Odlično, prelazimo na Zadatak 6, koji se fokusira na upravljanje ključevima u JKS (Java KeyStore) formatu, sertifikatima i autentikaciju na Tomcat serveru.

Ovo je izuzetno važan i praktičan zadatak.

🧐 Pojašnjenje Zadatka 6

Zadatak ima tri glavna, povezana dijela:

Dio A: Pronalaženje Server Keystore-a (JKS)

1. **Imate:** Veliki broj JKS datoteka (keystore1.jks do keystore50.jks).
2. **Cilj:** Pronaći samo jednu JKS datoteku koja sadrži ključni par (ili sertifikat) koji se može koristiti za **serversku autentikaciju** (SSL/TLS na serveru, npr. Tomcat).
3. **Lozinka:** Svi keystore-ovi su zaštićeni lozinkom **"sigurnost"**.

Dio B: Postavljanje CA i Potpisivanje

Kada se pronađe ispravan JKS:

1. **Kreiranje CA:** Koristiti ključni par iz pronađenog JKS-a za kreiranje **samopotpisanog CA tijela** (Root CA).
2. **Potpisivanje klijentskih sertifikata:** Koristiti taj novi CA za potpisivanje **dva nova klijentska sertifikata**.
3. **Konfiguracija:** Koristiti datoteku **openssl.cnf** (datu u materijalima) za sve OpenSSL operacije.

Dio C: Konfiguracija Tomcata

1. **Serverska autentikacija:** Koristiti pronađenu JKS datoteku za serversku autentikaciju na Tomcatu (tj. da server može dokazati svoj identitet klijentu).
2. **Klijentska autentikacija (Mutual TLS):** Koristiti novopotpisane klijentske sertifikate za klijentsku autentikaciju (tj. da klijent može dokazati svoj identitet serveru).

💻 Pojašnjenje Skripte (skripta.sh)

Skripta koju ste priložili rješava **samo Dio A** zadatka – pronalaženje ispravne JKS datoteke.

Analiza Skripte Liniju po Liniju:

1. Definisanje Liste Keystore-ova

```bash
keystores="keystore*"
```

- keystores: Varijabla koja sadrži wildcard keystore*, što će Bash prevesti u listu svih datoteka koje počinju sa keystore (tj. svih 50 JKS datoteka).

2. Petlja za Iteraciju

```bash
for keystore in $keystores
```

```
do
  # ... komanda
done
```

- Petlja prolazi kroz svaku JKS datoteku (npr. keystore1.jks, keystore2.jks, itd.).

3. Ključna Komanda za Pretragu

```
res=`keytool -list -v -keystore $keystore -storepass sigurnost 2>error1.txt`
```

- **keytool**: Standardni Java alat za upravljanje ključnim skladištima (keystores) i sertifikatima.
- **-list**: Naredba za ispis sadržaja keystore-a.
- **-v**: Verbose (detaljan) ispis. **Ovo je ključno**, jer detaljan ispis uključuje i atribute (proširenja) sertifikata, kao što je **Key Usage** (namjena ključa).
- **-keystore $keystore**: Trenutna JKS datoteka koja se provjerava.
- **-storepass sigurnost**: Lozinka za otvaranje keystore-a.
- **2>error1.txt**: Preusmjerava poruke o grešci (npr. loša lozinka ili neispravan format) u datoteku error1.txt. Standardni izlaz se smješta u varijablu res.

4. Provjera Namjene Ključa (Key Usage)

```
if [[ "$res" =~ "serverAuth" ]]
then
   echo -e "Keystore: $keystore"
   break
fi
```

- **if [["$res" =~ "serverAuth"]]**: Ovo je ključna linija za rješavanje zadatka.
  - Skripta provjerava da li **detaljni izlaz** komande keytool -list -v ($res) sadrži tekst **"serverAuth"**.
  - **"serverAuth"** je kratica za **Server Authentication** (Serverska autentikacija) unutar polja **Extended Key Usage (EKU)** sertifikata. Ako sertifikat sadrži EKU polje sa serverAuth, to znači da je namijenjen za upotrebu od strane web servera (kao što je Tomcat) za SSL/TLS.
- Ako se pronađe serverAuth, ispisuje se ime datoteke i petlja se prekida (break).

5. Rješavanje Zadatka Ovom Skriptom

Skripta uspješno rješava **Dio A** zadatka. Komentarisana linija (#Keystore: keystore22.jks) sugerira da je ispravna datoteka **keystore22.jks**.

🛠 Sugestije i Nastavak Rješavanja (Dio B i C)

Da biste u potpunosti riješili zadatak, trebate nastaviti sa **Djelom B** i **Djelom C**. S obzirom da se CA operacije bolje rade sa OpenSSL-om, a ne keytool-om, moramo konvertovati ključni par iz JKS u OpenSSL format.

Nastavak Skripte za Rješavanje Dijela B (Kreiranje CA i Potpisivanje):

Pretpostavljamo da je keystore22.jks ispravan.

```
# --- NASTAVAK: KORAK 2: KREIRANJE CA I POTPISIVANJE ---

FOUND_JKS="keystore22.jks"
```

```
JKS_PASS="sigurnost"
CA_CNF="openssl.cnf" # Konfiguracioni fajl iz materijala

# 1. Ekstrakcija ključa i sertifikata iz JKS
# Prebacivanje JKS u PKCS#12 (pomoćni korak)
keytool -importkeystore -srckeystore "$FOUND_JKS" -destkeystore temp.p12 -deststoretype
PKCS12 \
    -srcstorepass "$JKS_PASS" -deststorepass "$JKS_PASS" -noprompt

# Ekstrakcija privatnog ključa (CA ključ) i sertifikata (CA sertifikat)
openssl pkcs12 -in temp.p12 -nocerts -nodes -out CA.key -password pass:"$JKS_PASS"
openssl pkcs12 -in temp.p12 -nokeys -out CA.pem -password pass:"$JKS_PASS"

# 2. Inicijalizacija CA baze podataka (potrebno za 'openssl ca')
mkdir -p ca/newcerts
touch ca/index.txt
echo 01 > ca/serial
echo "CA struktura inicijalizovana."

# 3. Generisanje DVA KLIJENTSKA CSR-a (Certificate Signing Request)
# Generisemo kljucni par za klijenta 1
openssl req -new -newkey rsa:2048 -nodes -keyout client1.key -out client1.csr -subj "/CN=Client1"
# Generisemo kljucni par za klijenta 2
openssl req -new -newkey rsa:2048 -nodes -keyout client2.key -out client2.csr -subj "/CN=Client2"

# 4. Potpisivanje KLIJENTSKIH SERTIFIKATA (Koristimo 'openssl ca')
# PAZNJJA: Potrebno je da openssl.cnf ima sekciju za klijentske sertifikate.
# Potpisivanje klijenta 1
openssl ca -config "$CA_CNF" -keyfile CA.key -cert CA.pem -policy policy_anything -extensions
v3_req \
    -in client1.csr -out client1.crt -batch

# Potpisivanje klijenta 2
openssl ca -config "$CA_CNF" -keyfile CA.key -cert CA.pem -policy policy_anything -extensions
v3_req \
    -in client2.csr -out client2.crt -batch

echo "Kreirani sertifikati: client1.crt i client2.crt."

# --- KORAK 3: PRIPREMA ZA TOMCAT AUTENTIKACIJU ---

# 5. Konfiguracija Tomcata
# Za serversku autentikaciju, JKS (keystore22.jks) je spreman za upotrebu.
# Za klijentsku autentikaciju (Mutual TLS):
# - Morate Tomcatu dati CA.pem sertifikat (TrustStore) da bi verifikovao client1.crt i client2.crt.
# - Konfiguracija Tomcat server.xml fajla (koji je vjerovatno server.xml iz materijala) se mora ažurirati
sa:
#   * sslProtocol="TLS"
#   * clientAuth="true"
#   * keystoreFile="keystore22.jks"
#   * keystorePass="sigurnost"
#   * truststoreFile="putanja/do/CA.pem" (ili truststore u JKS formatu sa CA.pem)
```

Zaključak: Skripta rješava prvi i najteži dio (pronalazak JKS-a). Ostatak se svodi na pravilno korištenje keytool i openssl komandi za CA operacije, što je detaljno objašnjeno u nastavku skripte.
Da li želite da pređemo na sljedeći zadatak ili da Vam pomognem da sastavite konfiguracioni dio za Tomcat (server.xml)?