Podpisywanie i bezpieczne uruchamianie apletów

wg http://java.sun.com/docs/books/tutorial/security1.2/

Ograniczanie zabezpieczeń przed uruchamianymi apletami napisanymi przez uwierzytelnianych autorów

Pierwszy sposób zabezpieczania apletów (pint3_1.pdf)

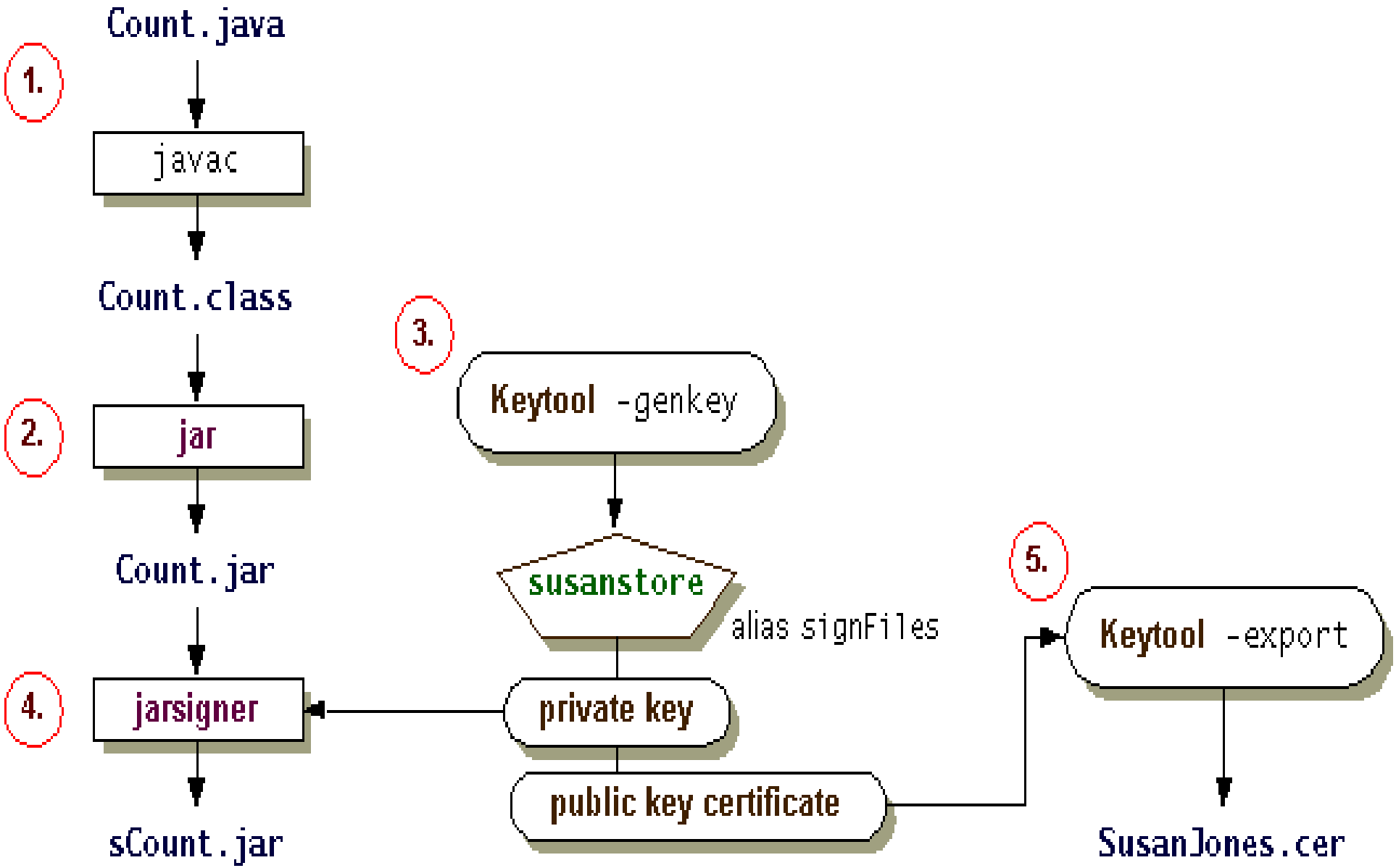
konfigurowanie ochrony za pomocą narzędzia **PolicyTool** - użytkownik apletu określa miejsca pobrania apletu i rodzaj operacji, jakie aplet może wykonać na jego komputerze na podstawie zaufania do źródła pochodzenia apletu

Drugi sposób zabezpieczenia apletów - podpisy cyfrowe

1) złożenie podpisu cyfrowego przez właściciela apletu

2) użytkownik apletu konfiguruje ochronę za pomocą narzędzia **PolicyTool** - określa miejsca pobrania apletu i rodzaj operacji, jakie aplet może wykonać na jego komputerze na podstawie podpisu cyfrowego dołączonego do kodu apletu i związanego z nim klucza głównego, który identyfikuje źródło pochodzenia apletu

Podpisywanie kodu aplikacji uruchamianej przez program java

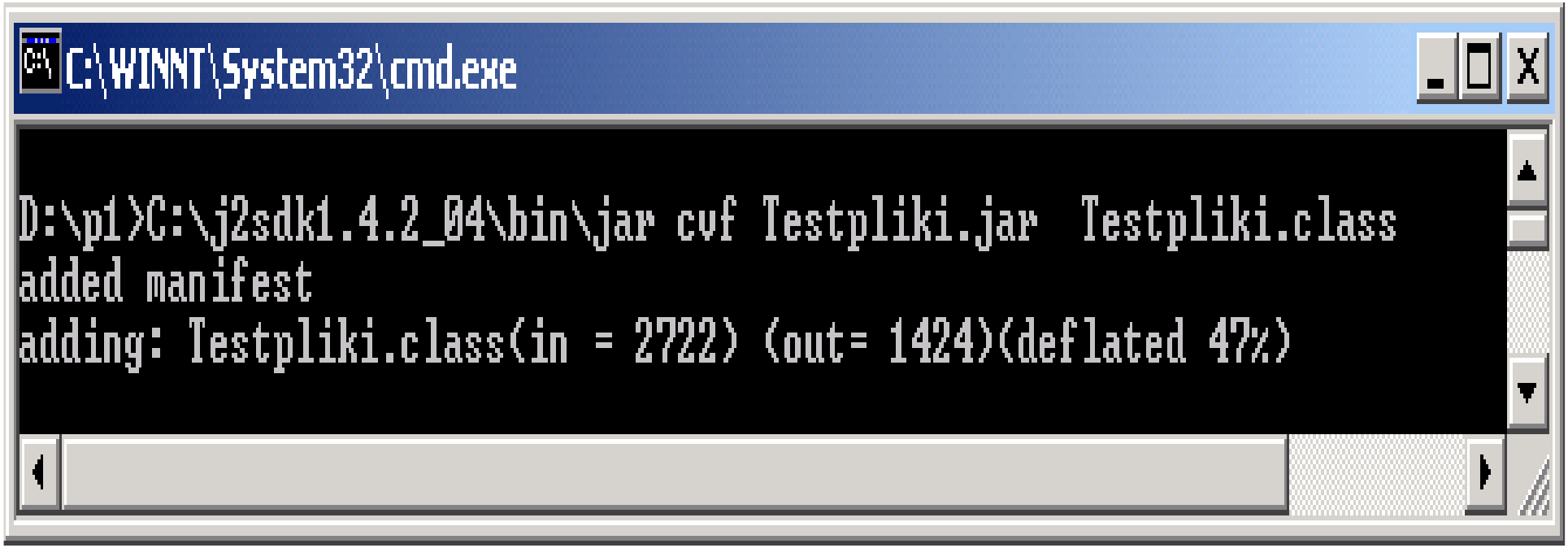


Złożenie podpisu cyfrowego przez właściciela apletu

1. Wykonanie oprogramowania i otrzymanie „”bajtkodu
2. Utworzenie pliku typu JAR zawierającego „bajtkod” za pomocą narzędzia typu jar.
3. Wygenerowanie kluczy prywatnego i publicznego za pomocą wywołania programu keytool -genkey.
4. *Opcjonalne generowanie certyfikatu* – twórca wysyła się żądanie otrzymania certyfikatu *(*certificate signing request -CSR) związany z kluczem publicznym do firmy wydającej certyfikaty bezpieczeństwa i importuje nadany certyfikat (certification authority - CA).
5. Podpisanie kodu typu JAR za pomoca programu jarsigner na podstawie klucza prywatnego i certyfikatu
6. Eksport certyfikatu i klucza publicznego za pomocą programu keytool -export. Użytkownik może teraz otrzymać podpisany plik typu JAR i certyfikat

1. Wykonanie oprogramowania i otrzymanie „”bajtkodu -**utworzenie pliku Testpliki.class i oraz Testpliki.html**
2. Utworzenie pliku typu JAR zawierającego „bajtkod” za pomocą narzędzia typu jar.

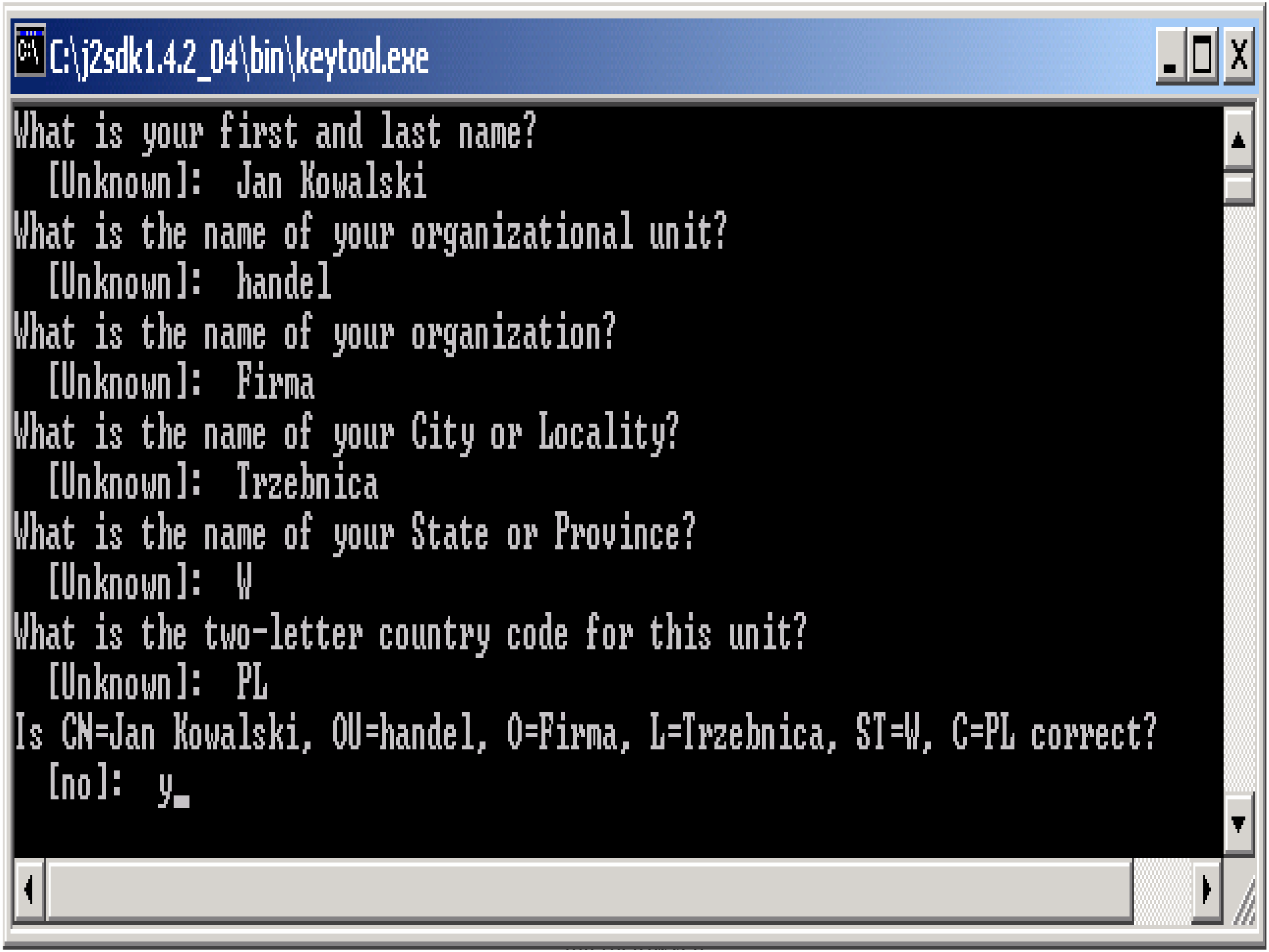
   C:\j2sdk1.4.2_04\bin\jar cvf Testpliki.jar Testpliki.class



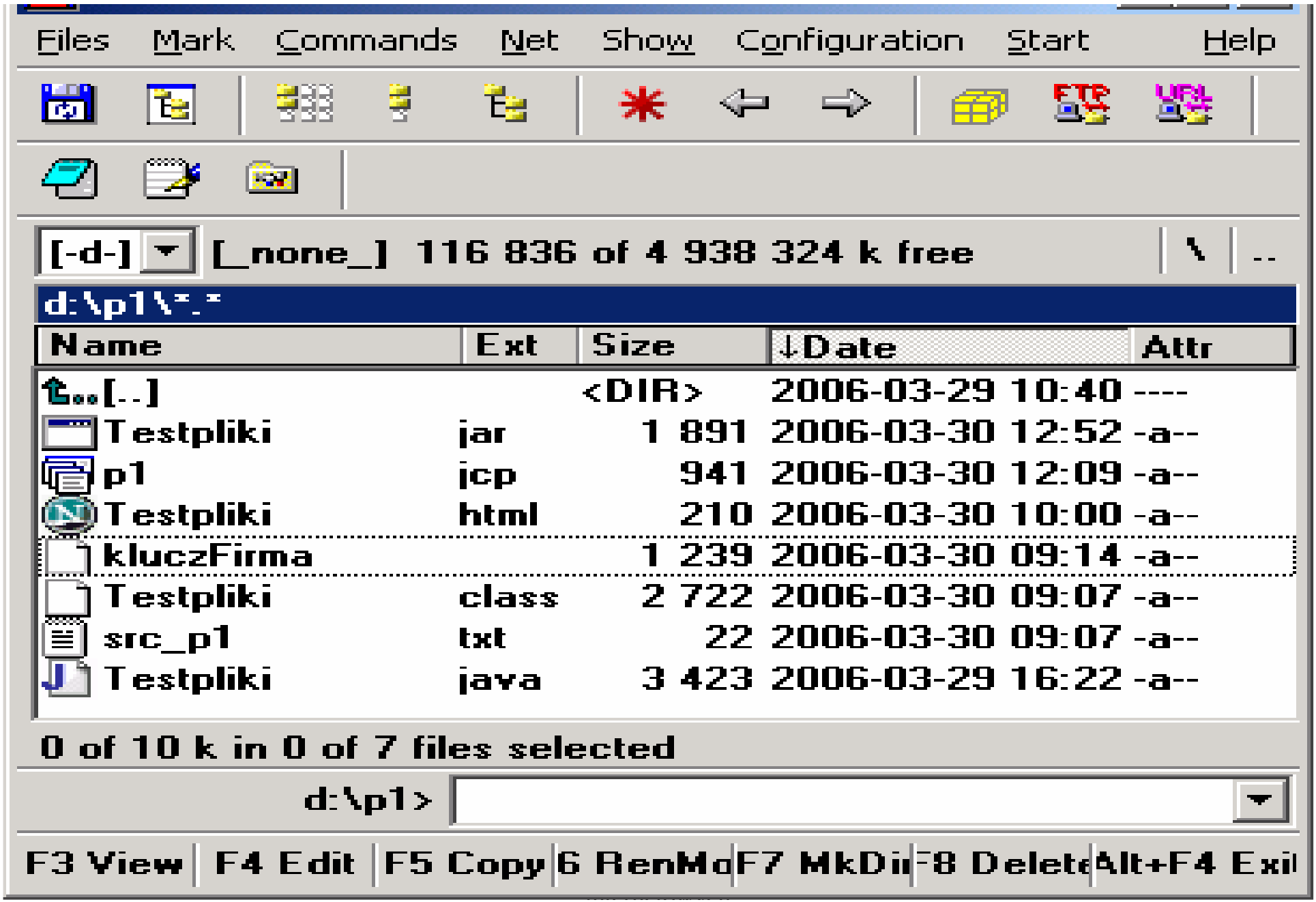
3. Wygenerowanie kluczy prywatnego i publicznego za pomocą wywołania programu **keytool -genkey.**

C:\j2sdk1.4.2_04\bin\keytool -genkey -alias Jan_1 -keypass klucz123 -keystore kluczFirma -storepass Firm456a

- polecenie generowania kluczy ***-genkey*.**
- ***-alias Jan_1*** oznacza alias używany w przyszłości i udostępniający wygenerowane klucze (keystore entry)
- ***-keypass klucz123*** specyfikuje hasło dla wygenerowanego klucza prywatnego. To hasło zawsze udostępnia klucz prywatny (keystore entry).
- ***-keystore kluczFirma*** oznacza nazwę pliku (i ścieżkę) zawierające definicje wygenerowanych kluczy
- ***-storepass Firm456a*** oznacza hasło dla wygenerowanych kluczy w pliku ***kluczFirma*.** Plik powstaje w katalogu bieżącym dla programu *keytool*


C:\j2sdk1.4.2_04\bin\keytool.exe
What is your first and last name?
[Unknown]: Jan Kowalski
What is the name of your organizational unit?
[Unknown]: handel
What is the name of your organization?
[Unknown]: Firma
What is the name of your City or Locality?
[Unknown]: Trzebnica
What is the name of your State or Province?
[Unknown]: W
What is the two-letter country code for this unit?
[Unknown]: PL
Is CN=Jan Kowalski, OU=handel, O=Firma, L=Trzebnica, ST=W, C=PL correct?
[no]: y

Stan katalogu, w którym powstają kolejne pliki



Program ***keytool*** tworzy:

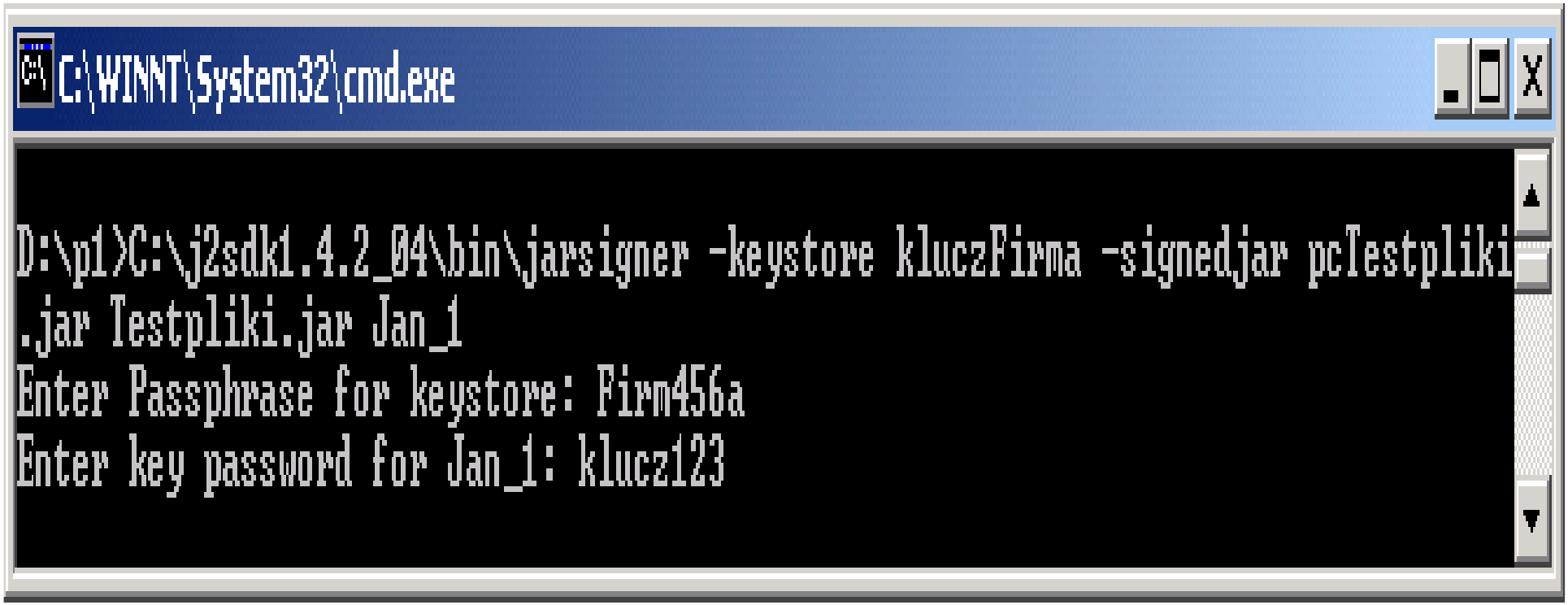
- plik z parą *klucz publiczny/klucz prywatny* o nazwie *kluczFirma* przypisujac mu hasło *Firm456a*
- encje z opisem polskiej firmy handlowej *Firma* z Trzebnicy i pracownika tej formy, Jana Kowalskiego.
- Program ten tworzy własny certfikat (bez korzystania z zewnętrznej firmy wydającej certyfikaty bezpieczeństwa), który zawiera:
  - ✓ klucz publiczny
  - ✓ encje z opisem firmy – główne pole certyfikatu.
  - ✓ jest związany z wygenerowanym kluczem prywatnym, który z kolei jest związany z aliasem *Jan_1*. Klucz prywatny jest związany z hasłem *klucz123*.
  - ✓ certyfikat jest ważny przez 90 dni, chyba że zostanie ustawiony za pomica opcji *-validity* na inny okres.

5. Podpisanie kodu typu JAR za pomoca programu jarsigner na podstawie klucza prywatnego i certyfikatu

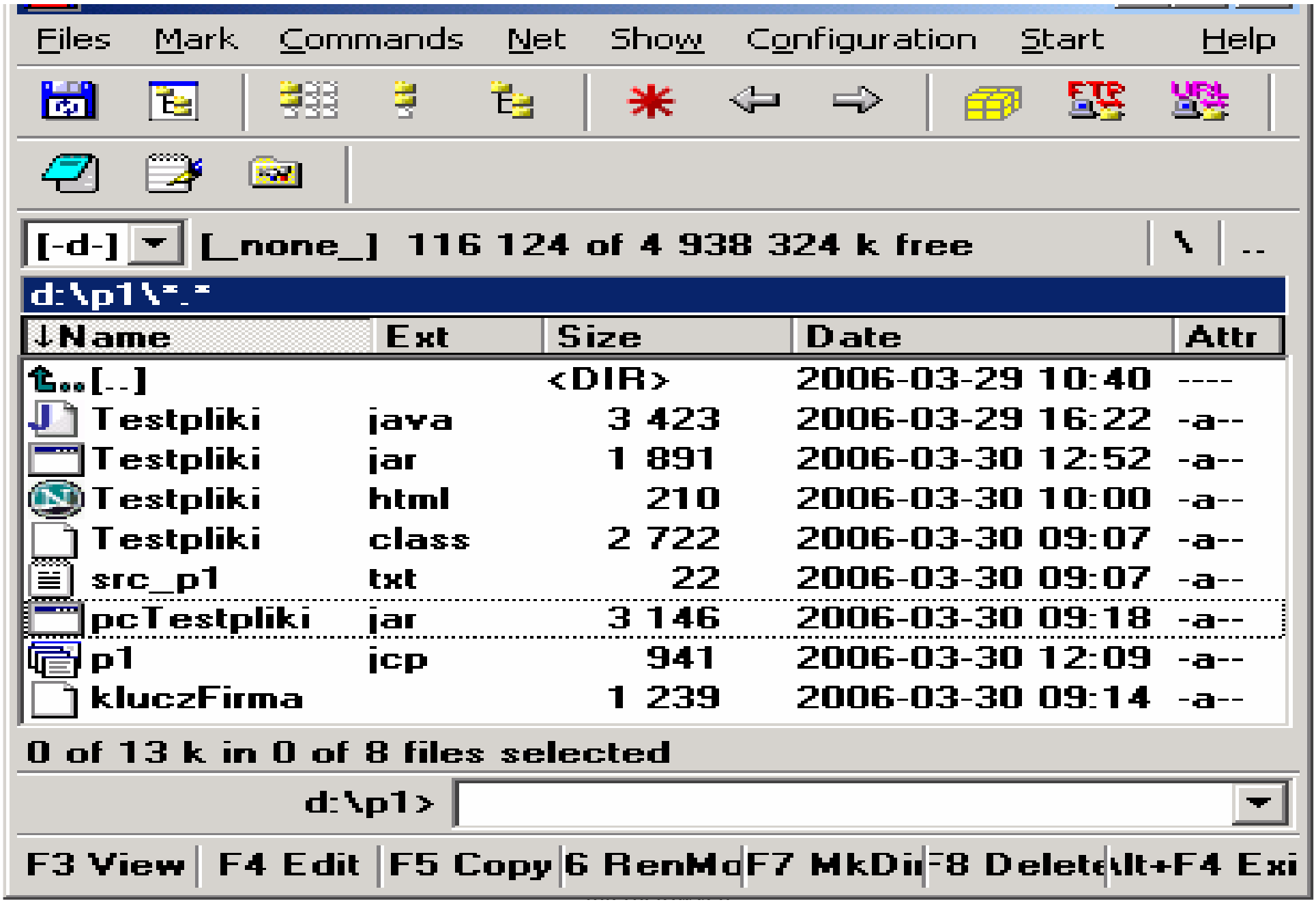
C:\j2sdk1.4.2_04\bin\jarsigner -keystore kluczFirma -signedjar pcTestpliki.jar Testpliki.jar Jan_1

Program **jarsinger:**

- podpisuje plik Testpliki.jar używając klucza prywatnego związanego z aliasem Jan_1 i tworzy plik pcTestplik.jar
- każe potwierdzić hasłem Firm456a plik kluczFirma i osobno klucz prywatny hasłem klucz123
- dokonuje ekstrakcji certyfikatu z pliku kluczFirma związanego z aliasem Jan_1 i dołącza do podpisu cyfrowego w pliku pcTestpliki.jar


C:\WINNT\System32\cmd.exe
D:\p1>C:\j2sdk1.4.2_04\bin\jarsigner -keystore kluczFirma -signedjar pcTestpliki
.jar Testpliki.jar Jan_1
Enter Passphrase for keystore: Firm456a
Enter key password for Jan_1: klucz123

Stan katalogu, w którym powstają kolejne pliki

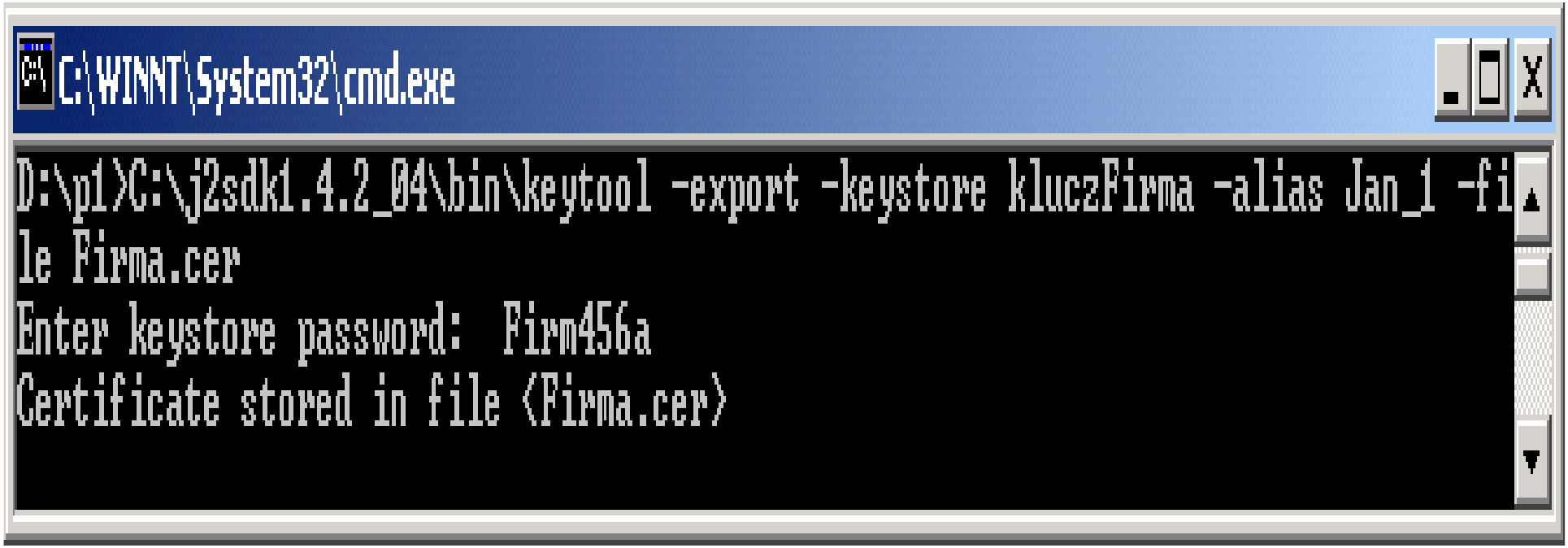


6. Eksport certyfikatu i klucza publicznego za pomocą programu keytool -export.

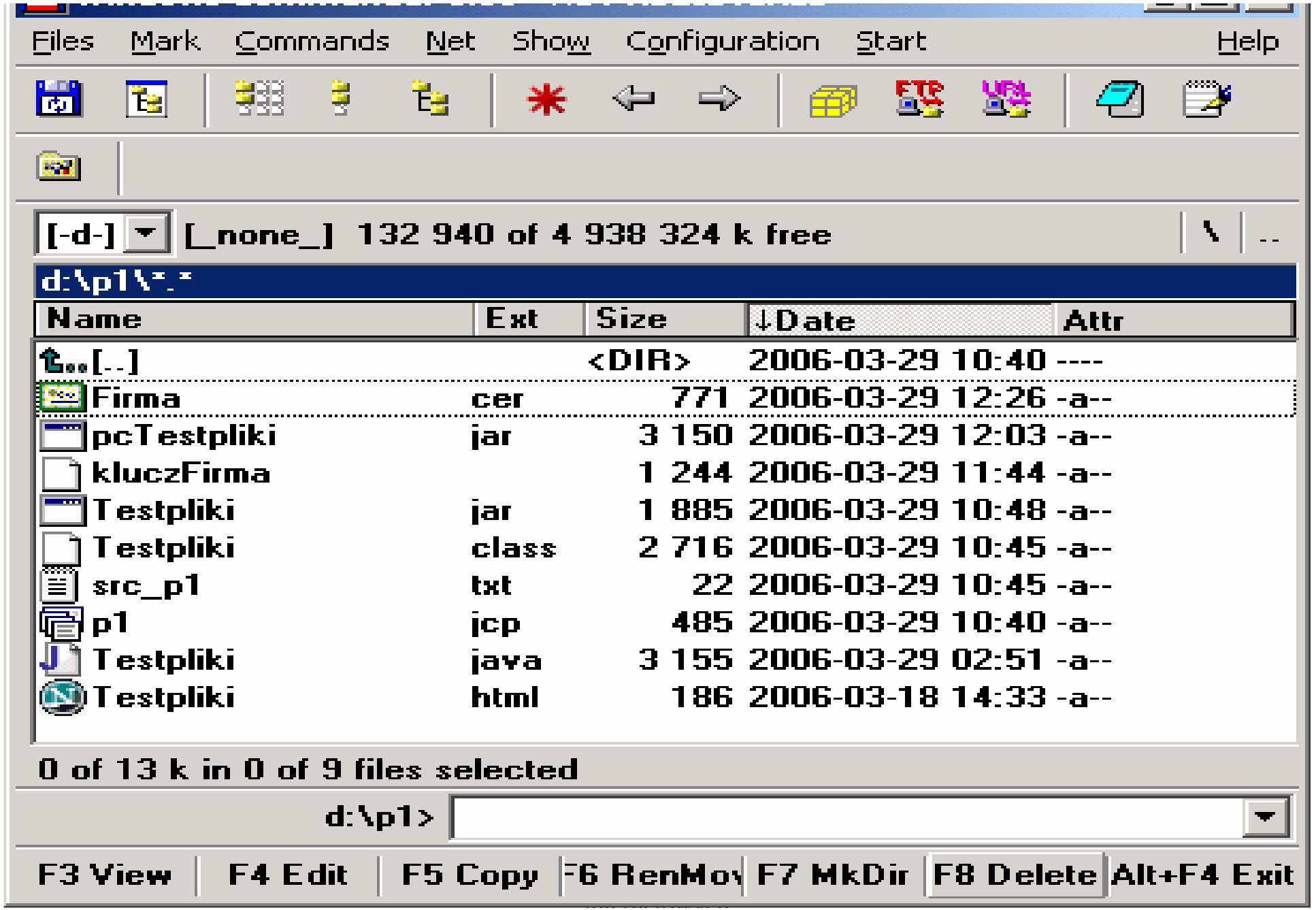
C:\j2sdk1.4.2_04\bin\keytool -export -keystore kluczFirma -alias Jan_1 -file Firma.cer

• użytkownik podpisanego apletu oczekuje na uwierzytelnienie podpisu cyfrowego, który posiada aplet Testpliki.class zawarty w podpisanym pliku pcTestpliki.jar, kiedy ten aplet dokonuje operacji na plikach na komputerze użytkownika, a dostępu broni systemowa polisa bezpieczeństwa policy.security.

• do uwierzytelnienia podpisu użytkownik potrzebuje klucz prywatny odpowiadający podpisowi. Twórca apletu musi wysłać użytkownikowi kopię certyfikatu uwierzytelnionego kluczem publicznym (dane z pliku kluczFirma) tworząc jego kopię w pliku Firma.cer. Należy potwierdzić tworzenie kopii hasłem Firm456a.


C:\WINNT\System32\cmd.exe
D:\p1>C:\j2sdk1.4.2_04\bin\keytool -export -keystore kluczFirma -alias Jan_1 -fi
le Firma.cer
Enter keystore password: Firm456a
Certificate stored in file <Firma.cer>

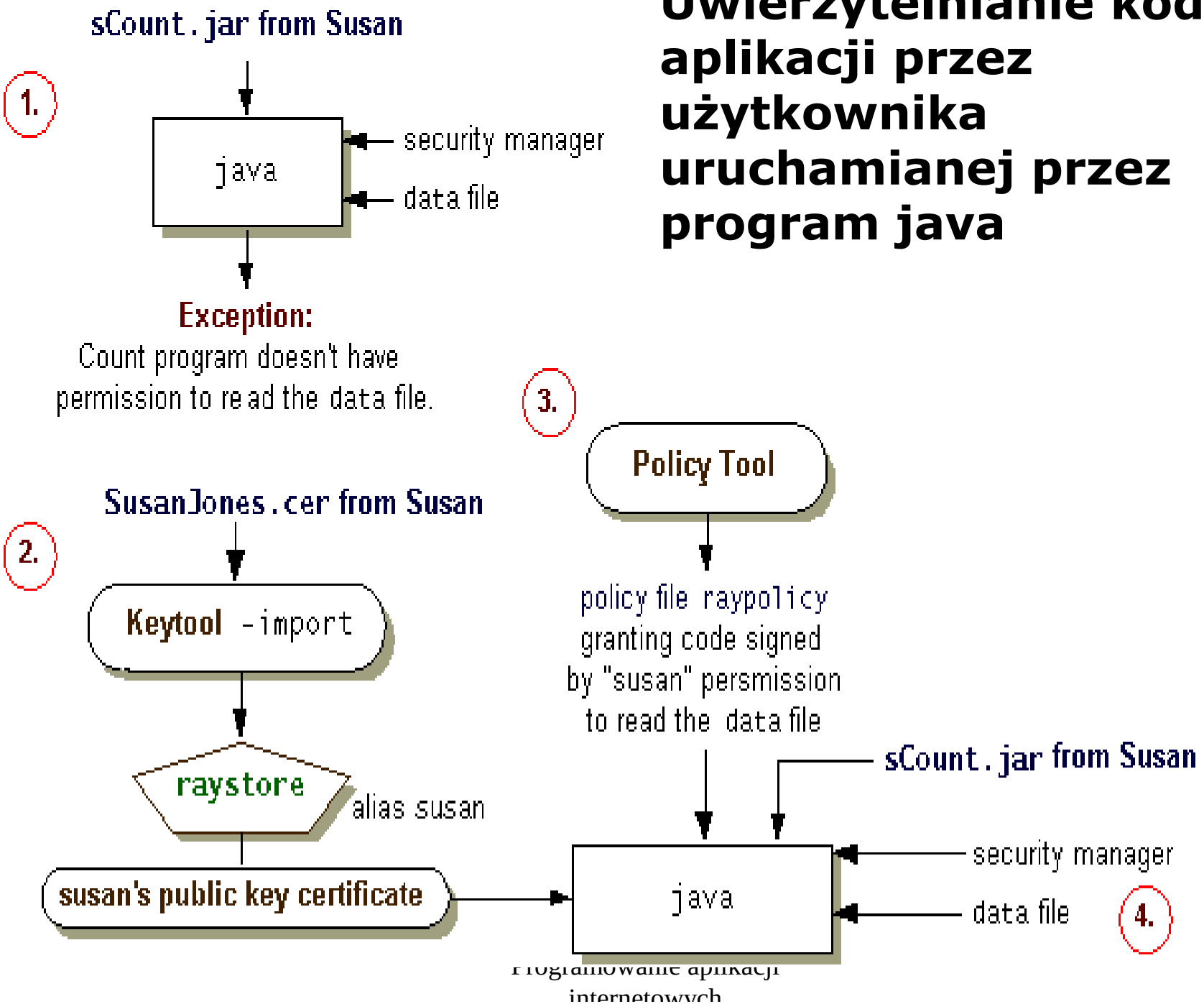
Stan katalogu, w którym powstają kolejne pliki



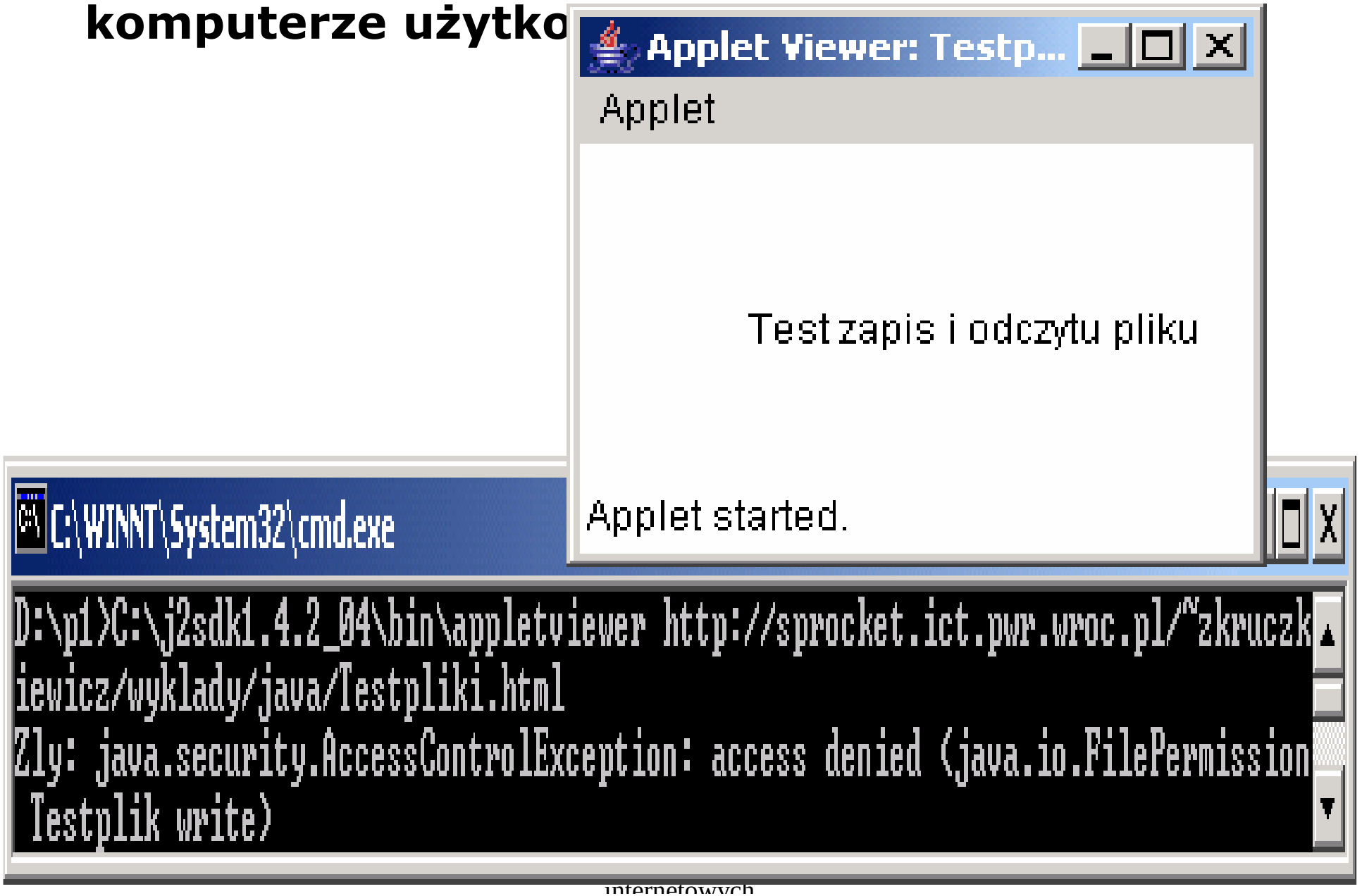
Użytkownik apletu konfiguruje ochronę przed uruchamianym apletem posiadającym podpis cyfrowy – uwierzytelnianie kodu apletu

1. Obserwuje zachowanie apletu – aplet nie może zapisać i odczytać pliku na komputerze użytkownika
2. Importuje certyfikat jako certyfikat uwierzytelniający używając programu **keytool -import** oraz aliasu **Jan** dla importowanego certyfikatu.
3. Konfiguruje plik uwierzytelniający ***mojapolisa1*** za pomocą programu **PolicyTool,** określający zakres uprawnień dla apletu np. w zakresie operacji plikowych, oznaczony aliasem **Jan.**
4. Testuje skutki konfigurowania zakresu uprawnień nadanych apletowi - aplet może zapisać i odczytać pliku na komputerze użytkownika

Uwierzytelnianie kodu aplikacji przez użytkownika uruchamianej przez program java



1. **Użytkownik obserwuje zachowanie apletu – aplet nie może zapisać i odczytać pliku na komputerze użytko**



```
<HTML>
<HEAD>
</HEAD>
<BODY BGCOLOR="000000">
<CENTER>
<APPLET    archive  = "pcTestpliki.jar"
           code     =  "Testpliki.class"
           width    =  "200"
           height   =  "100"
   >
</APPLET>
</CENTER>
</BODY>
</HTML>
```

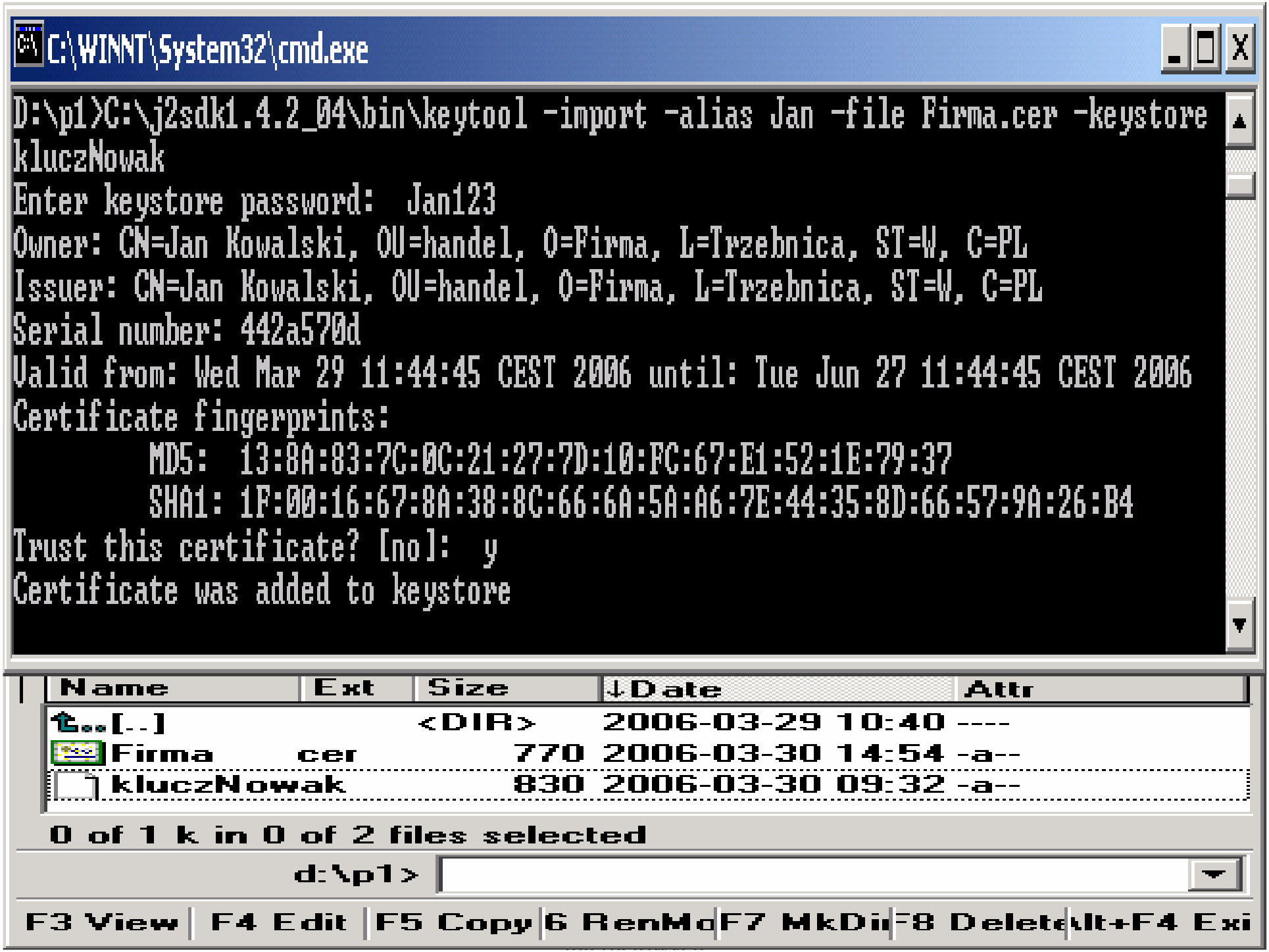
2. Użytkownik importuje certyfikat Jana jako certyfikat uwierzytelniający używając programu keytool -import oraz aliasu Jan

Użytkownik Nowak otrzymuje od Jana:

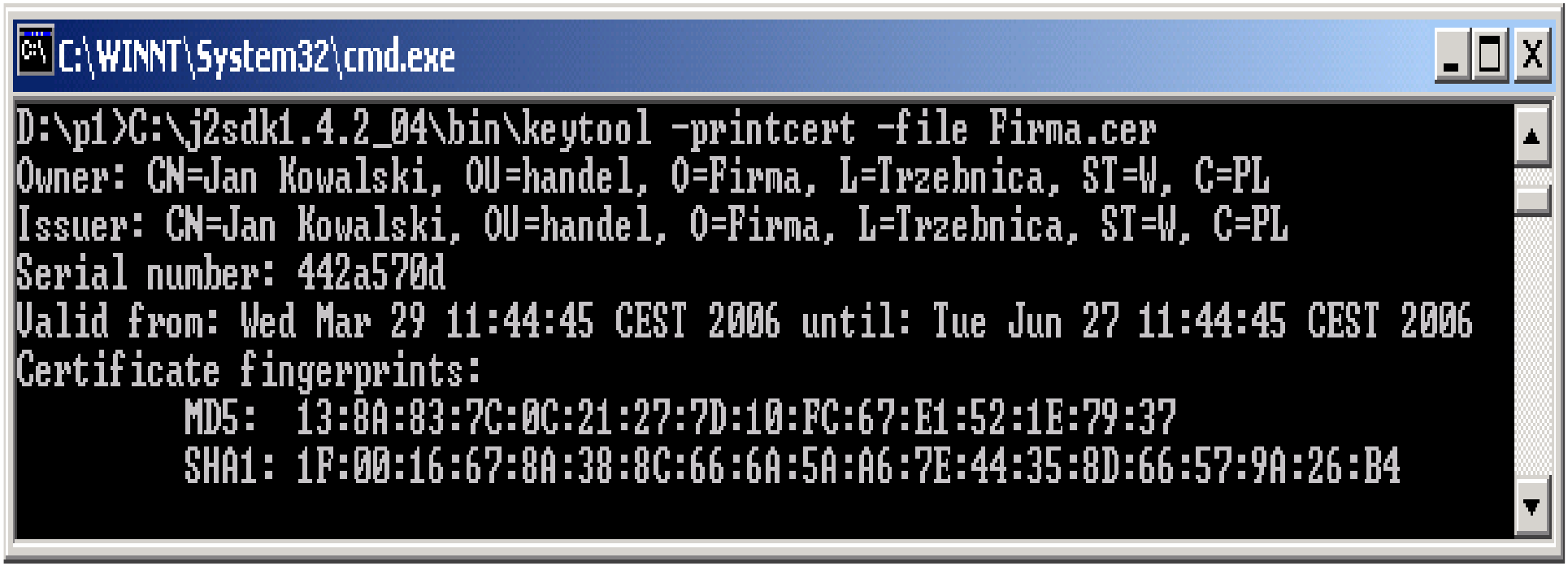
- **plik Firma.cer** zawierający certyfikat z kluczem publicznym odpowiadającym kluczowi prywatnemu użytemu do podpisania apletu.
- podpisany plik **pcTestplik.jar**

Przebieg importu i utworzenie pliku kluczNowak z kluczami po stronie użytkownika:

- przejście do katalogu bieżącego z otrzymanym od Jana plikiem **Firma.cer** z kluczem publicznym
- Wywołanie programu **C:\j2sdk1.4.2_04\bin\keytool -import -alias Jan -file Firma.cer -keystore kluczNowak**
- Użytkownik potwierdza hasłem **Jan123** utworzenie pliku z importowanym certyfikatem **kluczNowak**

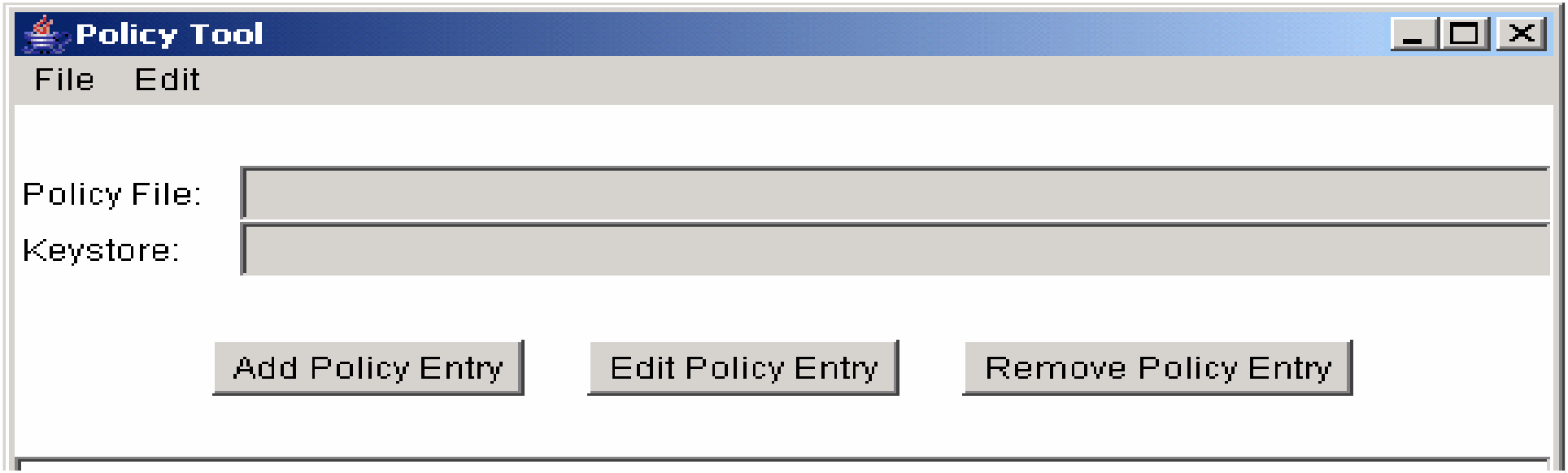

C:\WINNT\System32\cmd.exe
D:\p1>C:\j2sdk1.4.2_04\bin\keytool -import -alias Jan -file Firma.cer -keystore kluczNowak
Enter keystore password: Jan123
Owner: CN=Jan Kowalski, OU=handel, O=Firma, L=Trzebnica, ST=W, C=PL
Issuer: CN=Jan Kowalski, OU=handel, O=Firma, L=Trzebnica, ST=W, C=PL
Serial number: 442a570d
Valid from: Wed Mar 29 11:44:45 CEST 2006 until: Tue Jun 27 11:44:45 CEST 2006
Certificate fingerprints:
MD5: 13:8A:83:7C:0C:21:27:7D:10:FC:67:E1:52:1E:79:37
SHA1: 1F:00:16:67:8A:38:8C:66:6A:5A:A6:7E:44:35:8D:66:57:9A:26:B4
Trust this certificate? [no]: y
Certificate was added to keystore
Name Ext Size ↓Date Attr
[..] <DIR> 2006-03-29 10:40 ----
Firma cer 770 2006-03-30 14:54 -a--
kluczNowak 830 2006-03-30 09:32 -a--
0 of 1 k in 0 of 2 files selected
d:\p1>
F3 View F4 Edit F5 Copy 6 RenMo F7 MkDi F8 Delete Alt+F4 Exi

Wydruk certyfikatu twórcy apletu na żądanie użytkownika

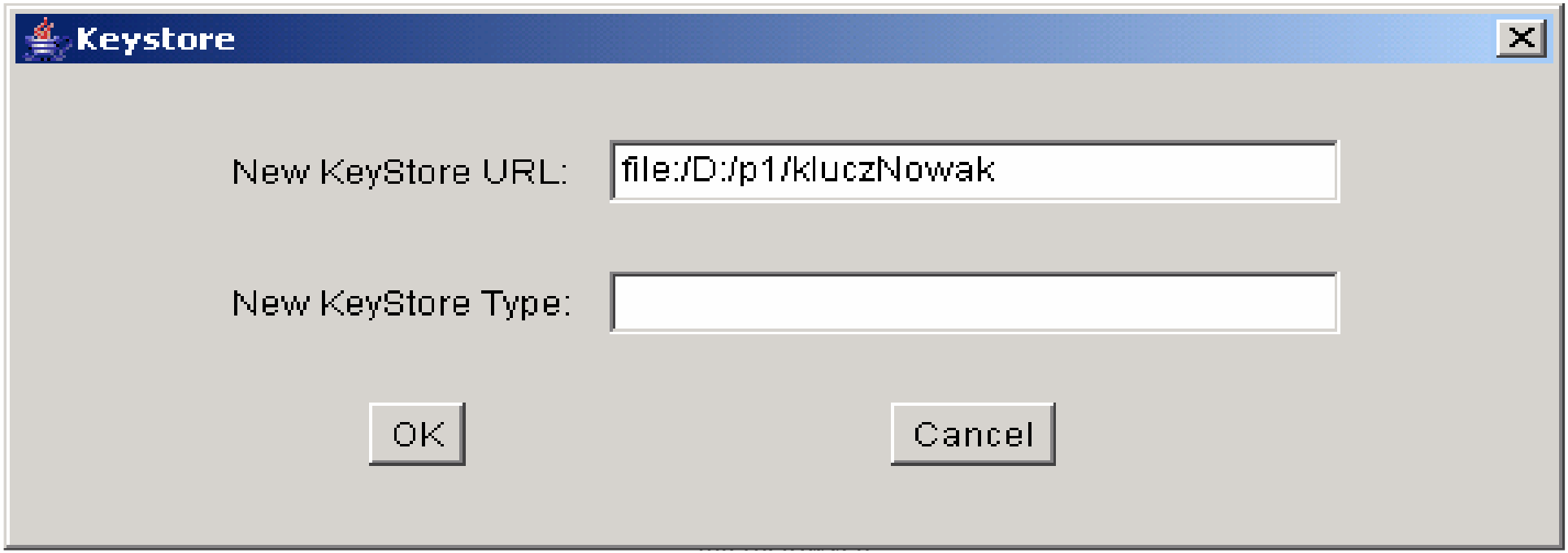


3. Użytkownik konfiguruje plik uwierzytelniający *mojapolisa1* za pomocą programu PolicyTool, określający zakres uprawnień dla apletu np. w zakresie operacji plikowych.

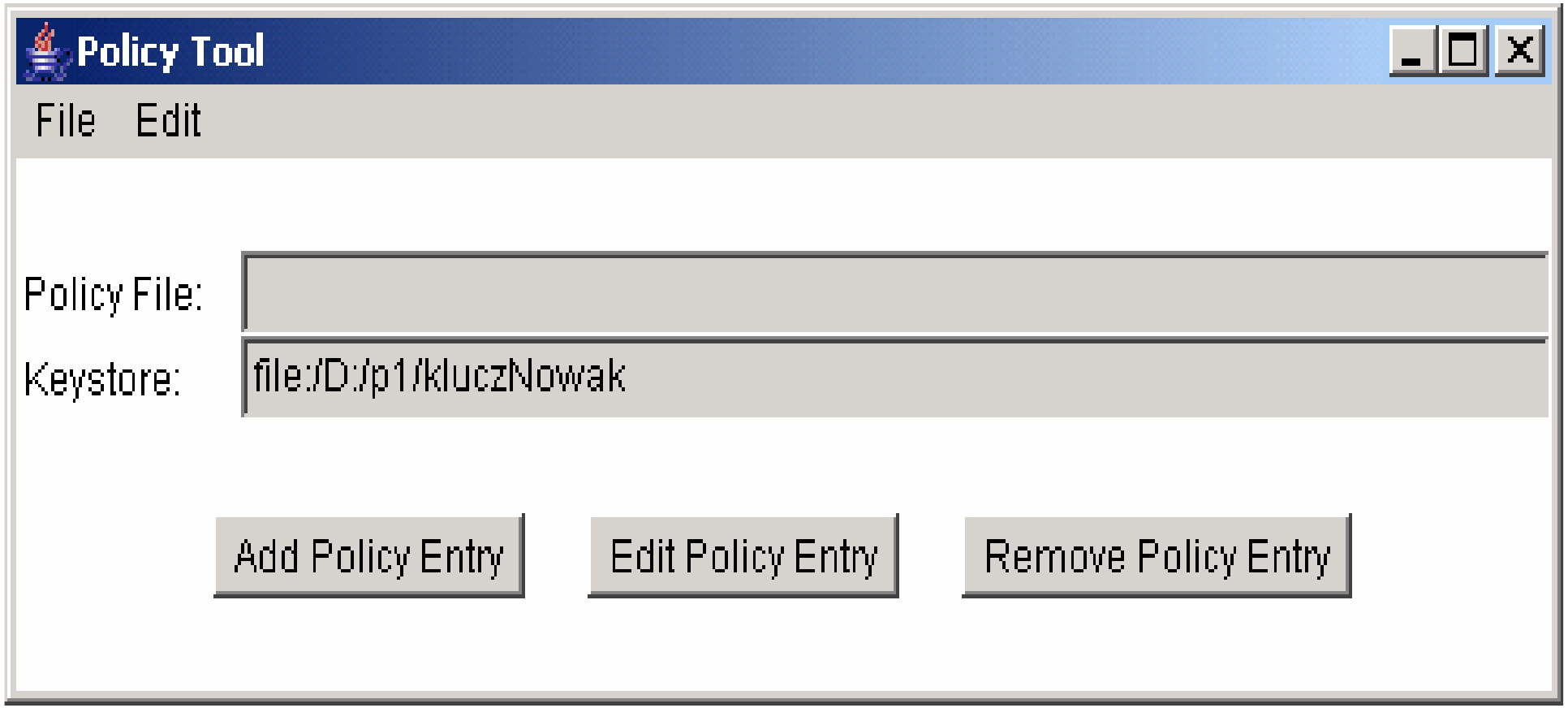
- Uruchomienie programu PolicyTool
- Specyfikcja pliku **kluczNowak** z kopią certyfikatu i klucza publicznego **file:/d:/p1/kluczNowak**
- Dodanie **CodeBase** przez podanie adresu URL apletu **http://sprocket.ict.pwr.wroc.pl/~zkruczkiewicz/wyklady/java/*** lub **file:/d:/p1/*** lub pustej linii
- Dodanie nowego uwierzytelnienia oznaczonego **SignedBy**, czyli **alias Jan**
- Podanie miejsca, nazwy, rodzaju operacji przetwarzanych plików np. **D:\p2\Testplik**
- Zapis danych bezpieczeństwa w pliku **mojapolisa1**



W opcii **Edit** specyfikuje się adres URL kopii certyfikatu **kluczNowak**



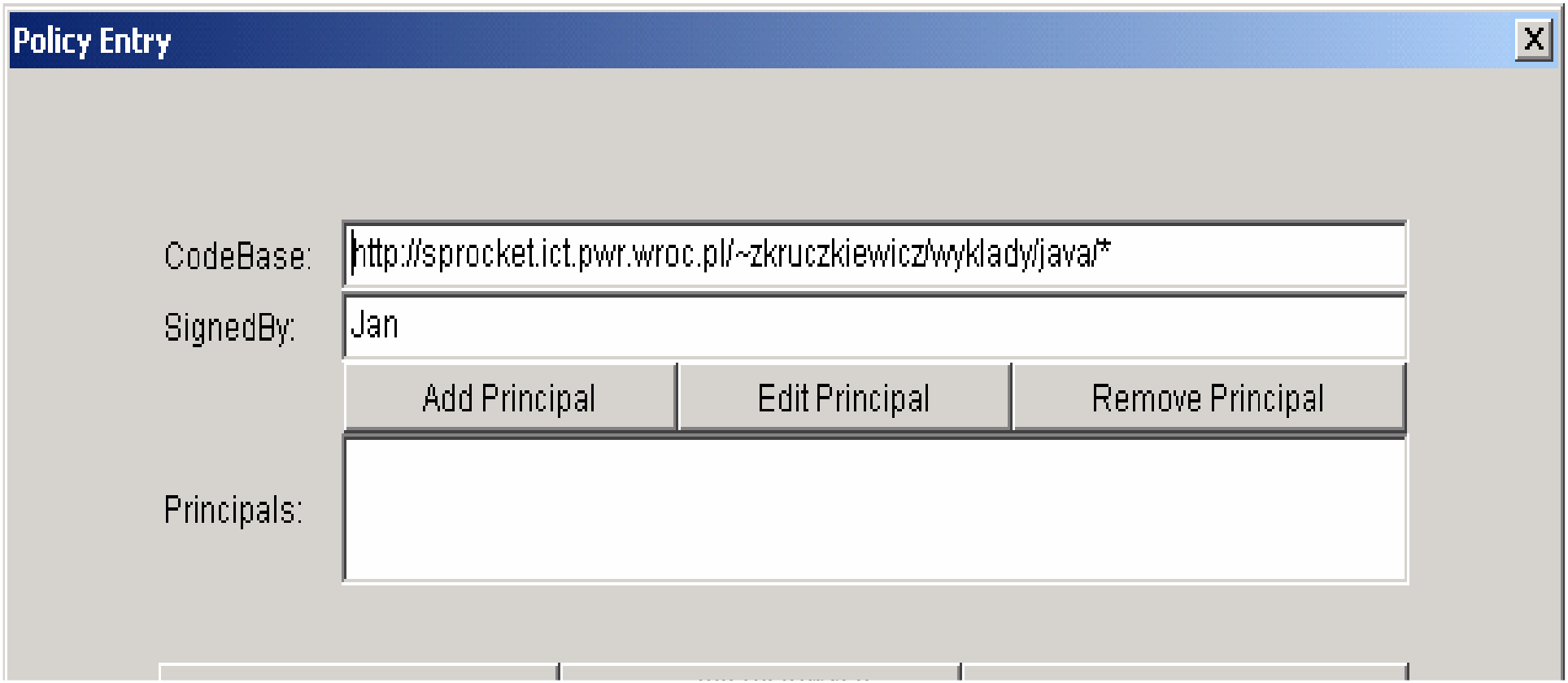
Plik kluczNowak zawiera kopię certyfikatu i klucza publicznego - należy podać jego nazwę w postaci file:/d:/p1/kluczNowak

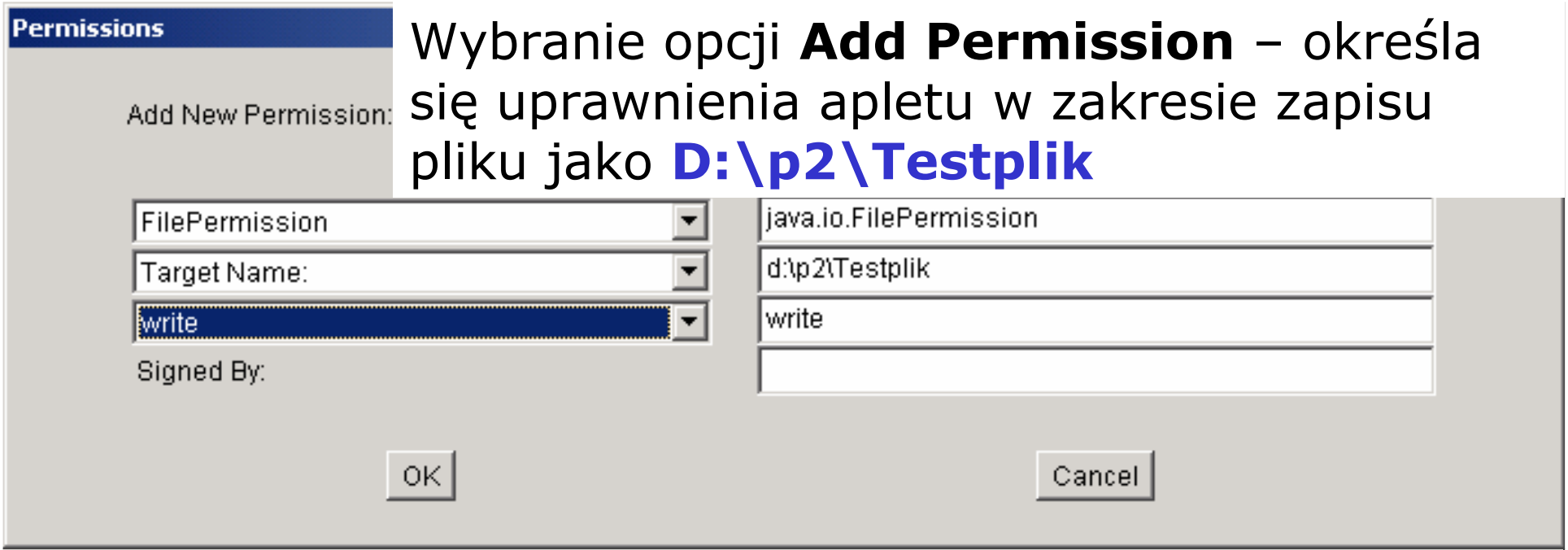


Po wybraniu opcji **Add Policy Entry** podaje się:

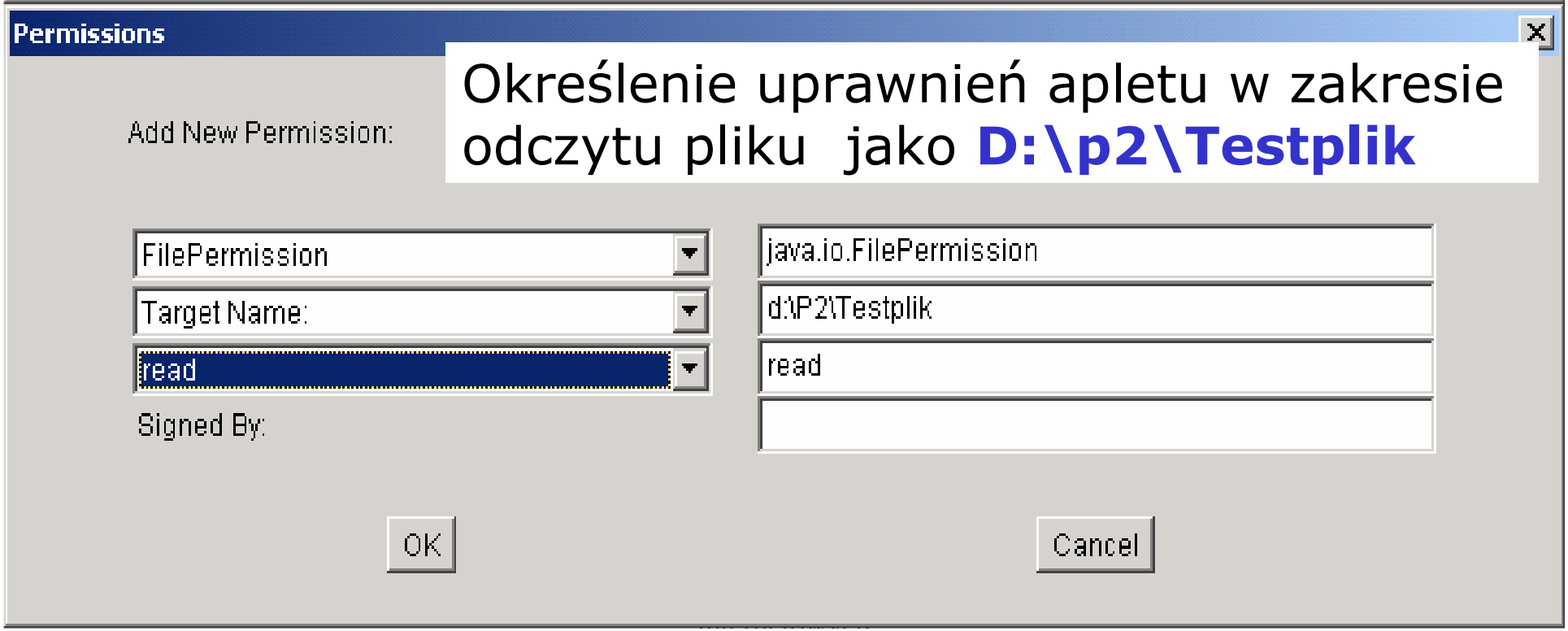
- **SignedBy** podając alias **Jan** dla kopii certyfikatu **kluczNowak** (związanego z kluczem prywatnym) zawierającego klucz publiczny apletu
- **CodeBase** przez podanie adresu URL

http://sprocket.ict.pwr.wroc.pl/~zkruczkiewicz/wklady/java/* lub **file:/d:/p1/*** lub pustej linii

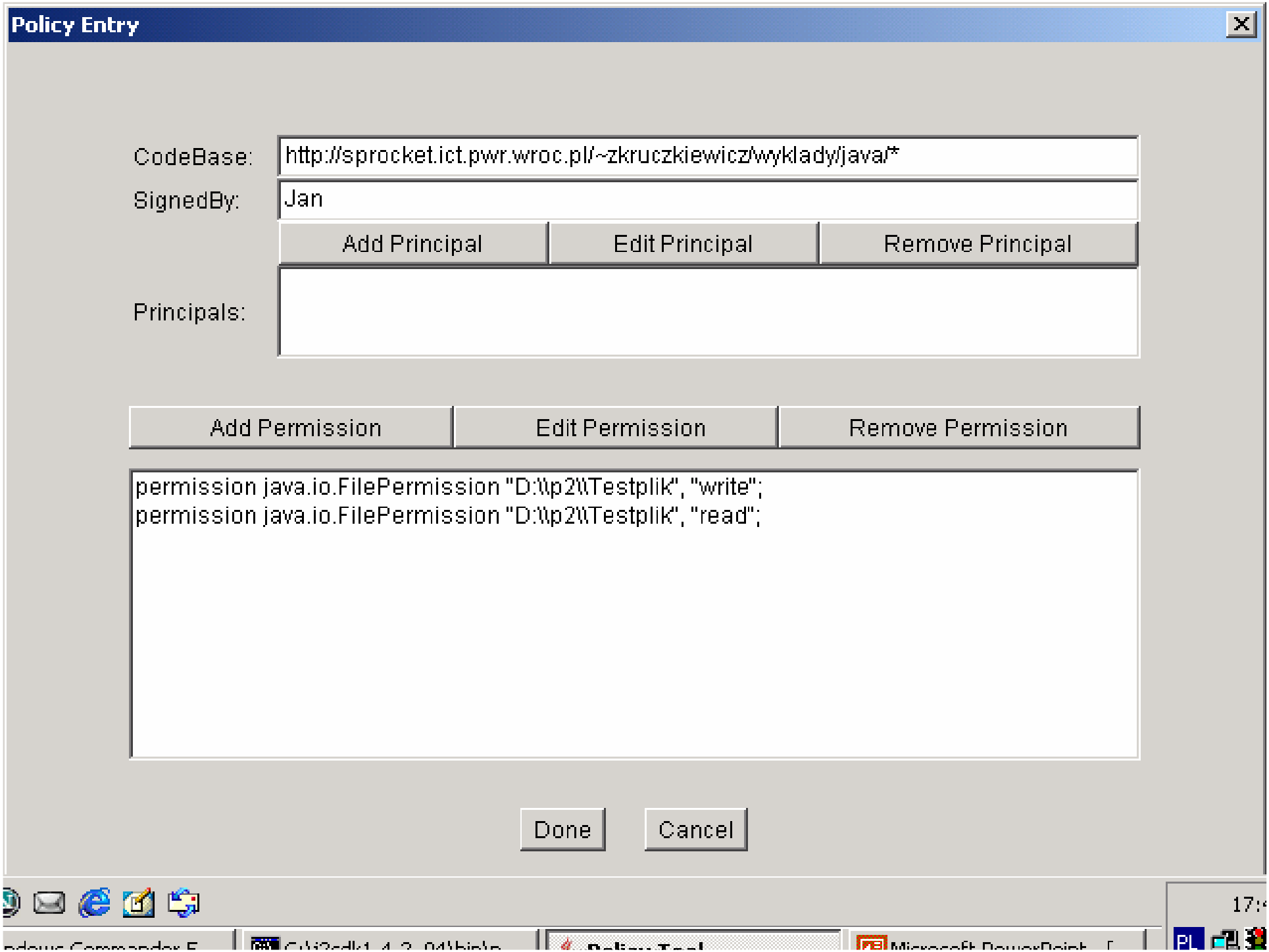




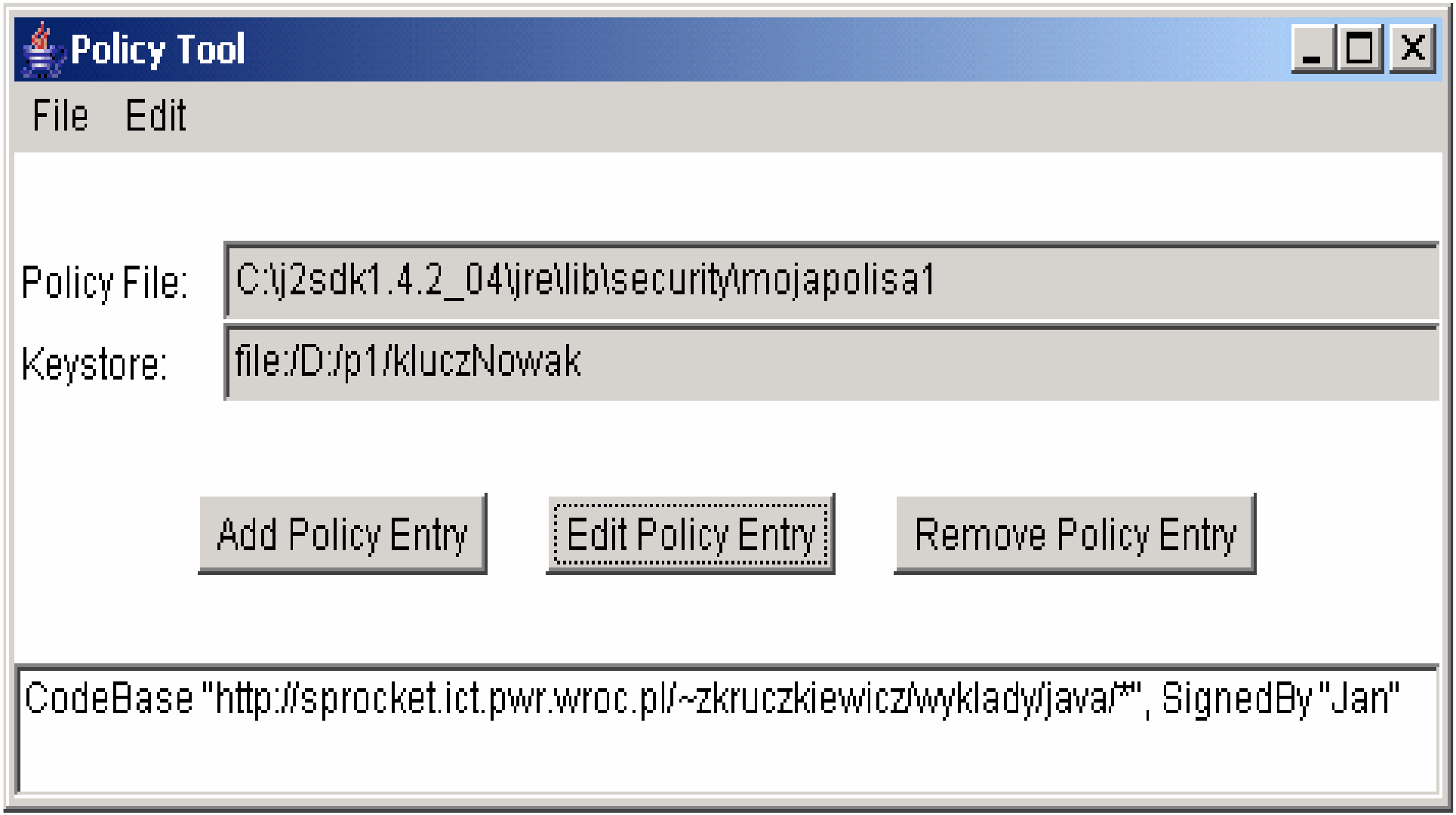
Wybranie opcji **Add Permission** – określa się uprawnienia apletu w zakresie zapisu pliku jako **D:\p2\Testplik**



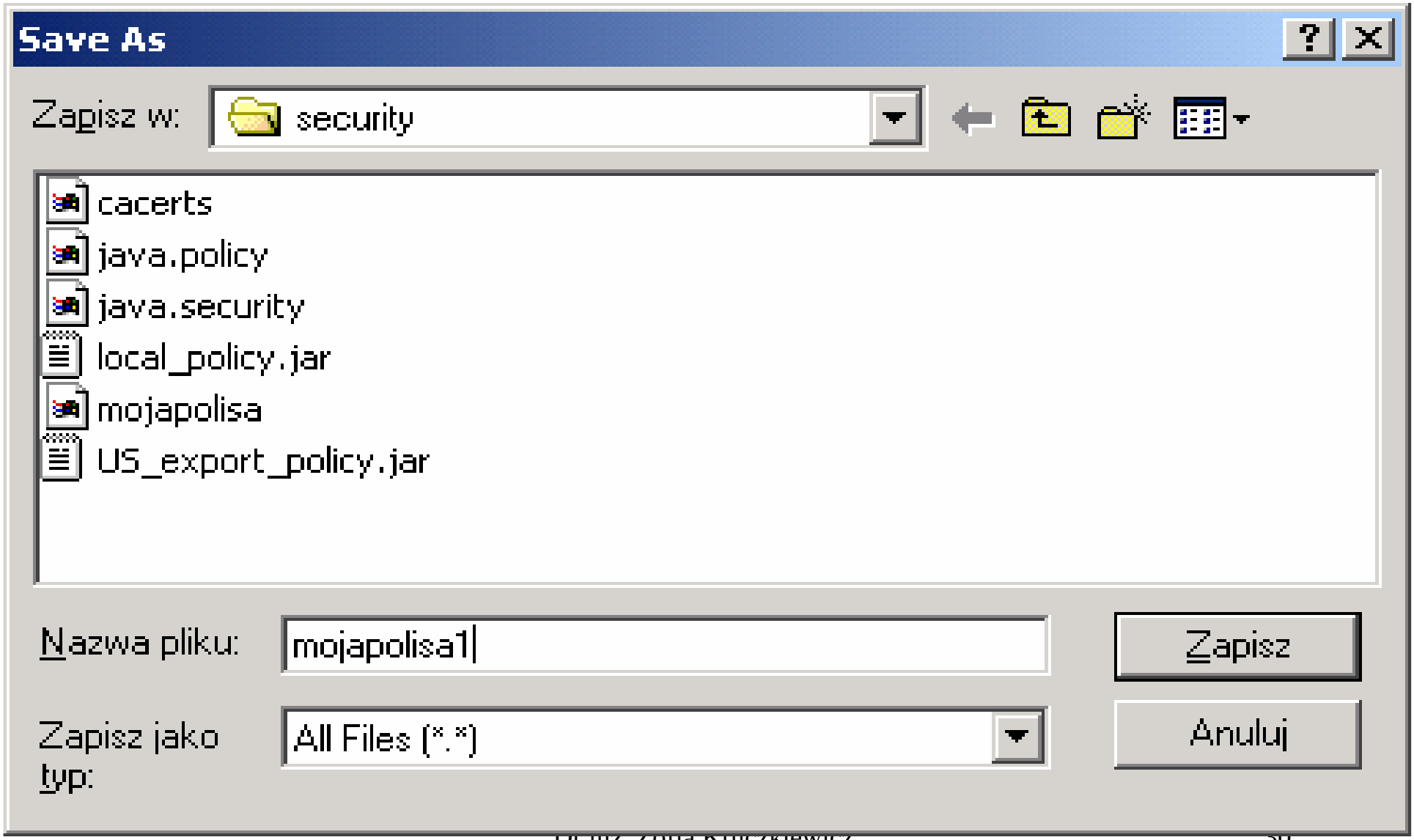
Określenie uprawnień apletu w zakresie odczytu pliku jako **D:\p2\Testplik**

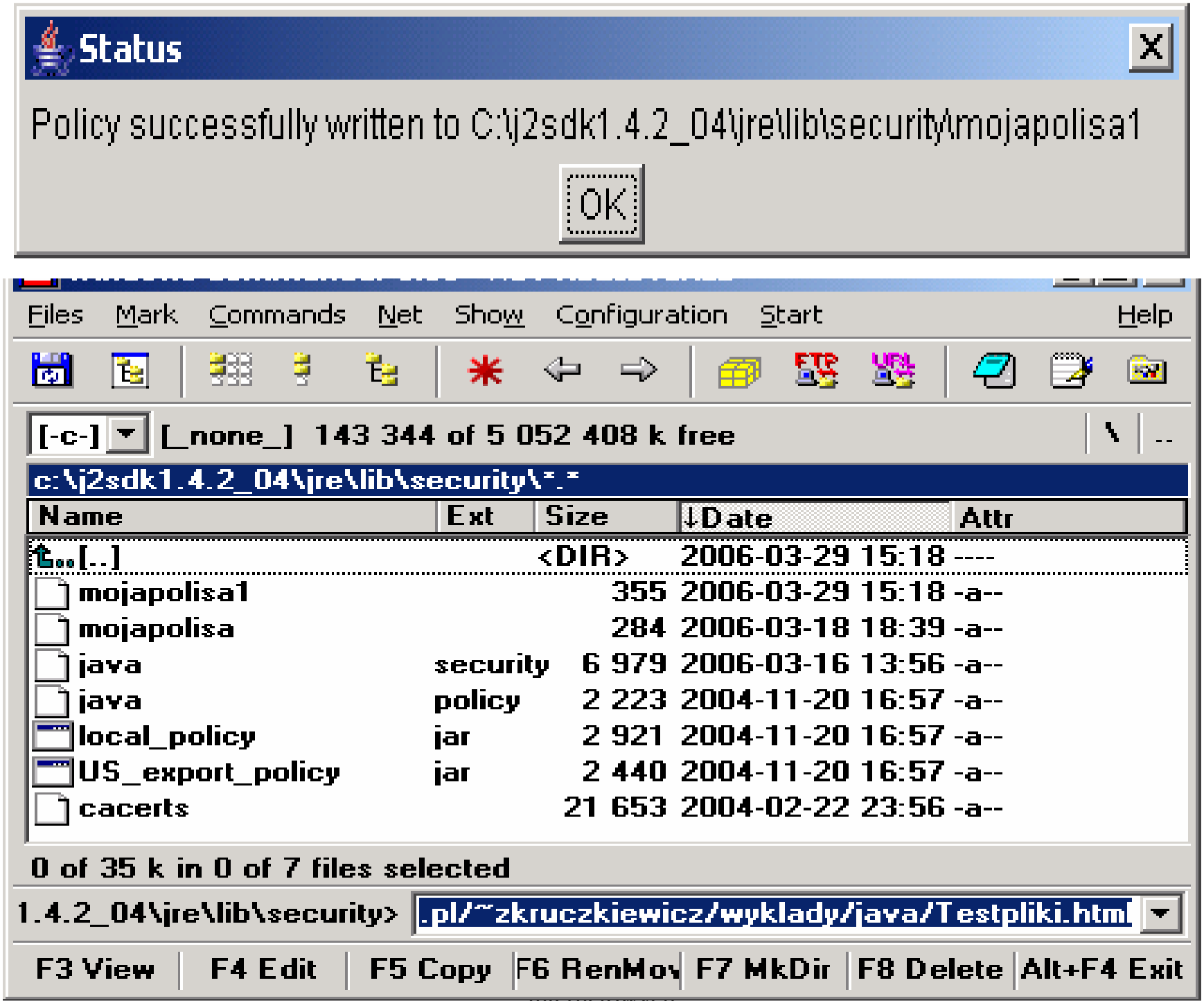

Policy Entry
CodeBase:
http://sprocket.ict.pwr.wroc.pl/~zkruczkiewicz/wyklady/java/*
SignedBy:
Jan
Add Principal
Edit Principal
Remove Principal
Principals:
Add Permission
Edit Permission
Remove Permission
permission java.io.FilePermission "D:\\p2\\Testplik", "write";
permission java.io.FilePermission "D:\\p2\\Testplik", "read";
Done
Cancel

Po zakończeniu wprowadzania danych uwierzytelniających

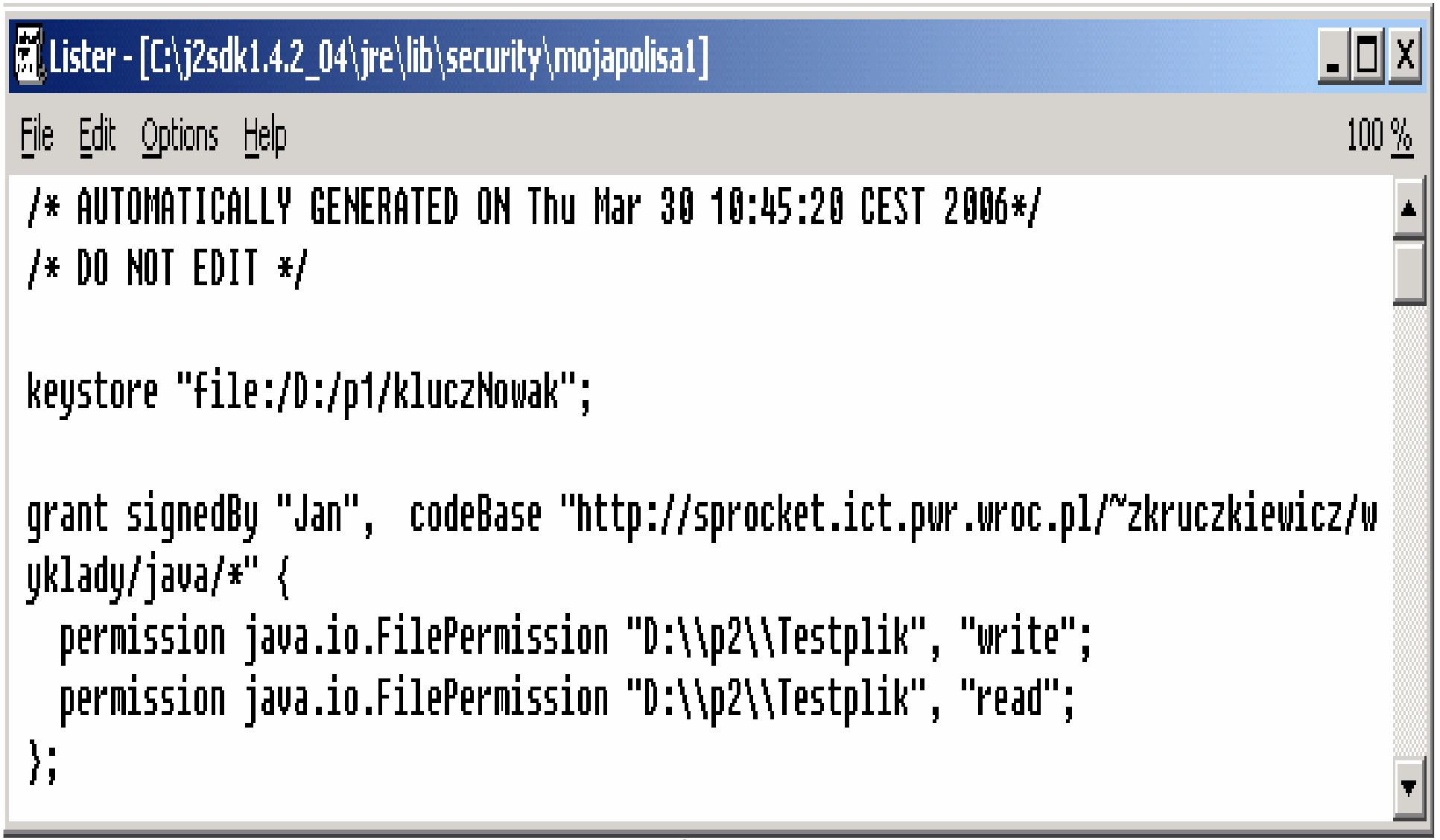


Zapis pliku uwierzytelniającego mojapolisa1 za pomocą opcji Save As z z menu File

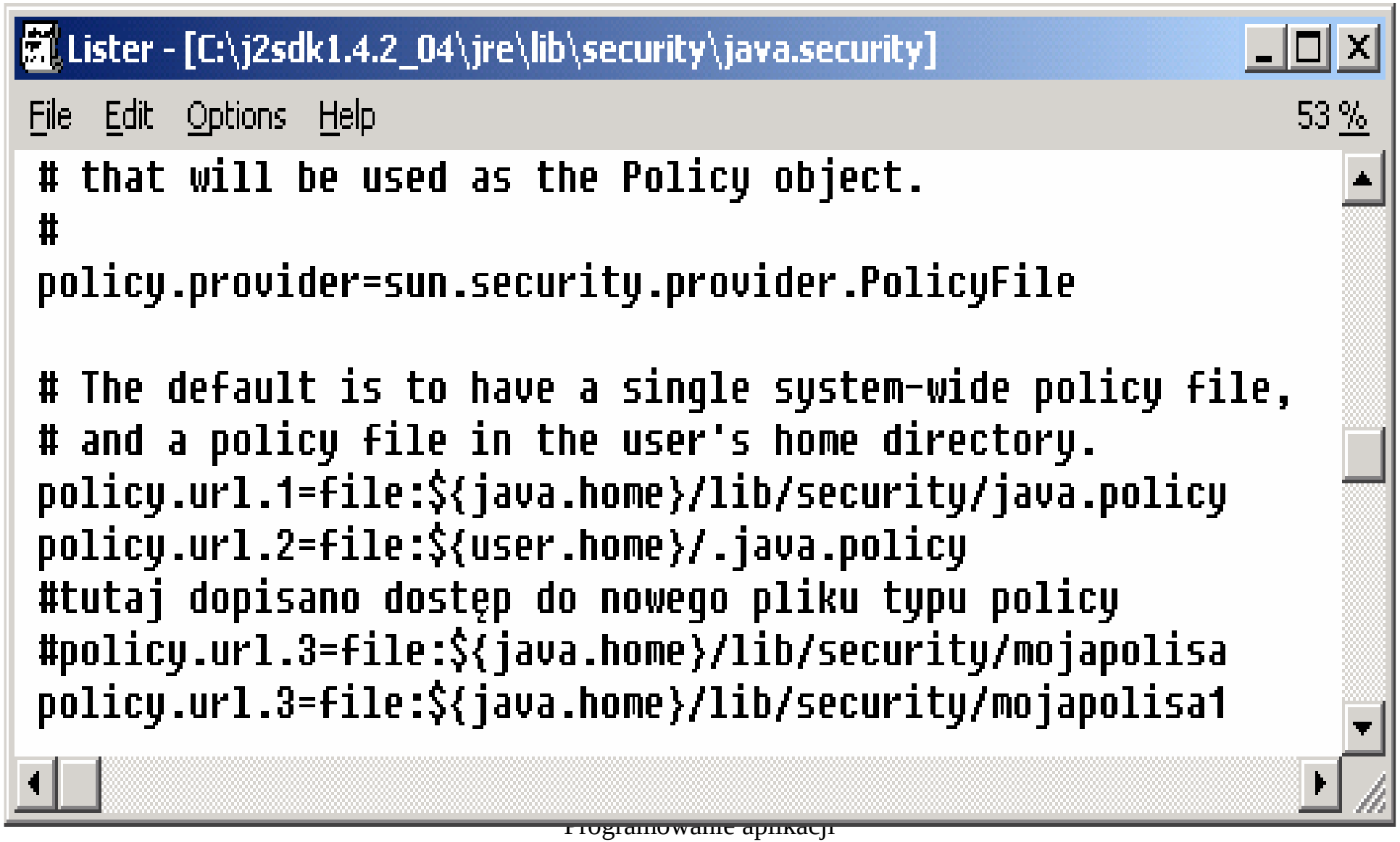



Status
Policy successfully written to C:\j2sdk1.4.2_04\jre\lib\security\mojapolisa1
OK
Files Mark Commands Net Show Configuration Start Help
[-c-] [_none_] 143 344 of 5 052 408 k free
c:\j2sdk1.4.2_04\jre\lib\security*.*
Name Ext Size ↓Date Attr
[..] <DIR> 2006-03-29 15:18 ----
mojapolisa1 355 2006-03-29 15:18 -a--
mojapolisa 284 2006-03-18 18:39 -a--
java security 6 979 2006-03-16 13:56 -a--
java policy 2 223 2004-11-20 16:57 -a--
local_policy jar 2 921 2004-11-20 16:57 -a--
US_export_policy jar 2 440 2004-11-20 16:57 -a--
cacerts 21 653 2004-02-22 23:56 -a--
0 of 35 k in 0 of 7 files selected
1.4.2_04\jre\lib\security> .pl/~zkruczkiewicz/wyklady/java/Testpliki.html
F3 View F4 Edit F5 Copy F6 RenMov F7 MkDir F8 Delete Alt+F4 Exit

Zawartość pliku uwierzytelniającego kod apletu po stronie użytkownika



Zawartość pliku ***java.security*** określająca uprawnienia apletu po stronie użytkownika za pośrednictwem pliku ***mojapolisa1***



4. Użytkownik testuje skutki konfigurowania zakresu uprawnień nadanych apletowi - aplet <u>może</u> zapisać i odczytać pliki na komputerze użytkownika