

Guacamole - HTML5 Remote Desktop Server

Das [Guacamole Projekt](#) ist eine **HTML5 Remote-Access Applikation**, die als zentrale Verwaltungskonsole entfernter Computer verwendet werden kann. Unterstützt werden derzeit die Protokolle **VNC, RDP, SSH** wie auch **Telnet**.

Mit der Version 0.9.10 ist die Verwendung von **WebSockets** nun auch ein fester Bestandteil, was besonders bei grafisch aufwendigen Verbindungen wie einer RDP-Session bemerkbar wird.



Alternativ: [Installation von Glyptodon Enterprise \(Webpage\)](#)

Installation von Guacamole 1.0.0

- [Installation auf Red Hat OS](#)
- [Installation auf Debian OS](#)

Die folgende Installationsanleitung wurde unter **Redhat 7.5, Ubuntu 16.04 sowie Debian 9 getestet**. **ACHTUNG:** Wird Guacamole noch auf einem älteren Linux z.B. einem Redhat 6.x / Debian 8 installiert, so muss als Webserver Tomcat7 anstelle von Tomcat8 verwendet werden.

Vorbereitungen für Guacamole

Um überhaupt einen funktionierenden Betrieb von Guacamole zu gewährleisten, werden zu Beginn erst einmal alle *von Guacamole gebrauchten Päckli* installiert:

```
# yum -y install epel-release wget
# wget -O /etc/yum.repos.d/home:felfert.repo
http://download.opensuse.org/repositories/home:/felfert/Fedora_25/home:felfert.repo

# yum -y groupinstall "Development Tools"
# yum -y install cairo-devel freerdp-devel freerdp-plugins git java-1.8.0-openjdk.x86_64 libguac libguac-client-rdp libguac-client-ssh libguac-client-vnc libjpeg-turbo-devel libpng-devel libssh2-devel libtelnet-devel libvncserver-devel libwebp-devel libvorbis-devel mariadb-server maven openssl-devel pango-devel pulseaudio-libs-devel terminus-fonts tomcat tomcat-admin-webapps tomcat-webapps uuid-devel

# rpm -Uvh
http://li.nux.ro/download/nux/dextop/el7/x86_64/nux-dextop-release-0-1.el7.nux.noarch.rpm
```

```
# yum -y install ffmpeg-devel
```

```
# apt-get install -y libjpeg-dev libcairo2-dev libossp-uuid-dev libpng-dev  
libfreerdp-dev libssh2-1-dev libssh-dev libwebp-dev libpulse-dev libavcodec-  
dev libavutil-dev libswscale-dev libpango1.0-dev libvncserver-dev maven  
tomcat8 tomcat8-admin tomcat8-user default-jdk default-jre java-common  
mariadb-server libtool dh-autoreconf git libvorbis-dev
```

Direkt nach der Installation der Päckli, sichern wird als erstes die lokale MySQL Datenbank ab.

Generieren eines neuen MaiaDB-Root Passwortes:

```
# openssl rand -base64 30 > /root/.mariadb-root-pw && cat /root/.mariadb-  
root-pw
```

```
Tb/qprITSryJDHEp29Xhr7/IuxMxZhGke/LZXEEJ
```

```
# systemctl start mariadb.service  
# systemctl enable mariadb.service
```

```
# mysql_secure_installation
```

```
Enter current password for root (enter for none): Enter  
Set root password? [Y/n]: Y  
New password: *****  
Re-enter new password: *****  
Remove anonymous users? [Y/n]: Y  
Disallow root login remotely? [Y/n]: Y  
Remove test database and access to it? [Y/n]: Y  
Reload privilege tables now? [Y/n]: Y
```

```
All done!
```

Zum Schluss der Vorbereitungen werden noch die drei Guacamole Kern-Komponenten sowie den MySQL Connector zum herstellen der Guacamole Datenbankverbindung heruntergeladen.

```
# mkdir ~/build && cd ~/build  
  
# wget https://www.blackgate.org/guac/guacamole-1.0.0.war  
# wget https://www.blackgate.org/guac/guacamole-auth-jdbc-1.0.0.tar.gz  
# wget https://www.blackgate.org/guac/guacamole-server-1.0.0.tar.gz  
# wget  
https://dev.mysql.com/get/Downloads/Connector-J/mysql-connector-java-5.1.45.
```

```
tar.gz
```

Kompilieren des Guacd - Servers

Als erstes wir das tar Archiv, indem sich der **Server Sourcecode** befindet lokal entpackt und anschliessend **kompiliert** und **installiert**. Darauf folgend erstellen wir noch zwei wichtige Verzeichnisse die von unserem Server später gebraucht werden.

```
# tar -xvf guacamole-server-1.0.0.tar.gz
# cd guacamole-server-1.0.0/

# ./configure --with-init-dir=/etc/init.d
# make
# make install
# cd ..

# mkdir -p /etc/guacamole/extensions
# mkdir /etc/guacamole/lib
```

WICHTIG: RDP Workaround aufgrund von Kompatibilitätsproblemen unter Debian / Ubuntu:

```
# mkdir /usr/lib/x86_64-linux-gnu/freerdp
# ln -s /usr/local/lib/freerdp/guac* /usr/lib/x86_64-linux-gnu/freerdp/
# ldconfig
```

Einrichten der Guacamole Datenbank

In diesem Schritt wird mit dem **DB-root** Benutzer auf den **MySQL Server** verbunden und die Datenbank **Guacamole** erstellt und abgefüllt. Weiter wird aus Security Gründen einen eigenen Benutzer dazu erstellt, welcher ausschliesslich auf die Guacamole DB berechtigt wird. So wird verhindert, dass dieser Benutzer Veränderungen an der Server eigenen DB Struktur vornehmen kann.

```
# mysql -u root --password=$(cat /root/.mariadb-root-pw)

CREATE DATABASE guacamole;
CREATE USER 'guacamole'@'localhost' IDENTIFIED BY
'v2gjMrY/2XmgJwhNE56scymqTiC337XkVKOHtYw9';
GRANT SELECT,INSERT,UPDATE,DELETE ON guacamole.* TO 'guacamole'@'localhost';
FLUSH PRIVILEGES;
quit
```

Hier wird das Guacamole **DB Authentifizierungstool jdbc** entpackt, in die Installation integriert und anschliessend die Datenbank abgefüllt. Zum Schluss wird dann noch der offizielle MySQL Connector driver ebenfalls mit einbezogen.

```
# tar -xvf guacamole-auth-jdbc-0.9.14.tar.gz
```

```
# cd guacamole-auth-jdbc-0.9.14/mysql/

# cp guacamole-auth-jdbc-mysql-0.9.14.jar /etc/guacamole/extensions/
# cat schema/*.sql | mysql -u root -p guacamole
# cd ../..

# tar -xvf mysql-connector-java-5.1.40.tar.gz
# cd mysql-connector-java-5.1.40
# cp mysql-connector-java-5.1.40-bin.jar /etc/guacamole/lib/

# cd ..
```

Einrichten des Web-Clients

Nun kann der Web-Client (*das User Interface*) von Guacamole auf dem System eingerichtet werden. Hierzu wird das selbst-entpackende *.war File welches den Client beinhaltet ins Webverzeichnis von **Tomcat8** verlinkt. Zu guter letzt wird dann noch das Guacamole HOME als Tomcat-Umgebungsvariable festgelegt.

```
# cp guacamole-0.9.14.war /etc/guacamole/guacamole.war
# ln -s /etc/guacamole/guacamole.war /var/lib/tomcat/webapps/
# mkdir /usr/share/tomcat/.guacamole
# touch /etc/guacamole/guacamole.properties

# ln -s /etc/guacamole/guacamole.properties /usr/share/tomcat/.guacamole/
# echo GUACAMOLE_HOME="/etc/guacamole" >> /usr/share/tomcat/conf/tomcat.conf
```

Befüllen der Haupt-Konfigurationsdatei von Guacamole. Alle Änderungen die hier hineingeschrieben werden, überschreiben lediglich den default Wert von Guacamole. (*Würde also auch wenn keine DB Authentifizierung verwendet würde ohne etwas zu ergänzen funktionieren.*)

```
# vim /etc/guacamole/guacamole.properties
```

```
# MySQL properties
mysql-hostname: localhost
mysql-port: 3306
mysql-database: guacamole
mysql-username: guacamole
mysql-password: v2gjMrY/2XmgJwhNE56scymqTiC337XkVK0HtYw9
```

Zum Schluss, wird noch den Autostart der Services aktiviert und das System anschliessend neugestartet!

```
# systemctl enable tomcat.service && chkconfig guacd on
# firewall-cmd --zone=public --add-port=8080/tcp --permanent
# firewall-cmd --reload
```

```
# systemctl reboot
```

Der Server sollte nun unter folgendem Link erreichbar sein: http://your_IP:8080/guacamole

Für weitere Details, Hilfestellungen: [Offizielles Guacamole Manual](#)

ACHTUNG: Das Standard Passwort von Guacamole muss unbedingt noch geändert werden!

- **Username:** guacadmin
- **Passwort:** guacadmin

Weiteres

Behebung von SELinux Problemen

Folgende SELinux Boolean aktivieren:

```
# setsebool -P tomcat_can_network_connect_db on
```

Depricated

Es kann vorkommen, dass mit den aktuellen SELinux-Core-Rules Probleme mit dem Zusammenspiel von Guacamole und Java auftreten können. Wie man diese einfach behebt, erläutere ich hier in den nachfolgenden Schritten:

- Installation von den Troubleshoot Packages (*Falls nicht schon vorhanden*):

```
# yum install setroubleshoot setools
```

- Um nun die Fehler aus unserem Audit.log automatisiert auszuwerten folgenden Befehl ausführen:

```
# sealert -a /var/log/audit/audit.log
```

Jeder hier generierte Report, beschreibt zuerst den Fehler, und erklärt danach möglichst genau, wie das Problem behoben werden kann. Ausgabe des Befehls:

```
[root@admin-server ~]# sealert -a /var/log/audit/audit.log
100% done
found 1 alerts in /var/log/audit/audit.log
-----
-----

SELinux is preventing /usr/lib/jvm/java-1.8.0-
openjdk-1.8.0.141-1.b16.el7_3.x86_64/jre/bin/java (deleted) from
```

name_connect access on the tcp_socket port 3306.

***** Plugin catchall (100. confidence) suggests

If you believe that java (deleted) should be allowed name_connect access on the port 3306 tcp_socket by default.

Then you should report this as a bug.

You can generate a local policy module to allow this access.

Do

allow this access for now by executing:

```
# ausearch -c 'java' --raw | audit2allow -M my-java
# semodule -i my-java.pp
```

Additional Information:

Source Context	system_u:system_r:tomcat_t:s0
Target Context	system_u:object_r:mysql_d_port_t:s0
Target Objects	port 3306 [tcp_socket]
Source	java
Source Path	/usr/lib/jvm/java-1.8.0-
openjdk-1.8.0.141-1.b16.el	7_3.x86_64/jre/bin/java (deleted)
Port	3306
Host	<Unknown>
Source RPM Packages	java-1.8.0-openjdk-
	headless-1.8.0.144-0.b01.el7_4.x86_64
Target RPM Packages	
Policy RPM	selinux-policy-3.13.1-166.el7_4.4.noarch
Selinux Enabled	True
Policy Type	targeted
Enforcing Mode	Permissive
Host Name	admin-server.blacknet
Platform	Linux admin-server.blacknet
	3.10.0-693.2.2.el7.x86_64 #1 SMP Tue Sep
12	22:26:13 UTC 2017 x86_64 x86_64
Alert Count	16
First Seen	2017-09-17 13:33:21 CEST
Last Seen	2017-09-18 08:55:14 CEST
Local ID	42523e63-a9ef-438e-8a07-7e8d128d669b

Raw Audit Messages

```
type=AVC msg=audit(1505717714.165:1003): avc: denied { name_connect }
for pid=1217 comm="java" dest=3306
scontext=system_u:system_r:tomcat_t:s0
tcontext=system_u:object_r:mysql_d_port_t:s0 tclass=tcp_socket
```

```
type=SYSCALL msg=audit(1505717714.165:1003): arch=x86_64
syscall=connect success=yes exit=0 a0=77 a1=7f4cb79f6380 a2=1c a3=504
```

```
items=0 ppid=1 pid=1217 auid=4294967295 uid=91 gid=91 euid=91 suid=91
fsuid=91 egid=91 sgid=91 fsgid=91 tty=(none) ses=4294967295 comm=java
exe=/usr/lib/jvm/java-1.8.0-
openjdk-1.8.0.144-0.b01.el7_4.x86_64/jre/bin/java
subj=system_u:system_r:tomcat_t:s0 key=(null)

Hash: java,tomcat_t,mysqld_port_t,tcp_socket,name_connect
```

- Wie oben ersichtlich, wird zu unserem Problem eine **Lösung durch Eingabe von folgenden zwei Befehlen** empfohlen:

```
# ausearch -c 'java' --raw | audit2allow -M guacamole-java
# semodule -i guacamole-java.pp
```

- **Der erste Befehl, erstellt im aktuellen Verzeichnis eine neue SELinux Regel im *.te Format (Text) und kompiliert sie** anschliessend in ein *.pp Format. (Der in der Ausgabe verwendete Namen "my-java" kann beliebig festgelegt werden! z.B. wie bei mir: guacamole-java)
- **Mit dem zweiten Befehl, wird die Regel dann permanent aktiviert!**

Interessante Links

- Wie wird die [Two Level Authentication](#) aktiviert?
- Wie wird eine [LDAP authentication](#) ermöglicht?
- Grosses [Troubleshooting mit Guacamole](#)

Upgrade auf neue Version von Guacamole

Um auf die neue Version von Guacamole zu upgraden, kann eigentlich wie oben bei der Neuinstallation vorgegangen werden. Es wird jedoch empfohlen, das Upgrade wie unten in den einzelnen Schritten beschrieben durchzuführen; Um auch alle gespeicherten Verbindungen und User zu erhalten, braucht es nämlich trotzdem die einte oder andere kleine Abänderung.



WICHTIG: Ein grosser unterschied besteht darin, dass der gesamte DB Create Part übersprungen wird und anstelle von diesem Abschnittes hier die Schritte aus dem Punkt [MySQL-DB Upgrade](#) durchgeführt werden.

Vorbereitungen für das Upgrade

Vor dem Beginn werden erstmals die zwei **Haupt-Services** die Guacamole auszeichnen **gestoppt**.

```
# systemctl stop tomcat.service
# systemctl stop guacd
```

Ist dies erledigt, kann man mit dem Upgrade eigentlich sofort beginnen.. Will man noch seine DB-Credentials nachschauen, kann man dies im guacamole.properties tun.

```
# cat /etc/guacamole/guacamole.properties
```

```
mysql-hostname: localhost
mysql-port: 3306
mysql-database: xxxxxxxxxx
mysql-username: xxxxxxxxxx
mysql-password: xxxxxxxxxx
```

Als erstes, werden die *alten Dateien / Verzeichnisse gelöscht*, welche nicht mehr von der neuen Guacamole Version unterstützt, gebraucht werden:

```
# rm -rf /etc/guacamole/extensions/*
# rm -f /var/lib/tomcat/webapps/guacamole.war
# rm -rf /var/lib/tomcat/webapps/guacamole
# rm -f /etc/guacamole/guacamole.war
```

Zum durchführen des Upgrades, wird nun weiter folgendermassen vorgegangen:

1. **Herunterladen der neuen Versionen (Server, WebClient, jdbc und falls gebraucht duo)**

```
# cd ~/build

# wget https://www.blackgate.org/guac/guacamole-server-0.9.1x.tar.gz
# wget https://www.blackgate.org/guac/guacamole-auth-jdbc-0.9.1x.tar.gz

##DOWNLOAD JUST ONE VERSION, HARDENED BY MICHAEL OR ORIGINAL VERSION:##
# wget https://www.blackgate.org/guac/guacamole-0.9.1x.war
# wget https://www.blackgate.org/guac/guacamole-0.9.1x_hardened.war

# wget https://www.blackgate.org/guac/guacamole-auth-duo-0.9.1x.tar.gz
```

2. **Entpacken** des Server Sourcecodes und der Extentions

```
# tar -xvf guacamole-server-0.9.1x.tar.gz
# tar -xvf guacamole-auth-jdbc-0.9.1x.tar.gz

# tar -xvf guacamole-auth-duo-0.9.1x.tar.gz
```

3. Server **kompilieren** und **installieren**.

```
# cd guacamole-server-0.9.1x/
```



```
# ./configure --with-init-dir=/etc/init.d
# make
# make install

# cd ..
```

4. Neue Versionen der Erweiterungen, aus den Source Ordnern in /ect/guacamole/extension kopieren! (z.B. auth-jdbc & auth-duo)

```
# cd guacamole-auth-jdbc-0.9.1x/mysql/

# cp guacamole-auth-jdbc-mysql-0.9.1x.jar /etc/guacamole/extensions/
# cd ../..

# cp guacamole-auth-duo-0.9.1x/guacamole-auth-duo-0.9.1x.jar
/etc/guacamole/extensions/
```

5. **Neuer WebClient** nach /etc/guacamole/ kopieren und Symlinks erneuern. **ACHTUNG: Nur eine Version kopieren!**

```
##COPY JUST ONE VERSION, SAME YOU DESIDED ABOVE!: (HARDENED BY MICHAEL
OR ORIGINAL VERSION)##

# cp guacamole-0.9.1x_hardened.war /etc/guacamole/guacamole.war
# cp guacamole-0.9.1x.war /etc/guacamole/guacamole.war

# ln -s /etc/guacamole/guacamole.war /var/lib/tomcat/webapps/
```

6. Falls nötig, **DB-Upgrade durchführen**. Ansonsten Dienste (wie ganz unten beschrieben) wieder starten.

MySQL-DB Upgrade

Um nun (falls Nötig → *wenn Files vorhanden*) das MySQL Upgrade durchzuführen, muss wie folgt vorgehen werden:

```
# cd guacamole-auth-jdbc-0.9.1x/mysql/schema/upgrade

# cat upgrade-pre-0.9.1x.sql | mysql -u root --password=$(cat
/root/.mariadb-root-pw) guacamole
```

ACHTUNG: Wenn zwei db Aktualisierungen notwendig sind; z.B. beim Upgrade von Guacamole 0.9.13 auf 0.9.15, muss die DB IMMER zuerst auf die erste vorherige Version in dem Fall 0.9.14 upgedated werden, noch bevor, schlussendlich auf die neuste Release Version 0.9.15 aktualisiert werden kann!

Nach dem erfolgreichen Upgrade, kann der Guacamole Server (nach einem daemon-reload) sowie der Tomcat Webserver anschliessend wieder gestartet werden!

```
# systemctl daemon-reload
```

```
# systemctl start guacd  
# systemctl start tomcat.service
```

Last update: **2019/04/24 09:03**