Wstęp

JBoss - czyli serwer aplikacji napisanych w Javie. Ze względu iż namęczyłem się z JBossem postanowiłem to opisać tu w tym miejscu bo to co piszą w JBoss Community na oficjalnej stronie to nie działa - a przynajmniej mi nie działało jakiś czas temu. Próbowałem wykonać proste rzeczy na JBossie, które np. w Apachu są bardzo proste, a w JBossie okazały się katorgą dla mnie. JBoss bazuje na Tomcacie, ale jest tak przerobiony, że składnia Tomcata nie zawsze pasuje. Dużo się naszukałem po forach jak co zrobić i może komuś się przyda ten poniższy opis.

Instalacja

Konto

Najpierw zaczynamy od stworzenia konta pod procesy JBossa – tak jest po prostu bezpieczniej – nie zalecam tego robić na koncie root'a. A więc tworzymy użytkownika:

```
$ useradd -c "JBoss" -d /home/users/jboss -G users -m jboss
```

Ustawiamy hasło użytkownikowi jboss:

```
$ passwd jboss
```

i logujemy się na użytkownika jboss.

Java

Niestety ciężko podać tu linka do Javy bo jest on generowany ze strony, więc musicie sami ściągnąć albo zainstalować z poldka.

Jeśli ściągamy ze strony: http://www.sun.com/java/ - to po ściągnięciu kopiujemy na serwer oraz rozpakowujemy:

```
$ chmod +x ./jdk-6u21-linux-i586.bin
$ ./jdk-6u21-linux-i586.bin
```

Tu należy stworzyć symboliczny link, gdybyśmy chcieli uaktualnić Jave w przyszłości nie zmieniając zmiennych tylko podgrywając nową wersje:

```
$ ln -s ./jdk1.6.0_21 ./jdk
```

Jeśli z poziomu poldka to:

```
$ poldek -i java-sun-jdk-base
```

W/w obydwa przypadki są poprawne, ale Java leży w innym miejscu. Jak pobraliśmy ręcznie i rozpakowaliśmy to Java leży w katalogu: **/home/users/jboss/jdk**, albo **/usr/lib/jvm/java-sun** jeśli

instalowaliśmy za pomocą poldka.

JBoss

Ściągamy i rozpakowujemy Jboss'a:

```
$ wget
http://ovh.dl.sourceforge.net/project/jboss/JBoss/JBoss-5.1.0.GA/jboss-5.1.0
.GA.zip
$ unzip ./jboss-5.1.0.GA.zip
```

Tworzymy symboliczny link – głównie dla wygody i ew upgrejdu JBoss'a:

```
$ ln -s ./jboss-5.1.0.GA ./jboss
```

Konfiguracja

Zmienne

Trzeba zdefiniować dwie zmienne jeśli chcemy, aby JBoss się poprawnie uruchomił:

```
$ echo "export JAVA_JBOSS=\$HOME/jboss" >> ./.bashrc
```

Jeśli instalowaliśmy ręcznie Jave to:

```
$ echo "export JAVA_HOME=\$HOME/jdk" >> ./.bashrc
```

jeśli z poziomu poldka to:

```
$ echo "export JAVA_HOME=/usr/lib/jvm/java-sun" >> ./.bashrc
```

Warto też dodać binaria Javy do zmiennej PATH:

```
$ echo "export PATH=\$PATH:\$JAVA_HOME/bin" >> ./.bashrc
```

Zmienne będą obowiązywać po ponownym zalogowaniu się.

JBoss nie uruchomi się gdy nie będzie mógł rozwiązać poprawnie hostname naszej maszyny, należy dodać do pliku /etc/hosts odpowiedni wpis (z poziomu root'a):

```
$ echo „10.0.0.5 nasz_hostname” >> /etc/hosts
```

Nasze IP uzyskamy z wyniku polecenia: **ifconfig**, a **hostname** z polecenia hostname.

JBoss

Domyślnie JBoss będzie nasłuchiwał na **localhoście (127.0.0.1)** jeśli chcemy się do niego dostać z zewnątrz to musimy zmienić wpisy o nasłuchiwaniu JBoss'a. W tym celu edytujemy plik: **$HOME/jboss/jboss-5.1.0.GA/server/default/deploy/jbossweb.sar/server.xml** i szukamy wpisu:

```
      <!-- A HTTP/1.1 Connector on port 8080 -->
      <Connector protocol="HTTP/1.1" port="8080"
address="${jboss.bind.address}"
               connectionTimeout="20000" redirectPort="8443" />
```

i zamieniamy go na:

```
      <!-- A HTTP/1.1 Connector on port 8080 -->
      <Connector protocol="HTTP/1.1" port="8080" address="0.0.0.0"
               connectionTimeout="20000" redirectPort="8443" />
```

Uruchomienie

JBoss'a możemy uruchomić za pomocą polecenia:

```
$ ~/jboss/bin/run.sh
```

Ze względu iż nie bardzo mamy jak wyjść z tego programu więc należało by uruchomić JBoss'a w screenie:

```
$ screen ~/jboss/bin/run.sh
```

Aby wyjść ze screena należy nacisnąć: **CTRL+a CTRL+d**, aby wejść w screena ponownie należy wpisać:

```
$ screen -r
```

JBoss nasłuchuje na porcie **8080 (http)**. Ten port można zmienić w pliku: **$HOME/jboss/jboss-5.1.0.GA/server/default/deploy/jbossweb.sar/server.xml** w sekcji, którą konfigurowaliśmy.

Aby móc przewijać logi w górę w screenie należy wejść w tryb kopiowania: **CTRL+a CTRL+[**, aby wyjść z trybu kopiowania naciskamy: **CTRL+c**. Jeśli chcemy wyłączyć JBoss'a będąc w screenie naciskamy **CTRL+c**, wtedy JBoss dostanie komunikat od systemu, że należy zamknąć demona i zacznie wyłączanie usługi. Jeśli chcemy wyłączyć JBoss'a z poziomu konsoli nie wchodząc do screena należy wykonać komendę:

```
$ ~/jboss/bin/shutdown.sh -S
```

JBoss jest gotowy do pracy i można uploadować aplikacje (rozszerzenia: **war** lub **ear**) do katalogu: **$HOME/jboss/jboss-5.1.0.GA/server/default/deploy**. Z doświadczenia wiem, że kopiowanie aplikacji do katalogu JBoss'a kończy się błędami w JBossie, gdy on jest uruchomiony. Dzieje się tak

ponieważ próbuje on deplojować aplikację w locie co często kończy się porażką bo jeśli aplikacja zbyt długo się kopiuje to on nie może rozpakować aplikacji (**eary** oraz **wary** są pakowane **zipem**) – nie może bo plik nie jest jeszcze cały na dysku. Najbezpieczniej jest zamknięcie JBoss'a, upload aplikacji i uruchomienie JBoss'a. Gdy podgrywamy nowszą wersję naszej aplikacji (**ear** lub **war**) to polecam usuwanie katalogów z tymczasowymi danymi generowanymi przez JBoss'a:

```
$ rm -R ~/jboss/server/default/{work,data,tmp}
```

usuwanie należy wykonywać przy zastopowanym JBossie. Deplojowanie aplikacji możemy też wykonać z konsoli webowej JBoss'a logując się na nią – domyślnie user=pass to admin=admin.

Bezpieczeństwo

Domyślnie JBoss jest zabezpieczony za pomocą bindowania się na **localhosta** – co nie umożliwia potencjalnym hostom dostanie się do niego z domyślnym hasłem. Jednak JBoss'a zbindowaliśmy na zewnętrzny adres i w tym momencie należałoby zmienić hasło oraz odpowiednio zabezpieczyć aplikacje systemowe JBoss'a.

Zmiana hasła

Zmianę hasła przeprowadzamy za pomocą wy-edytowania pliku:
~/jboss/server/default/conf/props/jmx-console-users.properties – domyślnie jest ono:

```
admin=admin
```

zmieńmy na inne:

```
admin=nowehaslomoje123
```

Zmiana hasła nie wymaga restartu JBoss'a.

SSL

Bezpiecznie się łączyć z naszymi aplikacjami za pomocą SSL'a. Najpierw musimy wygenerować odpowiednie certyfikaty:

```
$ openssl genrsa -out ./ssl.key 1024
$ openssl req -new -x509 -days 365 -key ./ssl.key -out ./ssl.crt
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:PL
```

```
State or Province Name (full name) [Some-State]:wojewodztwo
Locality Name (eg, city) []:Miasto
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Nazwa Firmy
Organizational Unit Name (eg, section) []:IT
Common Name (eg, YOUR name) []:nasza_domena.lan
Email Address []:admin@nasza_domena.lan
$ openssl pkcs12 -export -in ./ssl.crt -inkey ./ssl.key -out ssl.p12 -name
jboss
Enter Export Password:
Verifying - Enter Export Password:
$ keytool -importkeystore -srckeystore ./ssl.p12 -srcstoretype pkcs12 -
srcalias jboss -srcstorepass  nasze_tajne_haslo -destkeystore ./ssl.jks -
deststoretype jks -deststorepass nasze_tajne_haslo
```

Mamy wygenerowany certyfikat w formacje **jks**, teraz należy go skopiować do katalogu JBoss'a:

```
$ cp ./ssl.jks ./jboss/server/default/conf/
```

Aby włączyć SSL'a na JBossie musimy go najpierw zatrzymać:

```
$ ~/jboss/bin/shutdown.sh -S
```

Edytujemy plik **$HOME/jboss/jboss-5.1.0.GA/server/default/deploy/jbossweb.sar/server.xml**, znajdujemy sekcje:

```
      <!-- SSL/TLS Connector configuration using the admin devl guide
keystore
      <Connector protocol="HTTP/1.1" SSLEnabled="true"
           port="8443" address="${jboss.bind.address}"
           scheme="https" secure="true" clientAuth="false"
           keystoreFile="${jboss.server.home.dir}/conf/chap8.keystore"
           keystorePass="rmi+ssl" sslProtocol = "TLS" />
      -->
```

i zamieniamy ją na:

```
      <!-- SSL/TLS Connector configuration using the admin devl guide
keystore -->
      <Connector protocol="HTTP/1.1" SSLEnabled="true"
           port="8443" address="0.0.0.0"
           scheme="https" secure="true" clientAuth="false"
           keystoreFile="${jboss.server.home.dir}/conf/ssl.jks"
           keystorePass="nasze_tajne_haslo" sslProtocol = "TLS" />
```

Wartość zmienne **keystorePass** trzeba zmienić z **nasze_tajne_haslo** na hasło, które ustawiliście przy generowaniu certyfikatu.

Startujemy JBoss'a:

```
$ screen ~/jboss/bin/run.sh
```

JBoss nasłuchuje na porcie **8443 (https)** oraz na **8080 (http)**. Te porty można zmienić w pliku: **$HOME/jboss/jboss-5.1.0.GA/server/default/deploy/jbossweb.sar/server.xml**.

Vhosty

Definiowanie vhostów na JBossie jest toporne ponieważ trzeba zrobić odpowiednie wpisy w konfiguracji oraz w pliku aplikacji **ear** albo **war** – co wiąże się z rozpakowaniem i spakowaniem tych plików.

Zatrzymujemy JBoss'a:

```
$ ~/jboss/bin/shutdown.sh -S
```

Edytujemy plik $HOME/jboss/jboss-5.1.0.GA/server/default/deploy/jbossweb.sar/server.xml i dodajemy nową sekcję Host:

```
<Server>
...
    <Service name="jboss.web">
...
      <Engine name="jboss.web" defaultHost="localhost">
...
        <Host name="localhost">
...
        </Host>

        <Host name="nasz.vhost.pl">
            <Alias>alias.vhost.pl</Alias>

            <Alias>alias5.vhost.pl</Alias>

            <Valve
className="org.jboss.web.tomcat.service.jca.CachedConnectionValve"
cachedConnectionManagerObjectName="jboss.jca:service=CachedConnectionManager
"
transactionManagerObjectName="jboss:service=TransactionManager" />
        </Host>
...
      </Engine>
   </Service>
</Server>
```

Teraz trzeba dodać do pliku: **projekt.war/WEB-INF/jboss-web.xml** (jeśli to jest plik **ear** to w tym pliku musi być jeszcze **war** i w nim dodajemy - w tym **warze**)

```
<!DOCTYPE jboss-web PUBLIC
    "-//JBoss//DTD Web Application 4.2//EN"
    "http://www.jboss.org/j2ee/dtd/jboss-web_4_2.dtd">
<jboss-web>
```

```
    <context-root>/</context-root>
    <virtual-host>nasz.vhost.pl</virtual-host>
</jboss-web>
```

Jeśli nie ma takiego pliku: **projekt.war/WEB-INF/jboss-web.xml** to go musimy stworzyć, jeśli jest to musimy w/w treść dodać inteligentnie czyli nagłówek pliku może już być zdefiniowany.

Jak takie coś w bardzo szybki sposób przeprowadzić? Za pomocą programu **mc** – dzięki **unzipowi** i **zipowi** zainstalowanemu w systemie możliwe jest za pomocą **mc** operowanie na pliku **zip** (czyli **war** i **ear** też) wchodząc w niego. Problem się pojawi, gdy jest to plik **ear** wtedy musimy wejść w ten plik, skopiować plik **war** na dysk i wtedy wejść w plik **war** zmodyfikować go i znów skopiować do pliku **ear**. Jeśli w pliku **war** nie ma pliku: **projekt.war/WEB-INF/jboss-web.xml** to go tworzymy na dysku i kopiujemy go do wara za pomocą **mc**. Nie wiem czy dobrze to opisałem i wytłumaczyłem, ale mniej więcej tak to można zrobić;)

Allow/deny

Dostęp po adresie **IP** możemy dać lub też zabronić na poziomie **vhosta** bo właśnie tam się definiuje ACLe. Tu akurat jest bardzo prosto (chyba, że się korzysta z JBoss Community;)) - edytujemy plik: **$HOME/jboss/jboss-5.1.0.GA/server/default/deploy/jbossweb.sar/server.xml** i w sekcji danego hosta tworzymy wpisy:

```
<Valve className="org.apache.catalina.valves.RemoteAddrValve"
allow="10.100.*" deny="*" />
```

Taką regułkę najlepiej dać dla vhosta localhost ponieważ od razu odetniemy dostęp dla hostów z poza 10.100.* do aplikacji systemowych JBoss'a. Oczywiście zamiast 10.100.* dajemy swoje pule adresowe sieci lokalnej, albo tylko dla 127.0.0.1 :)

Zakończenie

Wszelkie uwagi mile widziane na **kamil|at|kamilm.net**, postaram się dopisać w najbliższym czasie jak ustawić odpowiednie parametry do przydzielania pamięci w JBossie bo to jest też bolączka tego serwera aplikacji.