap up with a message telling you the process was successful and where your certificates are stored:

```
Output
IMPORTANT NOTES:
 - Congratulations! Your certificate and chain have been saved at:
   /etc/letsencrypt/live/your_domain/fullchain.pem
   Your key file has been saved at:
   /etc/letsencrypt/live/your_domain/privkey.pem
   Your cert will expire on 2018-07-23. To obtain a new or tweaked
   version of this certificate in the future, simply run certbot again
   with the "certonly" option. To non-interactively renew *all* of
   your certificates, run "certbot renew"
 - Your account credentials have been saved in your Certbot
   configuration directory at /etc/letsencrypt. You should make a
   secure backup of this folder now. This configuration directory will
   also contain certificates and private keys obtained by Certbot so
   making regular backups of this folder is ideal.
 - If you like Certbot, please consider supporting our work by:
```

```
   Donating to ISRG / Let's Encrypt:   https://letsencrypt.org/donate
   Donating to EFF:                    https://eff.org/donate-le
```

Your certificates are downloaded, installed, and loaded. Try reloading your website using https:// and notice your browser's security indicator. It should indicate that the site is properly secured, usually with a green lock icon. If you test your server using the SSL Labs Server Test, it will get an A grade.

Let's finish by testing the renewal process.

Step 5 — Verifying Certbot Auto-Renewal

```
sudo systemctl status certbot.timer
```

```
Output
● certbot.timer - Run certbot twice daily
     Loaded: loaded (/lib/systemd/system/certbot.timer; enabled; vendor
preset: enabled)
     Active: active (waiting) since Tue 2020-04-28 17:57:48 UTC; 17h ago
    Trigger: Wed 2020-04-29 23:50:31 UTC; 12h left
   Triggers: ● certbot.service

Apr 28 17:57:48 fine-turtle systemd[1]: Started Run certbot twice daily.
```

To test the renewal process, you can do a dry run with certbot:

```
sudo certbot renew --dry-run
```

If you see no errors, you're all set. When necessary, Certbot will renew your certificates and reload Apache to pick up the changes. If the automated renewal process ever fails, Let's Encrypt will send a message to the email you specified, warning you when your certificate is about to expire.

How to Check if Your Linux Server is Under DDoS Attack

Linux

Source:	DataSource
Language:	English
Topic:	Linux
SubTopic:	Security
Last Edit By:	DochyJP
LastEdit:	2021-04-14
Document type:	Documentation
Status:	Active
Access:	free

Download this page as PDF

How to Check if Your Linux Server is Under DDoS Attack ?

What is DDoS?

DDoS, or **Distributed Denial of Service**, is a coordinated attack using one or more IP addresses designed to cripple a website by making its server inaccessible. This is done by overloading a server's resources and using up all available connections, bandwidth, and throughput. Just like when driving, your travel-time from point A to point B will be slower if there's too much traffic. By flooding a server with more connections than it can handle, the server becomes bogged down, making it unable to process legitimate requests. Even hardy servers can't handle the number of connections a DDoS can bring.

While there are various means to perform a DDoS attack, ranging from HTTP floods to Slowloris' lingering connections, the vast majority require live connections to your server. Lots of them.

The good news is, because these connections are live, you have the ability to see them as they are being made. Using a few simple commands, you can not only determine if a DDoS is happening, but additionally you can gain the information needed to help mitigate these attacks.

How to Check for DDoS

If you're concerned that your server might be under DDoS attack, the first thing you'll need to do is take a look at the load on your server. Something as simple as the uptime or top commands will give you a good idea of the server's current load.

But what is an acceptable load?

Well, that depends on your CPU resources or available threads. Typically though, the rule is one point per thread.

To determine your server's current load, you can use the grep processor /proc/cpuinfo | wc -l command, which will return the number of logical processors (threads). During a DDoS attack, you may see load at double, triple, or even higher over the maximum load you should have.

To start, use the two commands below to return your uptime and server load :

- Command 1 :

  ```
  grep processor /proc/cpuinfo | wc -l
  ```

- Command 2 :

  ```
  uptime
  ```

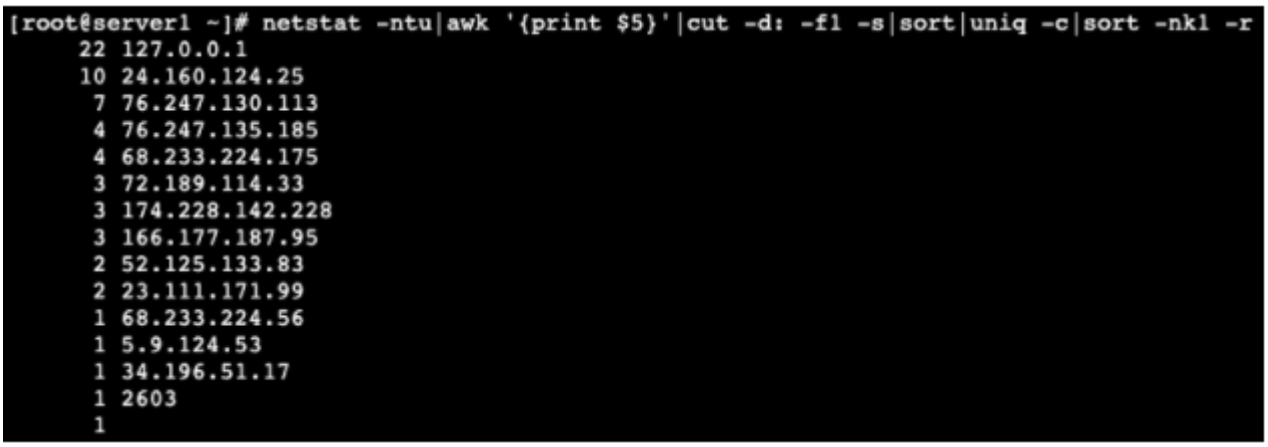
The load average displays load in the following intervals: 1 minute average, 5 minute average, and 15 minute average. In this scenario, a load average of greater than 7 could be a concern.

Unlike the above example, sometimes your server will respond fine over a backend connection like IPMI, but will still be slow when connecting over a public interface. To determine if this is the case, you will want to check your network traffic. This can be done with one of several tools including nload, bmon, iftop, vnstat, and ifstat. Your choice will depend on your operating system, but all of these tools can be installed via your package manager (apt, yum, etc.).

How to Check Which IPs are Connecting to Your Server

Since most DDoS attacks require connections to your server, you can check and see how many, and which, IP addresses are connecting to your server at once. This can be determined using netstat, a command used to provide all manner of details. In this instance though, we're only interested in the specific IPs making connections, the number of IPs, and possibly the subnets they're part of. To start, enter the following command into your terminal:

```
netstat -ntu|awk '{print $5}'|cut -d: -f1 -s|sort|uniq -c|sort -nk1 -r
```



When entered correctly, this command will return a descending list of which IPs are connected to your server and how many connections each one has. The results may also include artifact data, which will appear as non-IP info, and can be ignored.

Looking at your results, you will see connections listed ranging anywhere from 1 to about 50 connections per IP. This can be quite common for normal traffic. If however, you see some IPs with 100+ connections, this is something to scrutinize.

Included in the list, you may see known IPs, one or more of the server's own IPs, or even your own personal IP with many connections. For the most part, these can be ignored, as they are there normally. It's when you see single, unknown IPs with hundreds or thousands of connections that you should be concerned, as this can be a sign of an attack.

Mitigating a DDoS Attack

Once you have an idea of which IPs are attacking your server, blocking these specific IPs can be done with a few simple commands.

To start, use the following command, replacing "ipaddress" with the address of the IP you're trying to block.

```
route add ipaddress reject
```

Once you've blocked a particular IP on the server, you can crosscheck if the IP has been blocked successfully using:

```
route -n |grep ipaddress
```

You can also bock an IP address on the server using iptables by entering the following commands:

```
iptables -A INPUT 1 -s IPADDRESS -j DROP/REJECT
service iptables restart
service iptables save
```

After entering this series of commands, you will need to kill all httpd connections and restart httpd services. This can be done by entering:

```
killall -KILL httpd
service httpd startssl
```

If more than one unknown IP address is making large numbers of connections, either of these processes can be repeated for all offending IPs

DDoS Using Multiple IPs

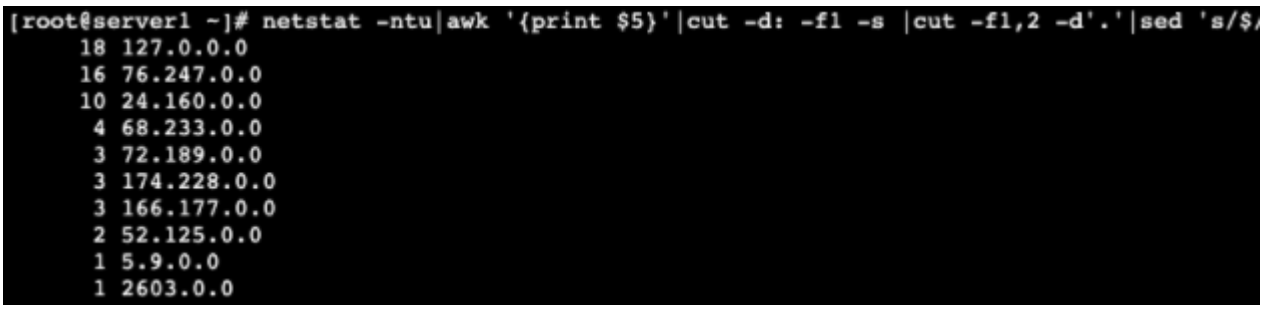
While a denial of service attack from a single IP making numerous connections can be easy to diagnose and fix, DDoS prevention becomes more complex as attackers use fewer connections spread across a larger number of attacking IPs. In these cases, you will see fewer individual connections even when your server is under DDoS. With the rise of the Internet of Things (IoT), these types of attacks have grown more common. By hacking into and utilizing "smart" devices, appliances and tools that feature internet connectivity, malicious actors have built networks of available IPs, referred to as botnets, capable of being deployed in coordinated DDoS attacks against specific targets.

So what can you do if you find large numbers of unknown IPs only making single connections? In this instance, it can be difficult to determine if this is natural traffic or a coordinated attack.

To start, you'll want to determine if these connections are coming from common subnets: common being the same /16 or /24. You can use the next two commands to list the subnets that contain the connected IPs, and how many IPs are in each subnet.

To find IPs from the same /16 (xxx.xxx.0.0) subnet, use:

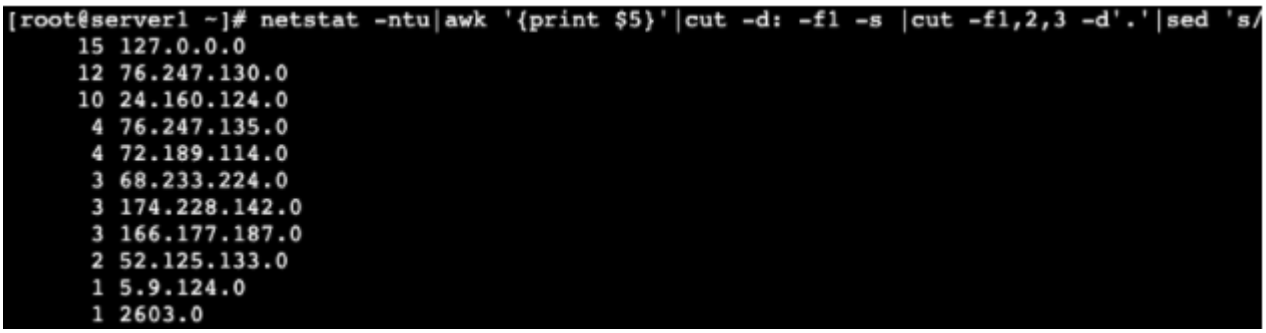
```
netstat -ntu|awk '{print $5}'|cut -d: -f1 -s |cut -f1,2 -d'.'|sed
's/$/.0.0/'|sort|uniq -c|sort -nk1 -r
```



When entered, this command will display any IP starting with the same two octets: ie. 192.168.xxx.xxx.

To find IPs from the same /24 (xxx.xxx.xxx.0) subnet, use:

```
netstat -ntu|awk '{print $5}'|cut -d: -f1 -s |cut -f1,2,3 -d'.'|sed
's/$/.0/'|sort|uniq -c|sort -nk1 -r
```



When entered, this command will display any IP starting with the same three octets: ie. 192.168.1.xxx.

Once you have determined if you are in fact under a multiple-IP DDoS attack, the steps to mitigate it are the same as those used above to combat single IP attacks, but replicated for many IPs.

These techniques are just a few of the tools available to check for possible attacks. While there are more advanced tools available, these methods can provide quick and easy results to determine if you may be experiencing a DDoS attack. The information these commands provide is useful even when not under attack, and familiarizing yourself with them can help strengthen any administrator's tool belt.

Back to top of page - Back to Welcome Page

Install Citrix Workspace on Ubuntu

Linux

Source:	DataSource
Language:	English
Topic:	Linux
SubTopic:	Citrix
Last Edit By:	DochyJP
LastEdit:	2021-04-14
Document type:	Documentation
Status:	Active
Access:	free

Download this page as PDF

The Citrix receiver has been replaced with Citrix workspace app. You can download the new app from one of the citrix website, like https://www.citrix.com/en-au/downloads/workspace-app/linux/workspace-app-for-linux-latest.html

Download the workspace app debian package. For some reason, the software installer can not open the file.

Find the .deb package on your disk :

```
$ ls icaclient_xx.yy.0.zzz_amd64.deb
```

Make sure you search the right version !

In the terminal, go to the folder containing the file and use the dpkg command to install the package:

```
$ sudo dpkg -i icaclient_xx.yy.0.z_amd64.deb
```

That's it, package is installed.

Install/upgrade/Downgrade Composer

Linux

Source:	DataSource
Language:	English
Topic:	Mediawiki
SubTopic:	Linux
Last Edit By:	DochyJP
LastEdit:	2021-04-14
Document type:	Documentation
Status:	Active
Access:	free

Installation of Composer

Use the following commands in your terminal to install Composer :

```
cd ~
curl -sS https://getcomposer.org/installer | sudo php
sudo mv composer.phar /usr/local/bin/composer
```

When moved to /usr/local/bin/composer, just type composer from any folder to run it.

Upgrade

Just type the following command to update Composer

```
composer self-update
```

Downgrade

Composer version 2.1.14 is currently the last version not impacted by the case sensitive package name issue blocking installations and updates of Mediawiki extensions. To downgrade, just use the self-update command with the version required.

```
composer self-update 2.1.14
```

After performing any self-update, with or without version specific, you can specify --rollback to go back to the previously installed version.

```
composer self-update --rollback
```

pre-release version

Finally, if you are feeling adventurous, you can update to a pre-release version by executing:

```
composer self-update --preview
```

Linux commands - Cheat Sheet

Linux

Source:	DataSource
Language:	English
Topic:	Linux
SubTopic:	Linux Commands
Last Edit By:	DochyJP
LastEdit:	2021-04-14
Document type:	User Guide
Status:	Active
Access:	free

Download this page as PDF

Generic Linux Commands

CHMOD

Get the chmod numerical value for a file

```
stat --format '%a' <file/folder name>
```

Get the chmod alphanumerical value for a file

```
stat --format '%A' <file/folder name>
```

Back to top of page - Back to Welcome Page

Linux fail2ban

Linux

Source:	DataSource
Language:	English
Topic:	Fail2Ban
SubTopic:	Linux
Last Edit By:	DochyJP
LastEdit:	2021-04-14
Document type:	Documentation
Status:	Active
Access:	Private

View Banned IP address and jail information

To find out which IP addresses are banned and at what time, you can view the logs from the server where fail2ban is installed:

```
cat /var/log/fail2ban.log
```

The following output shows the IP address “192.168.72.186” is banned by fail2ban and is in jail named “sshd.”

You can also use the following command with the jail name to show banned IPs:

```
sudo fail2ban-client status <jail_name>
```

For example, in our case, the banned IP address is in “sshd” jail, so that the command would be:

```
sudo fail2ban-client status sshd
```

How to find out why users are getting banned by Fail2Ban?

```
grep <myipaddress> /var/log/fail2ban.log
```

To see why it has been banned

To see why it has been banned, search for the IP address in corresponding service logs or use fail2ban-regex utility, for example:

- with service logs:

  ```
  grep <myipaddress>
  /var/www/vhosts/system/example.com/logs/error_log
  ```

- with fail2ban-regex:

  ```
  fail2ban-regex --print-all-matched <service-log>
  /etc/fail2ban/filter.d/<filter-name>.conf<pre>
  ```

The <filter-name> can be found in Tools & Settings > IP Address Banning (Fail2Ban) > Jails > <jail-name> > Settings in line beginning with filter = <filter-name>. For example if you are looking for a website and Apache jail, use the following command: <pre>fail2ban-regex --print-all-matched /var/www/vhosts/system/example.com/logs/error_log /etc/fail2ban/filter.d/apache-auth.conf

Unban an IP in fail2ban

To unban an IP address in fail2ban and remove it from the jail, use the following syntax:

```
sudo fail2ban-client set jail_name unbanip xxx.xxx.xxx.xxx
```

where “jail_name” is the jail where the banned IP address is in and “xxx.xxx.xxx.xxx” is the IP address that is banned.

For example, to unban an IP address “192.168.72.186,” which is in the jail “sshd,” the command would be:

```
sudo fail2ban-client set sshd unbanip 192.168.72.186
```

Verify if the IP address has been unbanned

Now to verify if the IP address has been unbanned, view the logs using the command below:

```
cat /var/log/fail2ban.log
```

In the logs, you will see an Unban entry. Or you can also use the following command to confirm if the IP address has been unbanned:

```
sudo fail2ban-client status <jail_name>
```

Replace “jail_name” with the name of the jail where the banned IP address was in.

If you do not find the IP address listed in the Banned IP list, it means it has been successfully unbanned.

How to whitelist an IP in Fail2ban

Edit the following file : /etc/fail2ban/jail.conf and add now add all IP you want. Each IP or range IP must be separated here with a whitespace. Ex: 192.168.0.1 192.168.5.0/32

```
ignoreip = 192.168.0.1 192.168.5.0/32
```

The line should be added in the [DEFAULT] section of the file. Save the file and restart Fail2Ban:

```
service fail2ban restart
```

Mediawiki backup script

Linux

Source:	DataSource
Language:	English
Topic:	Mediawiki
SubTopic:	Linux
Last Edit By:	DochyJP
LastEdit:	2021-04-14
Document type:	Documentation
Status:	Active
Access:	Micylou restricted

#!/bin/bash # # wikigial fullsitebackup.sh V1.0 # # Full backup of website files and database content. # # - Added mysql password variable (caution - this script file is now a security risk - protect it) # - Generates temp log file # - Updated backup and restore scripts have been tested on Ubunutu Edgy server w/Drupal 5.1 # # # Parameters: # tar_file_name (optional) # ################## # Configuration # ## ## UTM VM Database connection information ## #### dbname="my_wiki" # name of the database dbhost="localhost" # database server dbuser="wikidbadmin" # database user dbpw="Gyzmo6522!" # database user password dbport="3306" # database port webrootdir="/var/www/html/wiki/" ### folder where mediawiki root folder is locater startdir="/var/www/html/backups/" ### backup file location tarnamebase=sitebackup- datestamp=`date +'%Y%m%d'-'%H%M%S'` # Execution directory (script start point) logfile=$startdir"backup.log" # file path and name of log file to use # Temporary Directory tempdir=$datestamp ## ## #### ############### # Start menu ############### clear select delconf in "Backup" "Cancel" do echo "You have selected :" $delconf case $delconf in Backup) echo "Proceeding with Mediawiki full site backup"; break;; Cancel) exit;; esac break done ################## # Website Files ################## cd $webrootdir echo "SELECT THE VERSION TO BACKUP (ctrl-c to quit) :" echo "" select mediawikidir in mediawiki-*.*; do echo "" echo "You have selected $mediawikidir ($REPLY)." break done

########################## # Input Parameter Check # ########################## echo "Input parameters check" if test "$1" = "" then tarname=$tarnamebase$datestamp else tarname=$1 fi echo "Input parameters check" ################## # Begin logging # ################## echo "Beginning backup" echo "Beginning $mediawikidir site backup using FullSiteBackup.sh ..." > $logfile ## # Create temporary working directory # ## echo " Creating temp working temporary directory" echo " Creating temp working dir ..." >> $logfile mkdir $startdir/$tempdir ###################### # ZIP website files # ###################### echo " TARing website files..." echo " TARing website files into $webrootdir ..." >> $logfile cd $webrootdir sudo tar czvf $startdir/$tempdir/filecontent.tar.gz $mediawikidir ################################## # sqldump database information # ################################## echo " Dumping mediawiki database ..." >> $logfile ## echo " Dumping mediawiki database, using ..." >> $logfile ## echo " user:$dbuser; database:$dbname host:$dbhost " >> $logfile cd $startdir/$tempdir mysqldump --host=$dbhost -P$dbport --user=$dbuser --password=$dbpw --add-drop-table $dbname > dbcontent.sql clear ############################## # Create final backup file # # ########################## echo "Create final compressed tgz file..." echo " Creating final compressed (tgz) TAR file: $tarname$mediawikidir" >> $logfile sudo tar -czf $startdir/$tarname-$mediawikidir.tar.gz filecontent.tar.gz dbcontent.sql ########## # Cleanup # ########## echo " Removing temporary directory ..." echo " Removing temp dir $tempdir ..." >> $logfile cd $startdir rm -r $tempdir echo "$temdir directory removed " echo "Backup complete, have a nice day." echo ############## # Exit banner # # ############

Virtualbox Copy/Paste not working

Linux

Source:	DataSource
Language:	English
Topic:	Linux
SubTopic:	VirtualBox
Last Edit By:	DochyJP
LastEdit:	2021-04-14
Document type:	Documentation
Status:	Active
Access:	free

Download this page as PDF

Using VirtualBox under Windows11 as host and installing Ubuntu20 as guest operating system, I realized the copy/paste although set up as BiDiretional between the host and the guest, was not working.

I fixed this using the foloowing commands in the terminal of the guest OS Ubuntu20 :

```
sudo apt update
sudo apt install build-essential linux-headers-generic
sudo apt install build-essential dkms linux-headers-$(uname -r)
```

```
sudo apt install virtualbox-guest-x11
sudo VBoxClient --clipboard
```

Enjoy !

WIKI - KMS - Crontabs

Linux

Source:	DataSource
Language:	English
Topic:	Mediawiki
SubTopic:	Linux
Last Edit By:	DochyJP
LastEdit:	2021-04-14
Document type:	Documentation
Status:	Active
Access:	free

Download this page as PDF

== Introduction == The following scripts need to be scheduled on the server to maintain data consistencies :

- RunJobs.sh
- BuildSemanticData.sh
- Indexation.sh

These are PHP scripts but to ease the work of non-mediawiki addict admins, they are summarizede in these sh scripts.

RunJobs.sh

Even if a service is running in the background, it is necessary to run periodically because it is required for Semantic Data to be correct.

Code script RunJobs.sh

```
cd
clear
cd /usr/local/apache2/htdocs/mediawiki_latest/maintenance/
php runJobs.php
cd
```

cd clear cd /usr/local/apache2/htdocs/mediawiki_latest/maintenance/ php runJobs.php cd

Indexation.sh

Indexation.sh needs to be run in order to rebuild Mediawiki's indexes. Because the database is on an instable cluster, they often get corrupted (around once or twice a week in general).

Script Indexation.sh

```
cd
clear
cd /usr/local/apache2/htdocs/mediawiki_latest/maintenance/
php rebuildall.php
cd
```

BuildSemanticData.sh

BuildSemanticData.sh suffers not only from instable connections to the database cluster, but is also dependent on the cache of the browser of each user. Therefore, datas may look incomplete or different according who is watching what. To resolve this, the BuildSemanticData.sh scrip^t need to be ran regularly (2 or 3 times a day right after the RunJobs.sh script.

Code script BuildSemanticData.sh

```
cd
clear
cd
/usr/local/apache2/htdocs/mediawiki_latest/extensions/SemanticMediaWiki/maint
enance/
php rebuildData.php
cd
```

Crontab script

One script, ScriptingBulk.sh, is to be scheduled in root on each server (DEV, ACC, PRD) to run daily at It starts each of the 3 preceeding script in the following oprder :

1. RunJobs.sh
2. Indexation.sh
3. BuildSemantiocData.sh

Code script ScriptingBulk.sh

```
cd
clear
cd /usr/local/apache2/htdocs/scripting
sudo ./RunJobs.sh
sudo ./Indexation.sh
sudo ./BuildSemanticData.sh
cd
```

Scheduling the scripts to run

The contab created as root (sudo su -)is the following :

```
30 5,12,17 * * * /usr/local/apache2/htdocs/scripting/ScriptingBulk.sh
```

This means that the Mediawiki php scripts will run daily at 05:30, 12:30 and 17:30.

cron is a Unix daemon that allows users to schedule programs to run at set intervals. A **crontab** is the file that cron uses to determine what actions to perform and when. A cronjob is a the specific line (or job) in the crontab. The Toolserver provides a version of cron called **cronie**, which is more featureful than Solaris' built-in cron, and will be described here. For information on Solaris cron, refer to the **crontab(1)** manual page.

On the Toolserver, we **strongly** recommend that cron be used in conjunction with job scheduling. This is how to do this.

Basic commands

crontab has three basic commands:

`$ crontab -e`
: edits (or creates) your current crontab in a text editor

`$ crontab -l`
: displays your crontab

`$ crontab -r`
: removes your crontab entirely

You can change the text editor used for editing the crontab in your environment.

crontab syntax

Each line in the crontab file is a single cron job. The line contains five fields at the start indicating when the job should run, and the rest of the line is the command.

```
minute (0-59),
|       hour (0-23),
|       |       day of the month (1-31),
|       |       |       month of the year (1-12),
|       |       |       |       day of the week (0-6 with 0=Sunday).
|       |       |       |       |       commands
30      2       *       *       0,6     /some/command/to/run
30      2       *       *       1-5     /another/command/to/run >/dev/null
```

Use "*" to mean "every"; for example, "30 2 * * *" runs the program at 02:30 UTC, every day of every month. Only specify "day of month" *or* "day of week", not both.

On Linux, you can write "*/**x**" to mean "every **x**". For example, writing "*/10" in the "minute" column means "every ten minutes".

The crontab file must end with a blank line (that is, the last character of the final line in the file must be a newline character). Many editors, like **nano** (the default editor) and **vi** do this by default. The "**joe**" editor does *not* do this - be sure to end your file with a blank line every time. If you don't, you won't receive an error message, but nothing will happen at the scheduled time.

An example which would run every hour of the day would be:

```
0 * * * * php $HOME/scripts/myscript.php
```

This means every hour, i.e. *, at minute 0. There is no need to write */1 for the hour field, as */1 is identical to *.

Lines beginning with a # will be treated as comments, and ignored, along with blank lines.

[Back to top of this page]

[Back to Welcome Page]