TDDC03 Projects, Spring 2005

Improved Policytool for Java Permission Management

David Krzystek
Irene Anggreeni

Supervisor: Almut Herzog

Improved Policytool for Java Permission Management

Irene Anggreeni
Dept. of Computer and Information Science
Linkopings universitet
Linkoping, Sweden
irean824@student.liu.se

David Krzystek
Dept. of Computer and Information Science
Linkopings universitet
Linkoping, Sweden
davkr899@student.liu.se

Abstract

Java platform allows users to specify permissions for code from various sources, which are represented by a Policy object. The Policy implementation is configured by a static ASCII policy files. A policy file is basically a simple text file, which can be composed via any text editor or via graphical policytool utility, that comes with Sun's Java. We addressed the question of improving policytool. We started a survey of the tool to make a list of problematic domains, especially regarding usage by novice users. We found some major drawbacks which are the graphical user interface, the missing help and the integration with other relevant tools. We put effort to solve the mentioned issues by redesigning graphical user interface, implementing a new tool based on the new design, and adding more helper functionality. Our suggestion for future works on the tool is to put more effort to increase the usability for novice users, not merely features.

1. Introduction

Concerning the security aspect, Java provides a mechanism to control permissions for application execution. This is done by a set of rules specified within a policy file, preventing or granting certain functionality. These permissions are represented during runtime by a Policy object. A policy file is basically a simple text file, which can be composed via any text editor or via graphical *policytool* utility, that comes with Sun's Java. This utility is intended especially for beginners or novice users with lack of knowledge of the syntax of the policy file. Meanwhile, experienced users are more efficient with editing such files directly from editor.
Even though *policytool* is intended for novice users, it might be still difficult to use for some people, especially regarding the usability and integration with related tools such as *keytool* and *jarsigner*. The related tools still require knowledge from the user despite the fact that he or she is not experienced. In case that the user demands to perform some complementary task using the last mentioned tools, it is necessary to do it separately on the command line. The user interface is not intuitive which often leads the users to confusion. Furthermore, we experienced that the error messages are not informative for such users.
Looking at drawbacks of that tool, there seems to be opportunities to make it easier to use, which we have done in this project.

2. Evaluation

As mentioned in introduction, we were dealing with two problematic parts, usability and integration with relevant tools. Therefore we focus on issues related to both parts.

2.1. Usability

2.1.1 GUI/layout

Before
A major drawback of the current *policytool* is the graphical user interface. We encountered several problems such as the layout not being properly displayed. We, as users, regard the interface as sometimes too tedious to work on one particular task. For example, when creating a permission entry, the user has to face three dialog windows at the same time, as depicted in figure 1.

After
We improved the design of GUI so that important functionality is accessible just by one or two clicks from the main screen. Another improvement is based on data representation as a tree structure which is only used for viewing and performing direct operations. See figure 2.

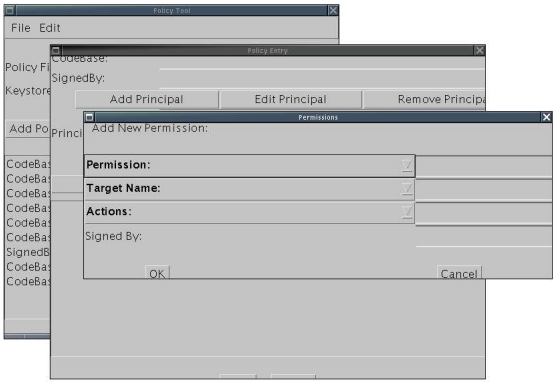

Figure 1. Disorganised layout while creating a new permission entry

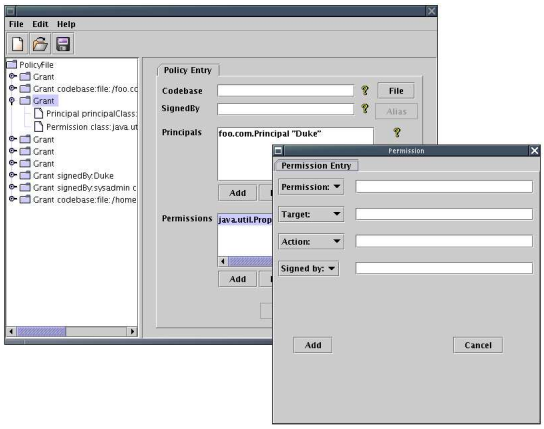

Figure 2. Use of tabbed panel and tree structure

2.1.2 Assistance components

Before
Most of the time, there is only one mechanism available to fill in values, which is through typing. Users might be confused because the lack of assistance forces users to remember a lot of information. Such information is available aliases or precise file location. Not to mention, the file location must be in URL format to adapt with the requirement of policy file syntax, which is demonstrated in figure 3.

After
Entered values are sometimes expected in a specific format, for instance the specification of *codebase* requires user to write *file:/* prefix. This implies usage of GUI components such as File Browser or drop down menu displaying list of aliases stored in *keystore*. These countermeasures together with the input validation reduce the error rate. Our implementation is shown in figure 4.

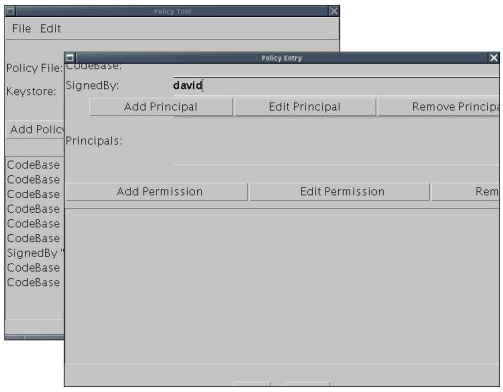

Figure 3. Values must by typed in

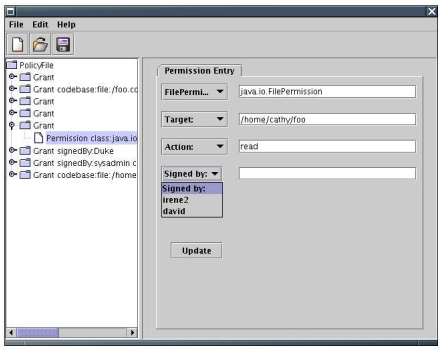

Figure 4. User can choose alias conveniently from drop down

2.1.3 Easy access to help

Before
Help is missing completely from the application. The only thing a user can do when he is stuck with a problem is to browse online documentation provided by Sun, or other resources. There is no integration between the available help resource with the current *policytool*.

After
We have integrated available documentation in our tool, together with question mark icons and tool tips. After clicking on a question mark icon, help documentation is provided as shown in figure 5 and 6.

2.1.4 Terminology

We analyzed problematic domains from different perspectives. One of our ideas was renaming terms, which could provide easier understanding in case of novice users. On the

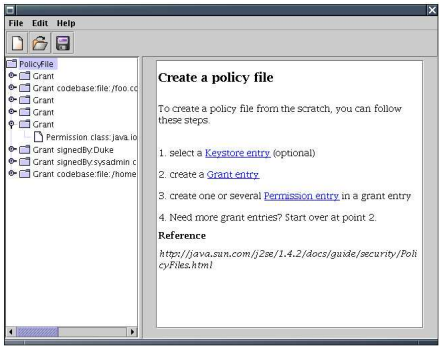

Figure 5. Integrated concise help

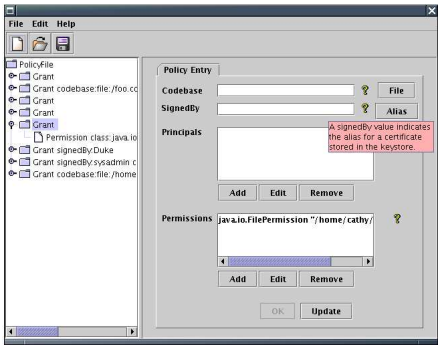

Figure 6. Short multi-line tool tips

other hand it could cause confusion of experienced users. Thus this was not implemented.

2.1.5 Integration

Before
For handling signed class files, users have to cooperate with other tools which are unfortunately only available from the command line and thus only outside the GUI of *policytool*. With these tools users might need to do particular operations on available key pairs and certificates within *keystore* as well as signing jar files.
After

- **keytool**
  We implemented fundamental operations for supporting the improved *policytool*. This includes browsing, creating, exporting and importing, and removing certificates. It is not our intention to create a GUI *keytool* and the same is valid in the case of *jarsigner*. This is shown in figure 7.

- **jarsigner**
  The situation of *jarsigner* integration does not differ from *keytool* integration. Simple GUI regarding *jarsigner* is displayed in figure [8].

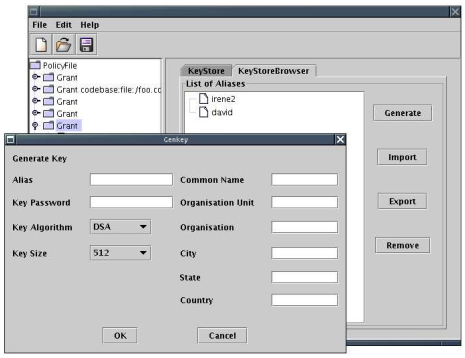

Figure 7. Keystore integration

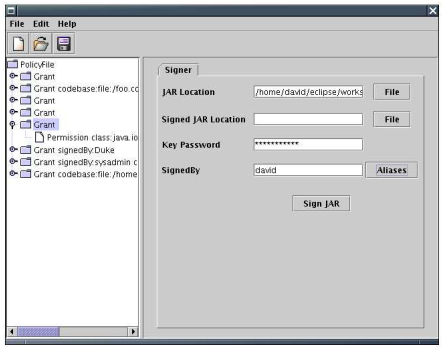

Figure 8. Jarsigner integration

2.2. Conclusions

We have tested Java *policytool* to make acquaintance of existing problems and proposed solutions to improve it. We have redesigned the graphical user interface to make the tool more usable for novice users. To be more complete, we added features to get necessary help and also assistance during filling in values required in the policy file. We have also implemented simple, but not extensive integrations with other related tools such as *keytool* and *jarsigner*. An idea for further enhancement is to provide wizard regarding that the aim of the tool is actually to help novice users. We have improved the *policytool* in such a way, that it is sufficient to accommodate truly novice users. Nevertheless, there are always opportunities for improvement from others to make it even better.

References

[1] Sun Microsystems, Inc. Default policy implementation and policy file syntax. http://java.sun.com/j2se/1.4.2/docs/guide/security/PolicyFiles.html (visited 2-May-2005).

[2] Sun Microsystems, Inc. Jarsigner - jar signing and verification tool. http://java.sun.com/j2se/1.4.2/docs/tooldocs/solaris/jarsigner.html (visited 2-May-2005).

[3] Sun Microsystems, Inc. Keytool - key and certificate management tool. http://java.sun.com/j2se/1.4.2/docs/tooldocs/solaris/keytool.html (visited 2-May-2005).

[4] Sun Microsystems, Inc. Permissions in the java2 sdk. http://java.sun.com/j2se/1.4.2/docs/guide/security/permissions.html (visited 2-May-2005).

[5] Sun Microsystems, Inc. Policy tool - policy file creation and management tool. http://java.sun.com/j2se/1.4.2/docs/tooldocs/solaris/policytool.html (visited 2-May-2005).