

POLICY SPECIFICATION AND TRANSLATION USING THE USAGE CONTROL SYSTEM

Presented below is a short tutorial on how to run our usage control system to specify and translate policies. The usage control policy editor provides graphical tools to specify policies and translation includes future to past translation, action refinement and Event-Condition-Action (ECA) rules configuration. In addition, there is a pseudo implementation of policy instantiation and an implementation for deployment which makes for exploring these 2 stages.

We shall consider one use case of the system. This will involve using web-based social network named TUMFormatics (designed using an open-source social network system named [Elgg](#)) and then indicating a picture for which we wish to exercise control over.

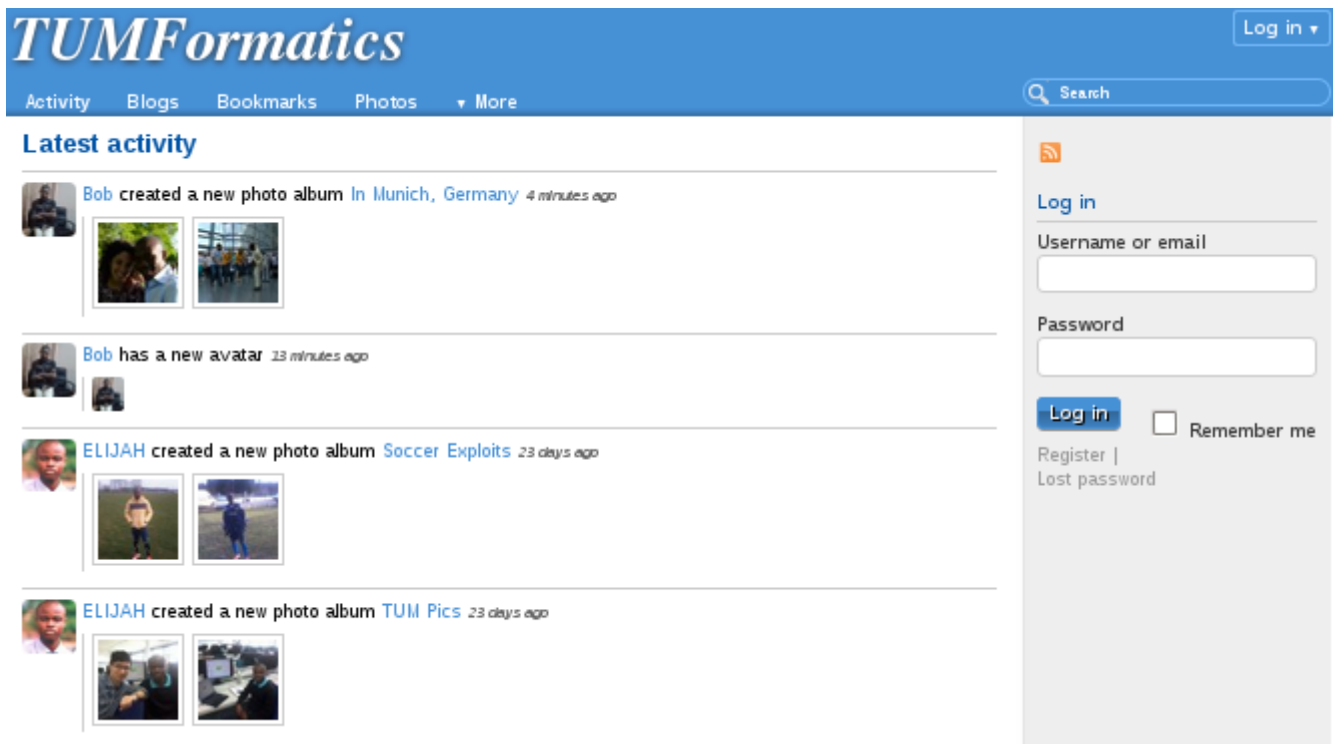
The system is made available and already setup in a Debian Linux (version 3.2.0) virtual machine and should be run in VMWare Player version 3.1.6-744570 or later.

Upon loading the virtual machine, choose the account presented as **Omuyelijah** and type password as **elijah**. You will be taken to the desktop environment where you can continue with the steps below.

1. Start a running instance of Mozilla Firefox web-browser from its short-cut link on the desktop.
2. In the address bar, type **localhost/elgg** to load the social network site. There is a sample user account which we can use for now. Log in with

Username: userBob

Password: passwordBob



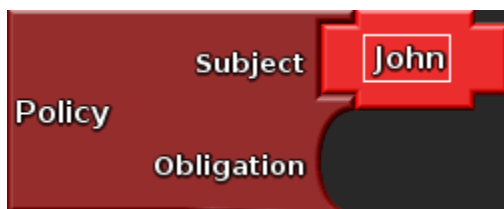
3. The sample user account has a profile picture as well as pictures organized into albums. Select a picture of your choice. Depending on the kind of picture, we can have slightly different commands. Right-click on your choice picture. If it is a profile picture, you will find the menu item ***create policy for profile picture***, else if it is an album picture two items show up, indicating the possibility of ***creating policies for the album picture or the entire album***. Creating policies for an album will create policies for every picture in the album concerned. Whichever is the case, click the appropriate menu item presented. This will launch the usage control policy editor.
4. We shall create a sample policy to state “picture must not be copied by John”. The editor provides a nice environment with graphical tools which support drag and drop. On the left side of the editor, there are block drawers each with a name, a distinct colour and a collection of blocks within it.



Click ***New Policy*** and from the fly-out, drag a ***Policy block*** on to the workspace.



Click on the red block in the recipient text and change it to John. Click and drag out the (yellow) true block to the trash can on the bottom right of your screen. We will prepare an new obligation.



5. Our policy will be in the form ***always(not(copy(picture)))***. We shall now bring in the needed blocks to our graphical policy. From the block drawer (referenced in step 4), click the

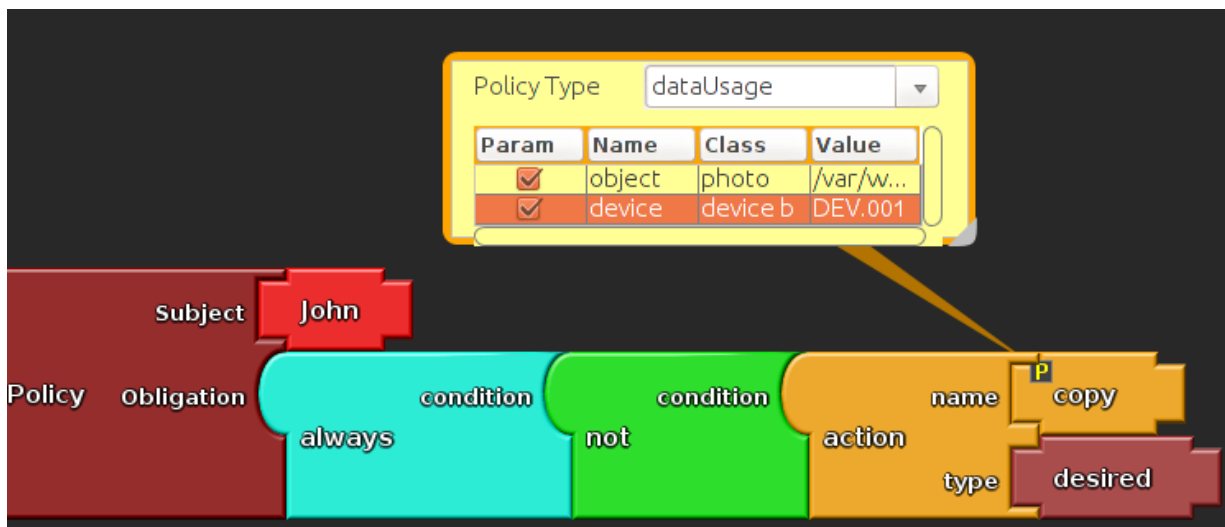
Temporal logic block and from the flyout, drag the **always** block into the obligation space.



6. In a similar manner, from the block drawer, click the **Propositional Logic** block and drag the **not** block to our diagram.

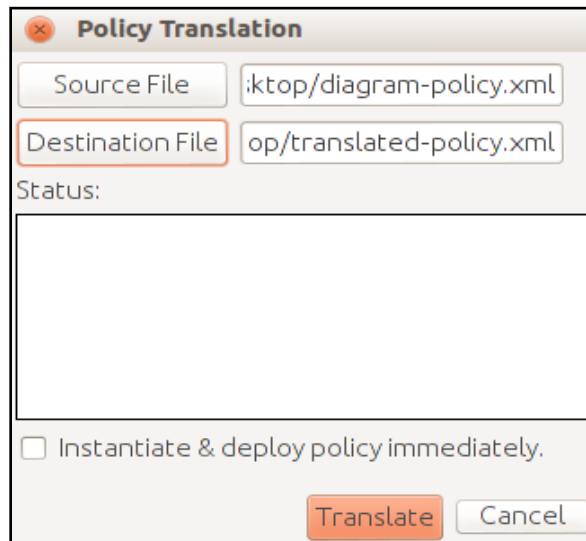


7. We are left with one more block: the **action** block. Drag this out from the **Action** block in the block drawer. In the name block, click the drop-down arrow and select **copy**. A parameter call-out appears. Let's have the policy type as **dataUsage**. A little look will reveal that in the table, we find the path to the image on the file system in the **value** column. To fill in some device values in the table, first check the box in the **param** column.



We are now done with policy specification. Let's save the policy from the file menu before we proceed, i.e. File > Save As ...

8. To translate the policy, we click Policy menu > Translate ... This will launch the translation dialog box, with our saved policy as **source file**. Specify the **destination file** name for the translated file. As stated earlier, we provide a sample implementation of instantiation and deployment. To try it out, click the check-box for it. Then click **Translate** button.



Policy Translation

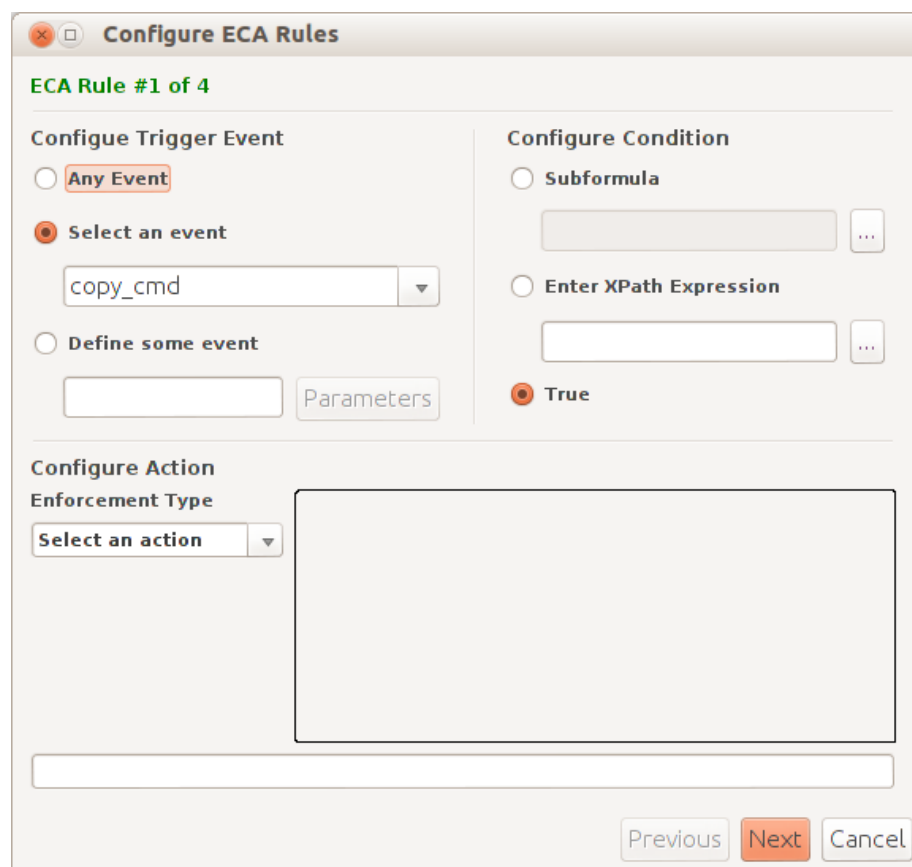
Source File:

Destination File:

Status:

☐ Instantiate & deploy policy immediately.

9. As the last stage of translation is ECA rules configuration, you will get a dialog window like below.



Configure ECA Rules

ECA Rule #1 of 4

Configure Trigger Event

☐ Any Event

☒ Select an event

▼

☐ Define some event

Configure Condition

☐ Subformula

...

☐ Enter XPath Expression

...

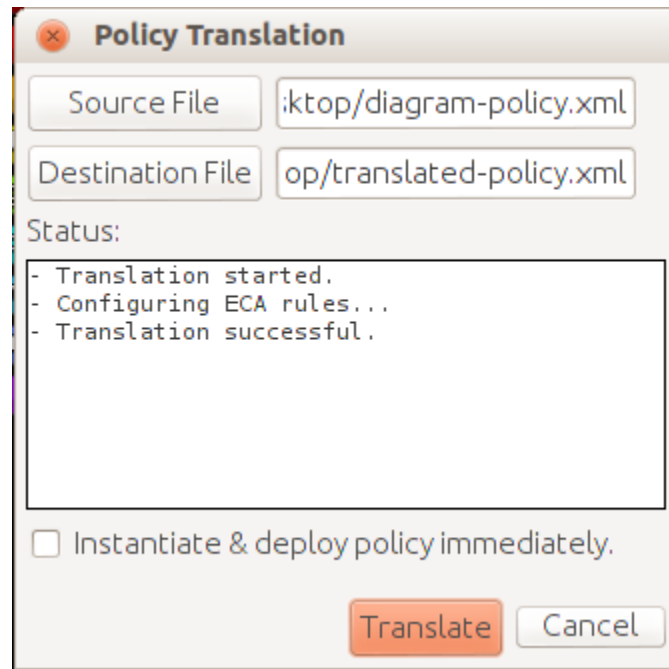
☒ True

Configure Action

Enforcement Type

▼

You can configure the ECA rules moving forward and backward with the **Next** and **Previous** buttons at the bottom of the window. In the end, click **Finish**. This will create a translated policy ready for instantiation and deployment. Messages during the translation (and if selected, instantiation and deployment) will appear on the report window output.



In summary, our implementation of usage control system handles policy specification and translation.