TRIPLEX: USER MANUAL

TRIPLEX is a tool that allows to specify protocols in a simple syntax and to analyse privacy properties. For technical details on the implementation of this tool, please refer to [1], [2].

1. Running TRIPLEX. To run TRIPLEX double click the file "TRIPLEX.jar". This will launch the application, which consists of the following sequence of steps:

1. Insertion of one or more abstract protocols (see **2. TRIPLEX: Inserting one or more abstract protocols.**).
2. Insertion of a scenario (see **3. TRIPLEX: Inserting a scenario.**).
3. Providing the initial knowledge for the given scenario (see **4. TRIPLEX: Initial knowledge.**).
4. Instantiating the abstract protocol using the given scenario (see **5. TRIPLEX: Protocol instantiation.**).
5. Studying privacy properties (see **6. TRIPLEX: Protocol analysis.**).
6. Drawing the coalition graph (see **7. TRIPLEX: Coalition graph.**).

2. TRIPLEX: Inserting one or more abstract protocols. The file "TRIPLEX.jar" opens a new window where it is possible to insert one or more abstract protocols.

There are four types of abstract items that one may use when inserting a protocol:

1. Role: a role is used to describe which actions/items belong to the same agent when a session of the abstract protocol is run between agents;
2. Identifier: an identifier is a name that uniquely identifies an agent; an agent may have multiple identifiers; this type of items includes keys;
3. Data: data that belong to a specific agent, e.g. age;
4. Global data: data that do not belong to any specific agent, e.g. nonces.

To add a protocol, click the button "Add protocol" (Figure 1.). This will open a popup (Figure 2.) where it is possible to type and save (using the button Save) an abstract protocol or to insert a protocol from a file with extension ".protocol" (using the button Load).

A protocol must have a name (different protocols must have different names) and its messages be specified in the syntax given below. Messages can be added using the button "Add message" or deleted by clicking the corresponding button "X" (Figure 2.). A message consists of three boxes:

1. A sender box: type the abstract identifier used by the sender of the message including its corresponding role (identifier{role}); e.g. "ip{s}" indicates that the role "s" uses an identifier "ip{s}" to send the message;
2. A receiver box: type the abstract identifier used by the receiver of the message including its corresponding role (identifier{role});
3. A message box: type the message that the sender wants to send to the receiver using the syntax below.

The syntax for terms is the following:

E ::= id{r,ID} | data{r,D} | globalData{G}

where id{r,ID} is an abstract identifier "id" that identifies a role "r", data{r,D} is abstract data regarding a role "r" and globalData{G} represents some information that does not belong to any specific agent. The tags ID, D and G denote that the terms are identifiers/keys, data and global data respectively.

The syntax for the primitives is:

- [E1,...,En]: for a list of n elements Ei;

- pk(E): for the public key matching the private key E;
- es(E1,E2): for symmetric encryption of a message E2 with a key E1;
- ea(pk(E1),E2): for asymmetric encryption of a message E2 with a public key pk(E1);
- el(pk(E1),E2,E3): for asymmetric encryption of a message E2 and label E3 with a public key pk(E1);
- s(E1,E2): for the digital signature of a message E2 with a private key E1;
- hash(E): for the hash of a message E;
- aka(E1,E2,E3,E4): for the derived key from authenticated key agreement (AKA) with (SK, randomness) pairs (E1;E2) and (E3;E4), where E1 and E3 are private keys;
- cred(E1,E2,E3,E4,E5): anonymous credential with user identifier E1, issuer private key E2, attributes E3, and randomness E4 (from the user) and E5 (from the credential issuer).

Figure 2. shows an example of insertion of a protocol named “Distribute”. It consists of a single message from “ci” (a credential issuer) using the abstract identifier “ip{ci}” to “r” (a researcher) using the identifier “ip{r}”. The message is [hash([hash(bsn{u,ID}),dom(r,ID)]),d1{u,D},d2{u,D}], therefore it represents the hash of (1) the hash of a pair of identifiers “bsn” and “dom” that identifies an agent “u” and a researcher “r” respectively, and (2) two data items regarding “u”, hence the agent identified by “bsn”.

To insert the protocol, click the button Continue. The popup will close and the protocol will be added to the main screen. If you want to delete or edit a protocol, click the corresponding buttons “X” or “Edit protocol”. When all the abstract protocols have been inserted correctly, click the button “Next”.

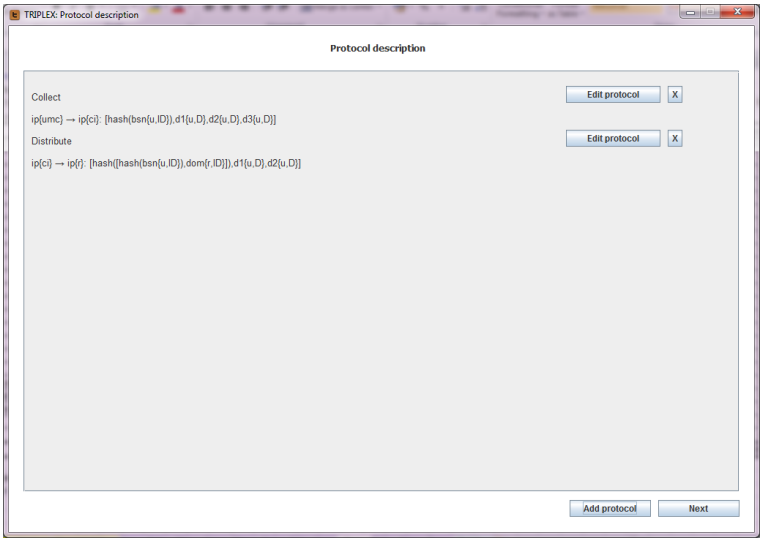

Figure 1. TRIPLEX: Protocol description

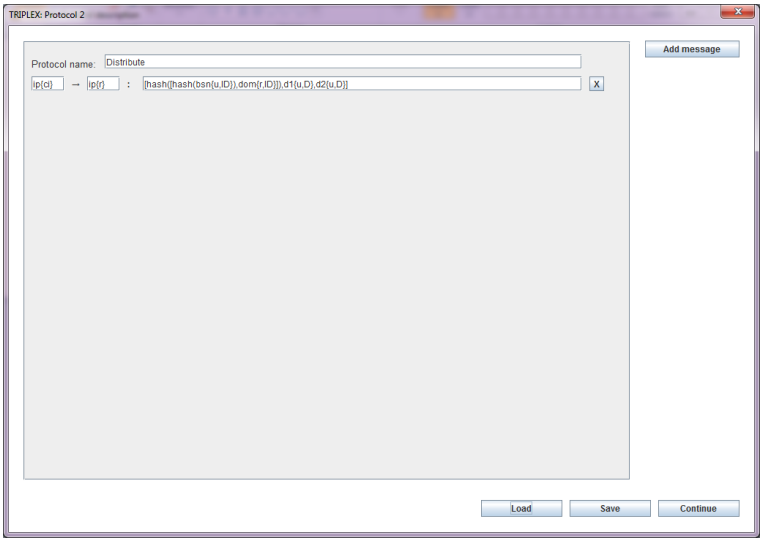

Figure 2. TRIPLEX: Protocol description – Add protocol

3. TRIPLEX: Inserting a scenario. After clicking the button “Next”, the screen for the protocol description will close and a new screen will appear to insert a scenario (Figure 3.), i.e. a set of concrete items. From this screen on, it is always possible to switch back to a previous screen by clicking the button “Back”.

In the screen named “Scenario” it is possible (1) to insert a new scenario by filling in the popups launched by the buttons “Entities”, “Identifiers”, “Data”, “Global data” and “Keys”, (2) to load a scenario/scenario + initial knowledge/scenario + initial knowledge + instantiation file with extension “.scenario”/“.knowledge”/“.instantiation” by clicking the button “Load” or (3) to save a scenario by clicking the button “Save” (in this screen you can save a “.scenario” file).

To confirm the information filled in the popups (case 1) click the button “Ok”. In the popup use the buttons “Add entity/identifier/data/global data/key” and “X” to insert or delete entries. If you want to edit the information in a popup after closing it, just click again the corresponding button.

While inserting the scenario or after loading a scenario, a table with a summary of the items that have been inserted is displayed in the main screen.

Note that the items in the scenario must not contain the character “-“ and that different items must have different names.

Figure 4. shows an example of how to fill in the entries in the popups. The first button, “Entities”, is to insert the names of the physical agents in the scenario (e.g. Alice). The popup for “Identifiers” (as well as the one for “Keys” not displayed here) requires a name for the identifier (key) and the selection of an entity, meaning that the identifier represents (the key belongs to) the specified entity (e.g. AliceID, Alice). Similarly, the popup “Data” requires a name for each data entry (which information is represented), the real content of the entry and the entity the data is related to (e.g. age_of_Alice, 18, Alice). Finally, the popup for “Global data” is equivalent to the one for “Data”, but do not require the selection of an entity for each entry, as it represents information that is not related to any specific entity.

Note that the scenario may contain more information than needed for the protocol instantiation and that it does not need to include all the types of items (e.g. global data may be left empty).

When the scenario insertion is completed you can proceed to the next screen by clicking the button “Next”.

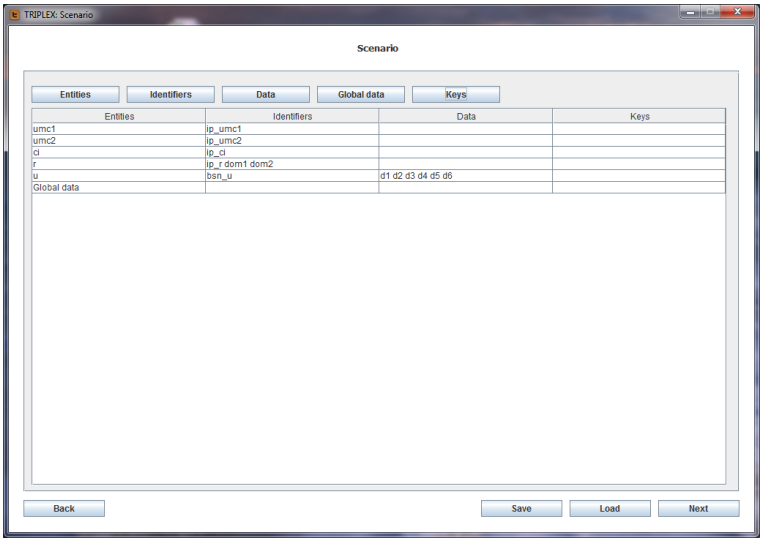

Figure 3. TRIPLEX: Scenario

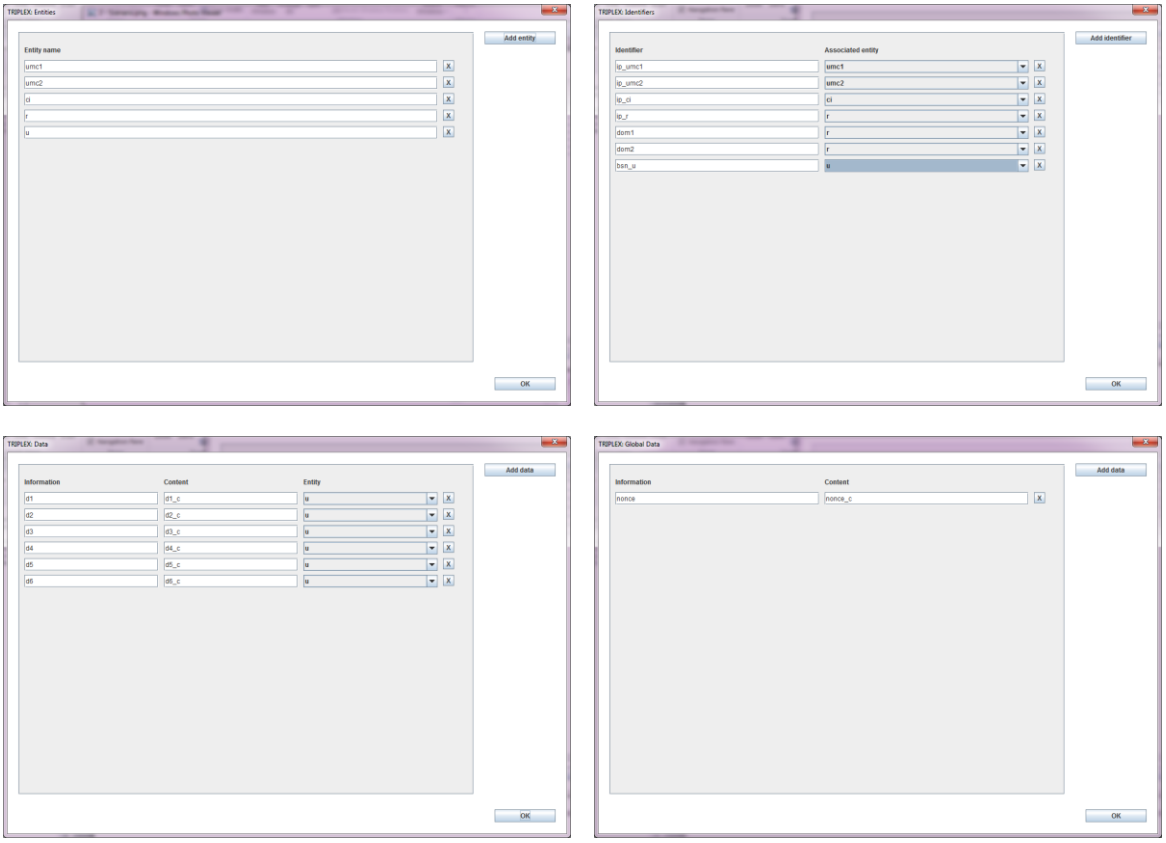

Figure 4. TRIPLEX: Scenario – Insert values

4. TRIPLEX: Initial knowledge. This screen consists of a table to insert the initial knowledge (Figure 5.), i.e. the knowledge that each entity initially has about the identifiers, data, global data and keys. The header line starts with an entry “Global” whose corresponding column will possibly contain the profiles of the globally known items. The other headers are the list of the entities in the scenario. The first column contains the list of all items in the scenario. In this table it is possible to specify whether an entity knows a specific item and the corresponding profile. For simplicity, you may use numbers: the same number in multiple entries means that the entities involved knows the corresponding items to be linked, i.e. to belong to the same user.

To save the scenario together with the initial knowledge inserted in the table in a file with extension ".knowledge", click "Save".

To instantiate the abstract protocol(s) click the button "Next". This will open a popup to save a file "TRIPLEX_state0.pl" containing the initial knowledge. Select a valid path, but do not change the name of the file.

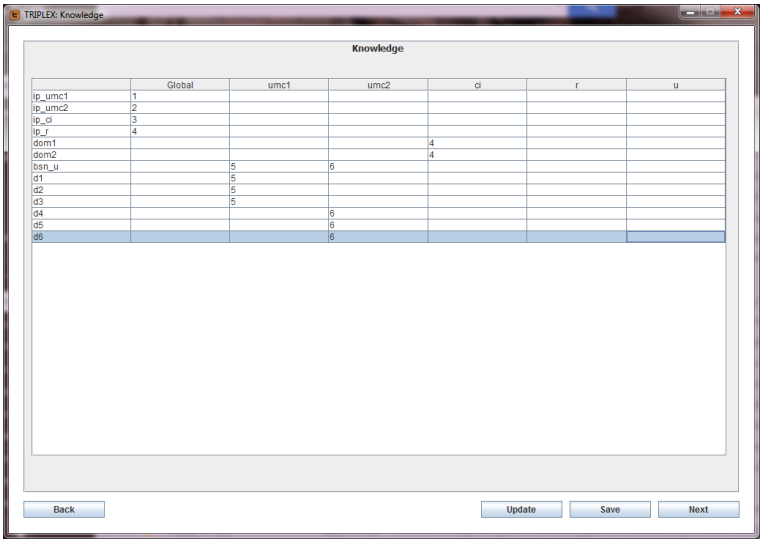

Figure 5. TRIPLEX: Initial knowledge

5. TRIPLEX: Protocol instantiation. In this screen the given scenario is used to instantiate the abstract protocol(s). First of all, for each protocol you need to specify how many sessions you want to run. For instance, in Figure 6. we request two sessions for each protocol. Then, click the button "Set roles". This will generate a list of roles (on the left), grouped per protocol per session. On the right side a list of boxes will appear and you need to select the entity from the scenario that you want to assign to each role. These boxes are automatically initialised to the first item of the list. Once all the roles are correctly instantiated, click the button "Set identifiers", which similarly will display all the abstract identifiers on the left (including keys) and boxes to select the instantiation from the scenario on the right. When the abstract identifiers are instantiated, repeat these steps after clicking first "Set data" and then "Set global data". When this phase is completed, click "Analysis" or "Graph". Note that you can use the button "Save" to save a file with extension ".instantiation" containing the scenario, initial knowledge and instantiation inserted so far.

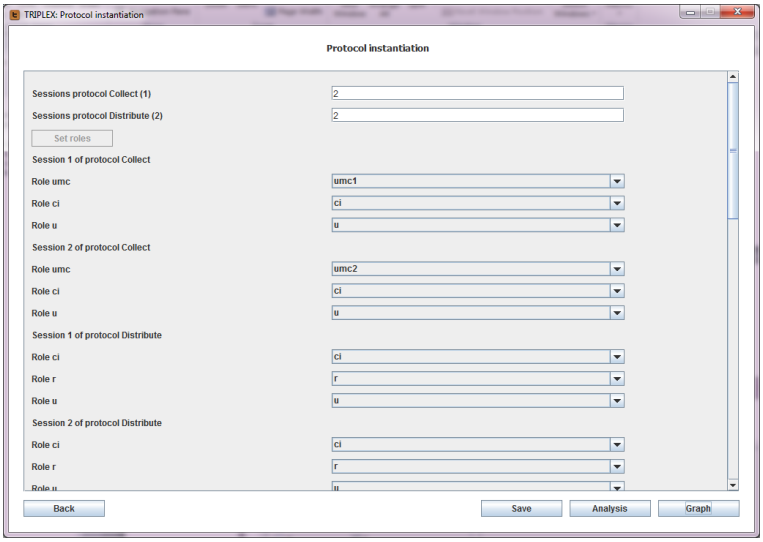

Figure 6. TRIPLEX: Instantiation

6. TRIPLEX: Protocol analysis. After clicking the button “Analysis” the screen in Figure 7. will appear. You can choose a coalition of actors and calculate the list of linked items by clicking the button “Linked items”. You can also analyse properties by clicking the button “Add property” that opens a popup to insert a property (Figure 8.). In the popup, insert a title for the property. Then, select “Detect” if you want to verify whether the given coalition can detect an item sent in a specific session (each session is called “sessionN-M”, where N+1 is the protocol it corresponds to – insertion order – and M+1 is the session number, e.g. “session0-1” corresponds to the second execution of the first protocol). Instead, select “Link” if you want to verify if the coalition is able to tell whether two sessions contain information about the same entity. Check the box “NOT” if you want to verify “non-detectability” or “non-linkability”. If you want to verify whether multiple properties hold, click the button “AND” and insert another property. If you want to verify whether at least one of a list of properties holds, click the button “OR” and insert another property. Note that the properties may only be specified in the following form: (A1 AND B1 AND…) OR (A2 AND B2 AND…) OR…, where Ai and Bi are “Detect” or “Link” properties. When done, click on the button “OK” to display the complete property in the text area and/or click “Continue” to go back to the main screen (Figure 7.) and analyse the property for the given coalition.

Note that a button with the name of the property will appear. It will be green if the property holds and red if it does not. You can display the property by clicking the button (Figure 9.) or delete it by clicking the corresponding “X” button. If the coalition is changed, the properties are automatically updated.

Click the button “Graph” if you need to visualise the coalition graph screen.

In Figure 8. the property “BSN not for research purposes” is being inserted. It tests whether the given coalition cannot detect the bsn of the entity “u” considering all sessions.

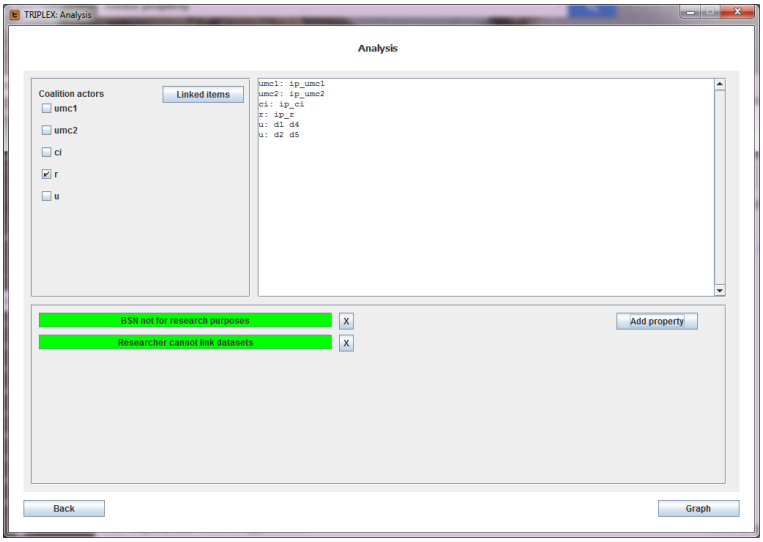

Figure 7. TRIPLEX: Analysis

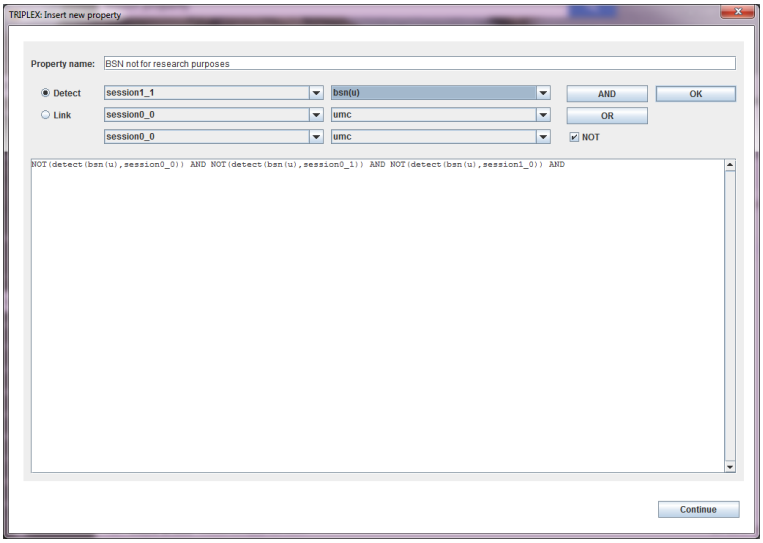

Figure 8. TRIPLEX: Analysis – Defining a property

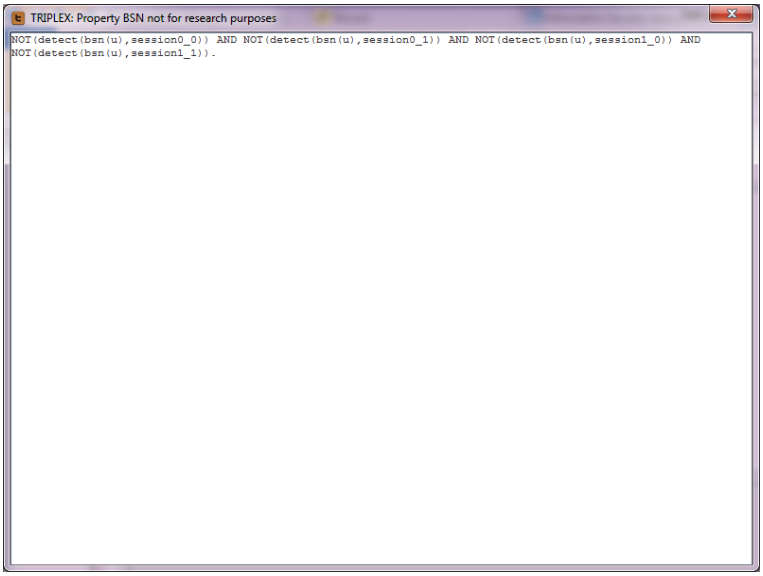

Figure 9. TRIPLEX: Analysis – Visualising a property

7. TRIPLEX: The coalition graph. The coalition graph screen allows to select a data subject and a coalition from the list of entities. After this step, click the button “Graph” to display the corresponding graph. Click the button “Analysis” if you want to move to the analysis screen.

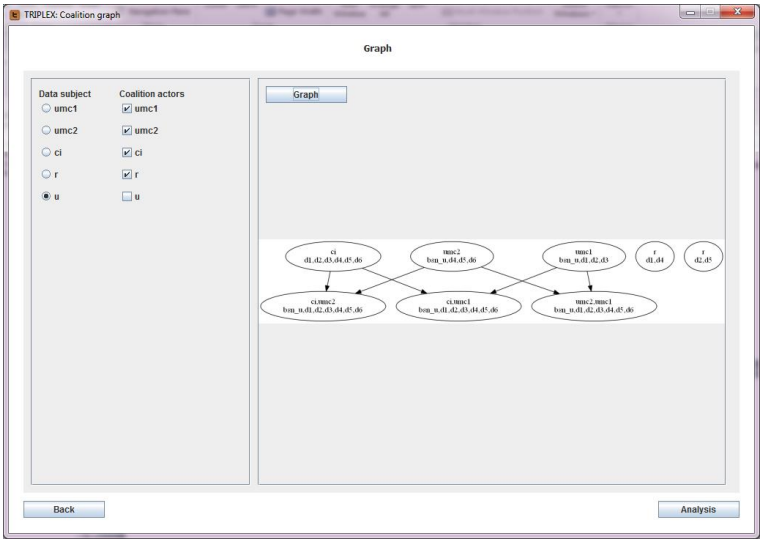

Figure 10. TRIPLEX: Coalition graph

[1] Veeningen, M.G., Weger, B.M.M. de & Zannone, N. (2013). Formal modelling of (de)pseudonymisation : a case study in health care privacy. In A. Jøsang, P. Samarati & M. Petrocchi (Eds.), Conference Paper : Security and Trust Management (8th International Workshop, STM 2012, Pisa, Italy, September 13-14, 2012. Revised selected papers), (Lecture Notes in Computer Science, 7783, pp. 145-160). Berlin: Springer.

[2] Veeningen, M.G., Weger, B.M.M. de & Zannone, N. (2011). Formal privacy analysis of communication protocols for identity management. In S. Jajodia & C. Mazumdar (Eds.), Conference Paper : Information Systems Security (7th International Conference, ICISS 2011, Kolkata, India, December 15-19, 2011. Proceedings), (Lecture Notes in Computer Science, 7093, pp. 235-249). Berlin: Springer.