Installation of TAK Server 5.1 on Ubuntu 22 - Single Server Option

B. Graham

Updated 2 July 2024

Introduction

The official TAK server installation instructions are long (cover multiple installation options), meandering (not presented in sequence so user must jump back and forth between sections) and incomplete (they do not include every step required to install TAK Server especially for users who are not used to Linux).

These instructions represent every step I followed in order to install TAK Server 5.1 on Ubuntu Desktop 22.04.2 under the single server option. The Single server installation is recommended for fewer than 500 users. I enabled SSH on the Ubuntu machine then performed all of the CLI work via a Windows PowerShell SSH session.

About TAK Server

TAK Server is a situational awareness server, that provides a dynamic Common Operating Picture to users of the Team Awareness Kit, including ATAK (Android), WinTAK (Windows) and WebTAK. TAK enables sharing of geolocated information in real time for military forces, law enforcement, and emergency responders. It supports both wireless and wired networks, as well as cloud and data center deployment

ATAK and WinTAK have a civil and military (CAN) versions. However, there is only one version of TAK Server.

Overview

These are the high level steps required to get TAK server working.

1. Install Ubuntu
2. Do steps not in the TAK Server Install guide that prep the VM (VM tools, SSH, curl).
3. Do Steps from Guide to prep Ubuntu (TCP and repo changes)
4. Do Steps from Guide to install TAK server and start it
5. Generate Certificates (described in Appendix B)
6. Configure TAK Server to use the certificates.
7. Authorize the `admin` cert to perform administrative functions using the UI
8. Do steps not in the guide to extract the certificates from the TAK Server so they can be used by clients (ATAK, WinTAK, WebTak, TAKx).
9. Verify that clients can connect to TAK Server.

---

1. Install Ubuntu 22

Install Ubuntu on bare metal or a VM. These instructions were developed with Ubuntu Desktop 22.04.4 (ubuntu-22.04.4-desktop-amd64.iso) running as VM on VMWare Workstation V17.5.0 build-22583795.

The VM was provisioned with the TAK Server recommended minimum requirements of

- 4 processor cores

- 8 GB RAM
- 40 GB disk storage

During the installation of Ubuntu you typically create a user account. I created a user account for `tech`

Verify the Ubuntu install with `hostnamectl` as follows:

```
tech@tech-virtual-machine:~$ hostnamectl

 Static hostname: TAKServer5
       Icon name: computer-vm
         Chassis: vm
      Machine ID: c928d69b5f034eb48ac1b5e0a2442997
         Boot ID: ba00a525d5f84770af75a0ed669a436e
  Virtualization: vmware
Operating System: Ubuntu 22.04.4 LTS
          Kernel: Linux 6.5.0-41-generic
    Architecture: x86-64
 Hardware Vendor: VMware, Inc.
  Hardware Model: VMware Virtual Platform
```

Copy the TAK installer file `takserver_5.1-RELEASE11_all.deb` to the Ubuntu machine. I used the built in Firefox browser to get the file from a NAS and downloaded it to Downloads folder: `\home\tech\Downloads`

The guide provides steps to verify the GPG signature of the installer (*.deb) file but this is optional. I did not perform these steps and they are not included in this guide.

---

2. Steps not in the TAK Server Install guide that prep the VM (VM tools, SSH, curl)

These steps were necessary (or useful) but are not in the official installation guide.

1. Install VMWare tools (if installing on a VM):

```
sudo apt install open-vm-tools-desktop open-vm-tools
```

2. Optional: install openssh (allows doing the install from PowerShell on a Windows machine):

```
sudo apt update
sudo apt upgrade
sudo apt install openssh-server
```

3. Install curl (this is not covered in the guide but is required):

```
sudo apt install curl
```

---

3. Steps from Guide to prep Ubuntu (TCP and repo changes)

4. Increase system limit for number of concurrent TCP connections (do once).

```
echo -e "* soft nofile 32768\n* hard nofile 32768" | sudo tee --append
/etc/security/limits.conf > /dev/null
```

This command adds two lines

> ```
> * soft nofile 32768
> * hard nofile 32768
> ```

to the file `/etc/security/limits/conf`

You can verify that the above command with:

```
tail /etc/security/limits.conf
```

which should look like this:

5. Install the postgres repository (required in order to install up-to-date Postgresql and PostGIS packages.)

```
sudo mkdir -p /etc/apt/keyrings

sudo curl https://www.postgresql.org/media/keys/ACCC4CF8.asc --output
/etc/apt/keyrings/postgresql.asc

sudo sh -c 'echo "deb [signed-by=/etc/apt/keyrings/postgresql.asc]
http://apt.postgresql.org/pub/repos/apt/ $(lsb_release -cs)-pgdg main" >
/etc/apt/sources.list.d/postgresql.list'

sudo apt update
```

---

4 Install TAK server and start it

5. Install TAK server.

Note: some versions (4.8, 4.9, 4.10) of the installation guide have a typo for the .deb file: sometimes it has two hyphens instead of an underscore and a hyphen.

Note: The TAK Server installation media has four.deb files. Use the one below, the others are for multi-server or federation hub installations. (I accidentally used `takserver-core_5.1-RELEASE11_all.deb` the first time which did not workout well.)

```
sudo apt install ./takserver_5.1-RELEASE11_all.deb
```

6. Start TakServer and also configure it to start at system boot.

```
sudo systemctl daemon-reload
sudo systemctl start takserver
sudo systemctl enable takserver
```

Note that `sudo systemctl start takserver` takes around 30 seconds to complete.

---

5. Generate Certificates (described in Appendix B)

TAK Server by default is TLS only, so certificate generation, including an administrative certificate is required for configuration.

TAK Server includes scripts for generating a private security enclave, which will create a Certificate Authority (CA) as well as server and client certificates.

7. Become `tak` user:

```
sudo su tak
```

8. Move to certs folder:

```
cd /opt/tak/certs
```

9. Use nano (or vi) to edit the certificate-generation configuration file, at this location: `/opt/tak/certs/cert-metadata.sh` and set options for country, state, city, organization, and organizational_unit. Leave the other items as is. Note that `atakatak` in the line `CAPASS=${CAPASS:-atakatak}` is a password that will be used for each certificate that is created in the next section. You can change this if necessary but I did not for the purposes of this guide.

```
nano /opt/tak/certs/cert-metadata.sh
```

My example (seems to need the " " around everything except country):

```
# Common configuration for all certificates
#  Edit these fields to be appropriate for your organization
#  If they are left blank, they will not be included.  Do not leave COUNTRY
#  blank (you may set it to "XX" if you want to be obtuse).
#
#  Values for each may be optionally set as environment variables.
#  Replace variables such as ${STATE} and ${CITY} as needed.
#

COUNTRY=CA
STATE="ON"
CITY="OTTAWA"
ORGANIZATION="DLCSPM"
ORGANIZATIONAL_UNIT="SIL"

CAPASS=${CAPASS:-atakatak}
PASS=${PASS:-$CAPASS}

## subdirectory to put all the actual certs and keys in
```

```
DIR=files
```

10. Create a Root certificate authority (CA) note: must be in `/opt/tak/certs` folder:

```
./makeRootCa.sh
```

Follow the prompt to name the CA (or take the default). I took the default. This command will result in a server certificate named `truststore-root.p12` (**TBC**)

11. Create a server certificate. This is the issuing certificate authority:

```
./makeCert.sh server takserver
```

this command will result in a server certificate named `/opt/tak/certs/files/takserver.jks`

12. Create admin and user certificates:

```
./makeCert.sh client admin
```

```
 ./makeCert.sh client user
```

These commands will result in certificates named `admin.p12` and `user.p12`

You can create many client certificates - the official guide recommends creating a unique certificate for every user. We won't do that here.

The generated CA truststores and certificates will be located at: `/opt/tak/certs/files`

---

6. Configure TAK Server to use the TAK Server Certificate

13. verify contents of `CoreConfig.xml`

Note: This is a verification step. There should be no need to change anything on this file.

Note: Always use the `tak` user account when editing `CoreConfig.xml` or generating certificates.
In `/opt/tak`, **check** the following settings in `CoreConfig.xml` (use vi or nano - nano in Powershell is difficult to read so I used vi)

```
vi ./CoreConfig.xml
```

1. In the `<tls>` element, the `keystoreFile` attribute should be set to the server keystore that was generated with makeCerts.sh. The official TAK server installation guide states that the path for `takserver.jks` entry should be `/opt/tak/certs/files/takserver.jks` However the actual line in the file as installed by this guide is: `keystoreFile="certs/files/takserver.jks" keystorePass="atakatak"` The difference is that the official guide states full path but example file does not:

```
57      <security>
58          <tls keystore="JKS" keystoreFile="certs/files/takserver.jks" keystorePass="atakatak" truststore="JKS"
            truststoreFile="certs/files/truststore-root.jks" truststorePass="atakatak" context="TLSv1.2" keymanager="SunX509"/>
59      </security>
```

2. In the `<tls>` element, the `truststoreFile` attribute should be set to the the trust store that was generated with makeCerts.sh. The official TAK server installation guide states that the path for `truststore-root.jks` is `/opt/tak/certs/files/truststore-root.jks` However the actual line in the file as installed by this guide is : `truststoreFile="certs/files/truststore-root.jks" truststorePass="atakatak"` Again the difference is in the path - absolute or relative.
3. In the element, verify the `TLS` input, specifying group-based filtering without requiring an authentication message:

```
input _name="stdssl" protocol="tls" port="8089" auth="x509"
```

Note in the real file there was a similar line but `auth="x509"` was not present while `coreVersion="2"` was present. I did not change this so the file `/opt/tak/CoreConfig.xml` retains `coreVersion="2"`. The complete `<network>` element of CoreCOnfig.xml is as follows:

```
<network multicastTTL="5" version="5.1-RELEASE-11-HEAD">
    <input _name="stdssl" protocol="tls" port="8089" coreVersion="2"/>
    <connector port="8443" _name="https"/>
    <connector port="8444" useFederationTruststore="true" _name="fed_https"/>
    <connector port="8446" clientAuth="false" _name="cert_https"/>
    <announce/>
</network>
```

Note - TBD: Need to better understand the difference between the guide (`auth="x509"`) and the as-is file (`coreVersion="2"` ").

14. Stop being user `tak`

```
exit
```

15. Restart the TAK Server:

```
sudo systemctl restart takserver
```

---

7. Authorize the admin cert to perform administrative functions using the UI

The guide says to execute a java command (`sudo java -jar /opt/tak/utils/UserManager.jar certmod -A /opt/tak/certs/files/admin.pem`) but this always fails by throwing an exception. The exception is due to a file permission problem which can be solved as follows:

16. Add the current user (`tech` in this example) to the tak group: `sudo usermod -aG tak <username>` where `<username>` is `tech` . Specifically:

`sudo usermod -aG tak tech`

17. Authorize the admin cert to perform administrative functions using the UI:

```
sudo java -jar /opt/tak/utils/UserManager.jar certmod -A /opt/tak/certs/files/admin.pem
```

This will produce the following response:

```
New User Added:
    Username:      'admin'
    Role:          ROLE_ADMIN
    Fingerprint:
2C:BB:FE:F5:EB:AB:1F:EB:35:C3:93:22:A1:91:D9:8B:B3:45:43:F3:9F:8A:CD:3D:14:EE:C6:86:24:5D:5
4:46
    Groups (read and write permission):
        __ANON__
```

---

8. Steps to extract the certificates from the TAK Server so they can be used by clients

The following steps are not in the official TAK Server installation guide.

18. Copy the necessary certificate files to `/home/files` (using the CLI). At this point the current user is `tech` and we are in the home folder. `pwd` should show `/home/tech`.
19. Note the earlier make certificate steps create a lot of files. However only three of these are used by external clients.

```
cd home/tech
mkdir certs
cd certs

cp /opt/tak/certs/files/user.p12  user.p12
cp /opt/tak/certs/files/admin.p12  admin.p12
cp /opt/tak/certs/files/truststore-root.p12  truststore-root.p12
```

20. Change the ownership of these files from `tak` to `tech`

```
cd files
sudo chown tech *.*
```

Now these files can be used by the tech account on the server machine for WebTAK interface. They can also be copied off the server where they can be used by external clients such as ATAK, WinTAK, TAKx or WebTAK.

Note: All clients require the `truststore-root.p12` certificate and one of either `admin.p12` or `user.p12` certificates. The `admin.p12` certificate should only be used by a browser to administer the TAK Server.

At this point the TAK Server has been correctly installed and you have created the certificates required for third party apps to connect to the TAK Server.

---

9. Administer the TAK Server from a Browser installed on the TAK Server

21. Import both the `truststore-root.p12` and `admin.p12` certificates into FireFox (this is the only browser pre-installed on Ubuntu 22.04).
22. Navigate to

```
https://localhost:8443/setup/
```

Follow the prompts to configure the TAK Server. This config process is clear from the prompts so is not described further.

10. Done

Extra Notes

WebTAK can be accessed either with X.509 client certificates (default https port 8443), or by username-password access using OAuth (https port 8446).
In either case, TAK Server must be configured with a server certificate and truststore (see Appendix B).

In the Admin UI menu, use Situation Awareness -> WebTAK to access WebTAK.

Aside on PKI and Certs

A brief primer on PKI - Keys come in two halves, a public key and a private key. The public key can be distributed publicly and widely, and you can use it to verify, but not replicate, information generated using the private key. The private key must be kept secret.

`.key` files are generally the ***private*** key, used by the server to encrypt and package data for verification by clients.

`.csr` is a ***certificate signing request***, a challenge used by a trusted third party to verify the ownership of a keypair without having direct access to the private key (this is what allows end users, who have no direct knowledge of your website, confident that the certificate is valid). In the self-signed scenario you will use the certificate signing request with your own private key to verify your private key (thus self-signed). Depending on your specific application, this might not be needed. (needed for web servers or RPC servers, but not much else).

`.cert` or `.crt` files are the signed ***certificates*** -- basically the "magic" that allows certain sites to be marked as trustworthy by a third party.

`.pem` is a text-based container using base-64 encoding. It could be any of the above files.

```
-----BEGIN EXAMPLE-----
...
-----END EXAMPLE-----
```

Not sure: `.pem` files are generally the ***public*** key, used by the client to verify and decrypt data sent by servers. PEM files could also be encoded private keys, so check the content if you're not sure.

`.p12` is a PKCS12 file, which is a container format usually used to combine the private key and certificate.

A `JKS` keystore is a native file format for Java to store and manage some or all of the components above, and keep a database of related capabilities that are allowed or rejected for each key.

Apparently you can use a private key to generate a p12 keystore then convert it to jks keystore.