A key distribution system

Laura Martignon
879707@stud.unive.it

Riccardo Maso
885271@stud.unive.it

Nicola Aggio
880008@stud.unive.it

Content

1. Key Distribution Center: the protocol
2. PEPA model of the system
    2.1. PEPA specification
    2.2. Derivation graph and CTMC
    2.3. Infinitesimal generator matrix
    2.4. GBEs
    2.5. Steady state probabilities
    2.6. Utilisation
    2.7. Throughput
3. Bisimulation to aggregate the PEPA components
    3.1. Derivation graph
    3.2. Bisimulation
    3.3. Steady state probabilities - comparison
4. Analysis
    4.1. Utilisation
    4.2. Throughput

Key Distribution Center: the protocol

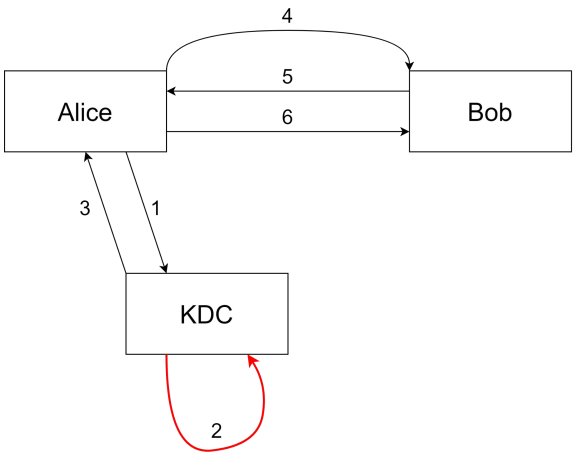
Components involved

- Two entities exchanging a message (**Alice** and **Bob**)
- **KDC** (Key distribution center): for each exchange a session key is shared (symmetric keys). KDC is a trusted third party that handles key distribution.

In the original model KDC only takes care of the distribution, but not of the keys **generation**.

Now we are going to add this operation to the protocol and see how the system model and the relative performances change.

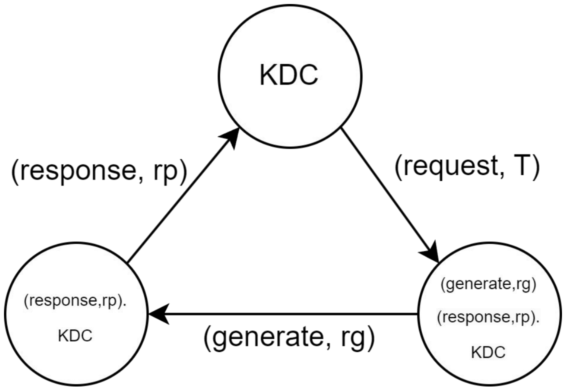
Formalization of the protocol



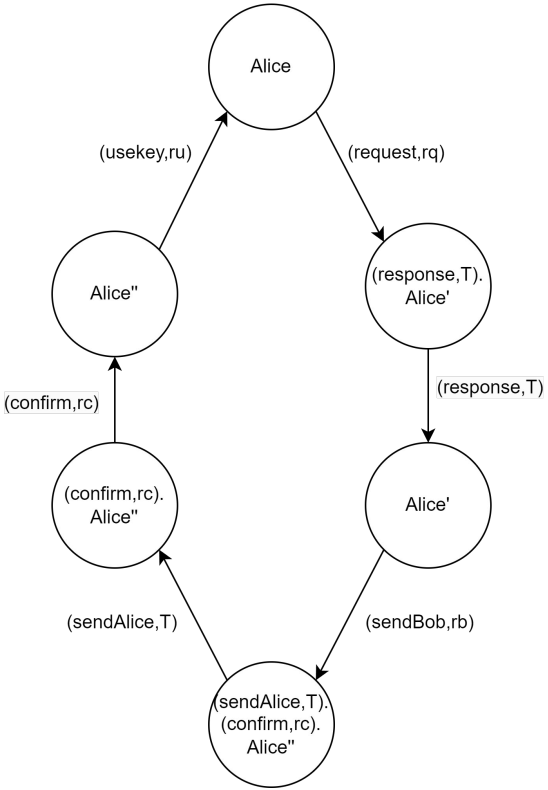
1. Alice sends request to KDC with nonce N_1
2. Generation of the key: it was not modeled in the original protocol
3. Then KDC sends the response:
   $E\{K_A\}[KS|request|N_1|E\{K_B\}[K_S|ID_A]]$
   - K_S is a session key for Alice and Bob to use
   - Only Alice can read the response of the KDC (encryption with K_A)
   - Alice cannot decrypt the part encoded with Bobs key, she can only send it on.
4. $E\{K_B\}[K_S|ID_A]$ (Alice forwards to Bob)
5. $E\{K_S\}[N_2]$ (Bob descripts his part)
6. $E\{K_S\}[f(N_2)]$ (Alice proved she has received Bob's message)

PEPA model of the system

PEPA specification of the single components



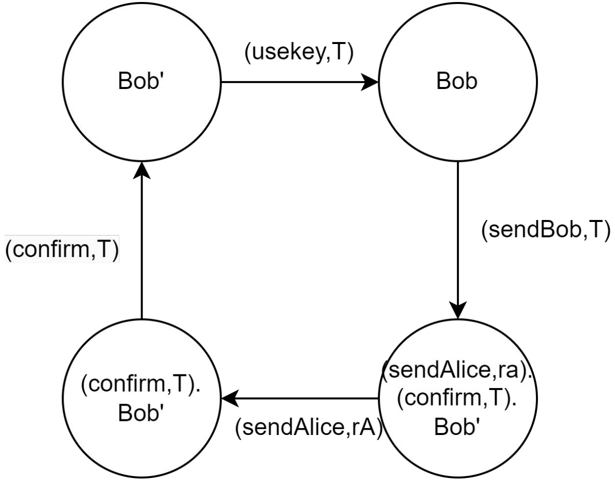
$$KDC \stackrel{def}{=} (request, \top).(generate, r_g).(response, r_p).KDC$$



$$Alice \stackrel{def}{=} (request, r_q).(response, \top).Alice'$$

$$Alice' \stackrel{def}{=} (sendBob, r_B).(sendAlice, \top).(confirm, rc).Alice''$$

$$Alice'' \stackrel{def}{=} (usekey, r_u).Alice$$



$$Bob \stackrel{def}{=} (sendBob, \top).(sendAlice, r_A).(confirm, \top).Bob'$$

$$Bob' \stackrel{def}{=} (usekey, \top).Bob$$

PEPA specification of the system

$$KDC \stackrel{def}{=} (request, \top).(generate, r_g).(response, r_p).KDC$$

$$Alice \stackrel{def}{=} (request, r_q).(response, \top).Alice'$$

$$Alice' \stackrel{def}{=} (sendBob, r_B).(sendAlice, \top).(confirm, rc).Alice''$$

$$Alice'' \stackrel{def}{=} (usekey, r_u).Alice$$

$$Bob \stackrel{def}{=} (sendBob, \top).(sendAlice, r_A).(confirm, \top).Bob'$$

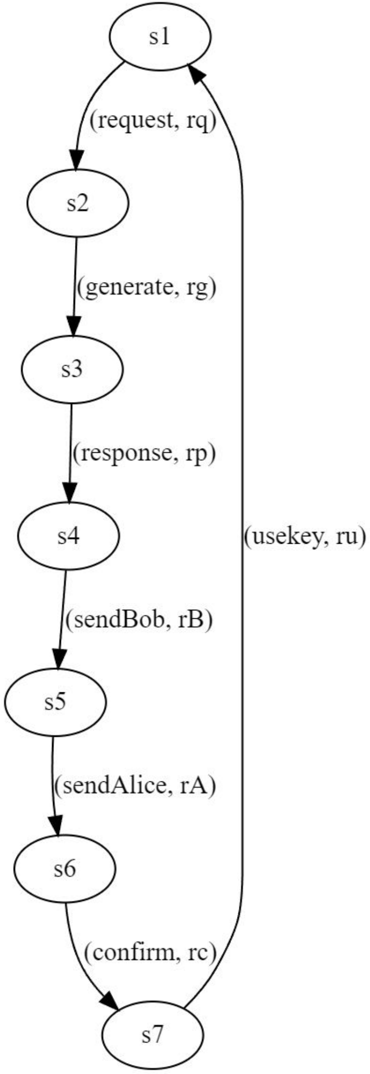
$$Bob' \stackrel{def}{=} (usekey, \top).Bob$$

$$System \stackrel{def}{=} KDC \underset{\mathcal{L}}{\bowtie} (Alice \underset{\mathcal{K}}{\bowtie} Bob \| \ldots \| Alice \underset{\mathcal{K}}{\bowtie} Bob)$$

$$\mathcal{L} = \{request, response\}$$

$$\mathcal{K} = \{sendbob, sendAlice, confirm, usekey\}$$

Derivation graph



$$s1 : KDC \underset{\mathcal{L}}{\bowtie} (Alice \underset{\mathcal{K}}{\bowtie} Bob)$$

$$s2 : (generate, r_g).(response, r_p).KDC \underset{\mathcal{L}}{\bowtie} ((response, \top).Alice' \underset{\mathcal{K}}{\bowtie} Bob)$$

$$s3 : (response, r_p).KDC \underset{\mathcal{L}}{\bowtie} ((response, \top).Alice' \underset{\mathcal{K}}{\bowtie} Bob)$$

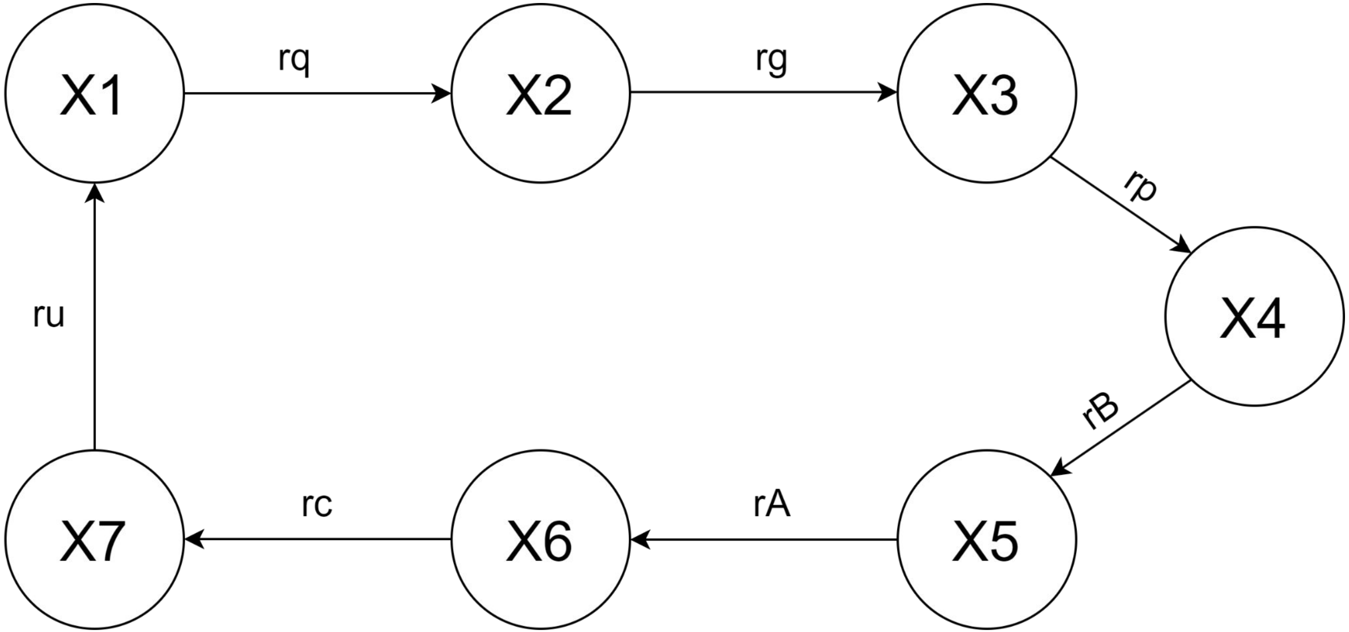
$$s4 : KDC \underset{\mathcal{L}}{\bowtie} (Alice' \underset{\mathcal{K}}{\bowtie} Bob)$$

$$s5 : KDC \underset{\mathcal{L}}{\bowtie} ((sendAlice, \top).(confirm, rc).Alice'' \underset{\mathcal{K}}{\bowtie} (sendAlice, rc).(confirm, \top).Bob')$$

$$s6 : KDC \underset{\mathcal{L}}{\bowtie} ((confirm, rc).Alice'' \underset{\mathcal{K}}{\bowtie} (confirm, \top).Bob')$$

$$s7 : KDC \underset{\mathcal{L}}{\bowtie} (Alice'' \underset{\mathcal{K}}{\bowtie} Bob')$$

Markov process underlying PEPA model



Infinitesimal generator matrix

$$\begin{bmatrix} -r_q & r_q & 0 & 0 & 0 & 0 & 0 \\ 0 & -r_g & r_g & 0 & 0 & 0 & 0 \\ 0 & 0 & -r_p & r_p & 0 & 0 & 0 \\ 0 & 0 & 0 & -r_B & r_B & 0 & 0 \\ 0 & 0 & 0 & 0 & -r_A & r_A & 0 \\ 0 & 0 & 0 & 0 & 0 & -r_c & r_c \\ r_u & 0 & 0 & 0 & 0 & 0 & -r_u \end{bmatrix}$$

Transposed infinitesimal generator matrix

$$\begin{bmatrix} -r_q & 0 & 0 & 0 & 0 & 0 & r_u \\ r_q & -r_g & 0 & 0 & 0 & 0 & 0 \\ 0 & r_g & -r_p & 0 & 0 & 0 & 0 \\ 0 & 0 & r_p & -r_B & 0 & 0 & 0 \\ 0 & 0 & 0 & r_B & -r_A & 0 & 0 \\ 0 & 0 & 0 & 0 & r_A & -r_c & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

Global balance equations system

$$\begin{cases} -r_q\pi(X_1) + r_u\pi(X_7) = 0 \\ r_q\pi(X_1) - r_g\pi(X_2) = 0 \\ r_g\pi(X_2) - r_p\pi(X_3) = 0 \\ r_p\pi(X_3) - r_B\pi(X_4) = 0 \\ r_B\pi(X_4) - r_A\pi(X_5) = 0 \\ r_A\pi(X_5) - r_c\pi(X_6) = 0 \\ \pi(X_1) + \pi(X_2) + \pi(X_3) + \pi(X_4) + \pi(X_5) + \pi(X_6) + \pi(X_7) = 1 \end{cases}$$

Steady state distribution - formulas (I)

If we let R to be

$$
\begin{aligned}
R = r_A * r_B * r_c * r_g * r_p * r_q + r_A * r_B * r_c * r_g * r_p * r_u \\
+ r_A * r_B * r_c * r_g * r_q * r_u + r_A * r_B * r_c * r_p * r_q * r_u \\
+ r_A * r_B * r_g * r_p * r_q * r_u + r_A * r_c * r_g * r_p * r_q * r_u \\
+ r_B * r_c * r_g * r_p * r_q * r_u
\end{aligned}
$$

, we then have the following steady state distributions (next slide).

Steady state distribution - formulas (II)

$$\pi(X_1) = \frac{r_A * r_B * r_c * r_g * r_p * r_u}{R}$$

$$\pi(X_2) = \frac{r_A * r_B * r_c * r_p * r_q * r_u}{R}$$

$$\pi(X_3) = \frac{r_A * r_B * r_c * r_g * r_q * r_u}{R}$$

$$\pi(X_4) = \frac{r_A * r_c * r_g * r_p * r_q * r_u}{R}$$

$$\pi(X_5) = \frac{r_B * r_c * r_g * r_p * r_q * r_u}{R}$$

$$\pi(X_6) = \frac{r_A * r_B * r_g * r_p * r_q * r_u}{R}$$

$$\pi(X_7) = \frac{r_A * r_B * r_c * r_g * r_p * r_q}{R}$$

Utilisation (I)

The KDC will be utilised whenever it is engaged in a *request*, *generate* or *response* activity. Therefore, to derive the utilisation we associate a reward of 1 with each of these activities:

$$\rho_1 = 1 \qquad \rho_2 = 1 \qquad \rho_3 = 1 \qquad \rho_4 = 0 \qquad \rho_5 = 0 \qquad \rho_6 = 0 \qquad \rho_7 = 0$$

Thus, the utilisation of the KDC is the total probability that the model is in one of the states in which the KDC is in use, in our case:

$$\begin{aligned} U_{KDC} &= \rho_1 * \pi(X_1) + \rho_2 * \pi(X_2) + \rho_3 * \pi(X_3) \\ &= \pi(X_1) + \pi(X_2) + \pi(X_3) \end{aligned}$$

Utilisation (II)

If we let the parameters to be:

$$r_A = r_B = r_c = r_g = r_p = r_q = 1.0 \quad \text{and} \quad r_u = 1.1$$

, the results are

$$U_{KDC} = \frac{33}{76} = 43.42\%$$

Steady state probabilities

1	KDC	Alice	Bob	0.14473684210526316
2	(gener...	(respon...	Bob	0.14473684210526316
3	(respo...	(respon...	Bob	0.14473684210526316
4	KDC	Alice1	Bob	0.14473684210526316
5	KDC	(sendAli...	(sendAl...	0.14473684210526316
6	KDC	(confirm...	(confir...	0.14473684210526316
7	KDC	Alice2	Bob1	0.13157894736842105

Throughput (I)

The throughput of the KDC is the expected number of completed *(request*, *generate*, *response)* activities per unit of time. Since each activity is visited only once, this throughput is the same as the throughput of either of the activities.
The throughput of activity *generate* is found by associating a reward equal to the activity rate with each instance of the activity. Thus, the rewards associated with states are:

$$\rho_1 = 0 \qquad \rho_2 = 1 \qquad \rho_3 = 0 \qquad \rho_4 = 0 \qquad \rho_5 = 0 \qquad \rho_6 = 0 \qquad \rho_7 = 0$$

The throughput is then computed as:

$$T_{generate} = \rho_2 * \pi(X_2) = \pi(X_2)$$

Throughput (II)

If we let the parameters to be:

$$r_A = r_B = r_c = r_g = r_p = r_q = 1.0 \quad \text{and} \quad r_u = 1.1$$

, the results are

$$T_{generate} = \frac{11}{76} = 0.1447$$

generate	0.14473684210526316

Throughput (III)

The throughputs of the other activities follow:

$$T_{request} = \rho_1 * \pi(X_1) = \pi(X_1) = \frac{11}{76} = 0.1447$$

$$T_{response} = \rho_3 * \pi(X_3) = \pi(X_3) = \frac{11}{76} = 0.1447$$

$$T_{sendBob} = \rho_4 * \pi(X_4) = \pi(X_4) = \frac{11}{76} = 0.1447$$

$$T_{sendAlice} = \rho_5 * \pi(X_5) = \pi(X_5) = \frac{11}{76} = 0.1447$$

$$T_{useKey} = \rho_6 * \pi(X_6) = \pi(X_6) = \frac{11}{76} = 0.1447$$

$$T_{confirm} = \rho_7 * \pi(X_7) = 1.1 * \pi(X_7) = \frac{11}{76} = 0.1447$$

Action	Throughput
confirm	0.14473684210526316
generate	0.14473684210526316
request	0.14473684210526316
response	0.14473684210526316
sendAlice	0.14473684210526316
sendBob	0.14473684210526316
usekey	0.14473684210526316

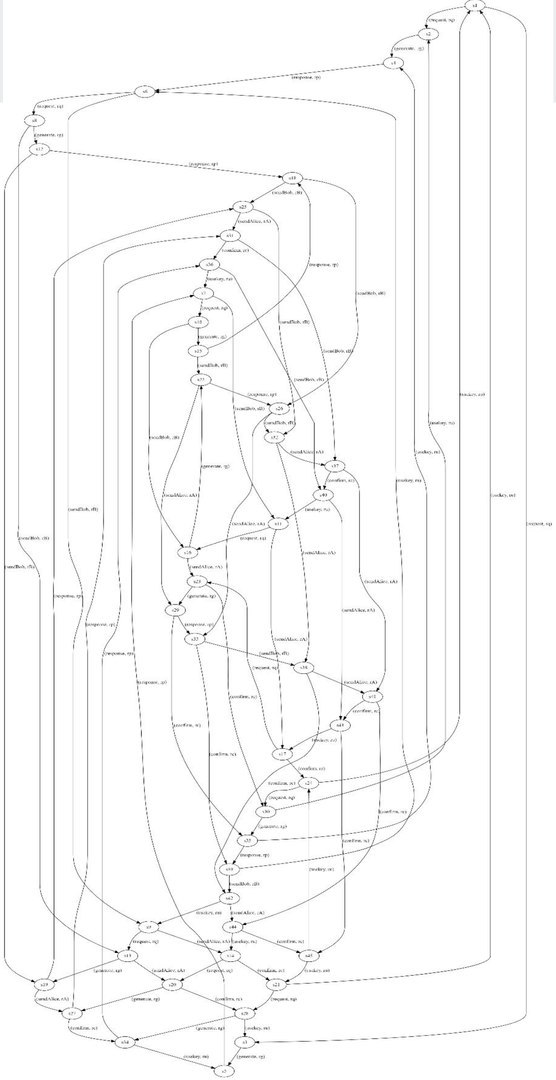
Bisimulation to aggregate the PEPA components

Derivation graph with two pairs

System (in PEPA) with 2 pairs of users:

$$System \stackrel{def}{=} KDC \underset{\mathcal{L}}{\bowtie} (Alice \underset{\mathcal{K}}{\bowtie} Bob \| Alice \underset{\mathcal{K}}{\bowtie} Bob)$$

- 1 pair: 7 states
- 2 pairs: 45 states
- 3 pairs: 275 states
- 4 pairs: 1625 states



Bisimulation

Two agents are considered to be bisimilar when their externally observed behaviour appears to be the same.

Two agents, $P, Q \in \mathcal{P}$, are strongly bisimilar ($P \sim Q$) iff, there is some relation $\mathcal{R}$ over $\mathcal{P} \times \mathcal{P}$ such that if $(P, Q) \in \mathcal{R}$ then for all $\alpha \in \mathcal{A}ct$:

- if $P \xrightarrow{\alpha} P'$, then for some Q', $Q \xrightarrow{\alpha} Q'$ and $(P', Q') \in \mathcal{R}$;
- if $Q \xrightarrow{\alpha} Q'$, then for some P',$P \xrightarrow{\alpha} P'$ and $(P', Q') \in \mathcal{R}$;

Thus, if P and Q are strongly bisimilar, any action performed by one must be matched by the other. Additionally, any subsequent action must also be matched without exception.

State-to-state: equivalences are used to simplify the model.
A set of equivalent states can be replaced by one macro-state.

Bisimulation in PEPA

In **PEPA** two components are strongly bisimilar if:

- any *a* activity of one can be matched by an *a* activity of the other
- every *a-derivative* of one is strongly bisimilar to some *a-derivative* of the the other
- the apparent rates of all action types are the same in the two components.

For any action type α, the total conditional transition rates from those components to any equivalence class, via activities of type α are the same.

$$q[P, S, \alpha] = q[Q, S, \alpha]$$

Proof sketch

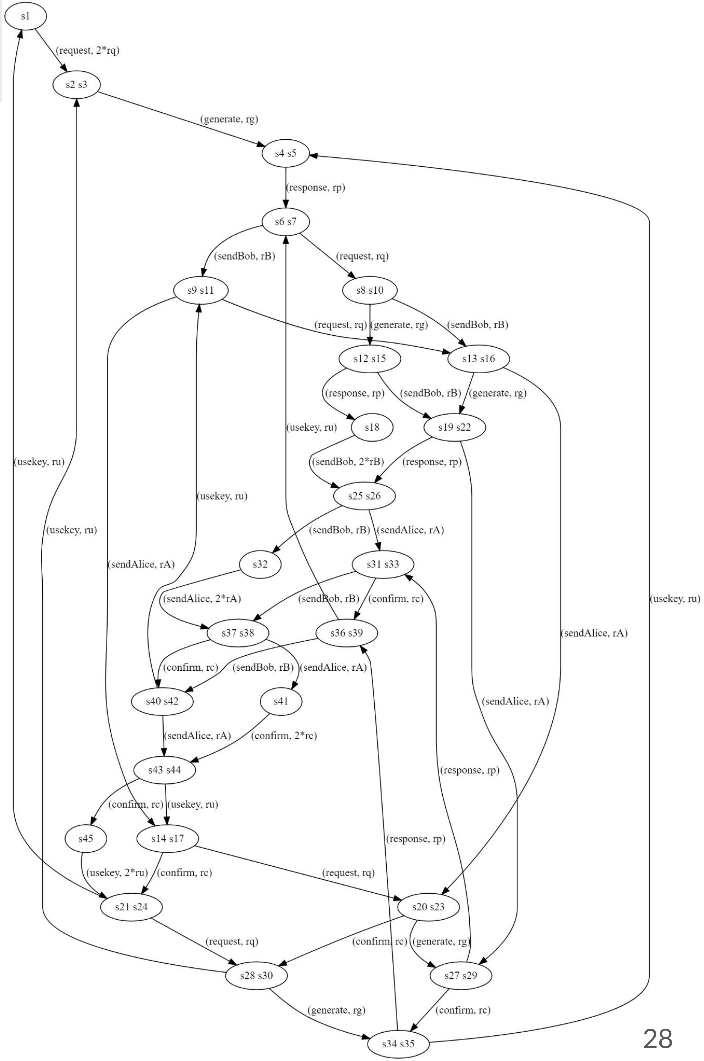
$$
\begin{aligned}
q[s_{21}, s_1, usekey] &= q(s_{21}, s_1) = r_u \\
q[s_{24}, s_1, usekey] &= q(s_{24}, s_1) = r_u \\
q[s_{21}, [s_{28}, s_{30}], request] &= q(s_{21}, s_{28}) + q(s_{21}, s_{30}) = r_q + 0 = r_q \\
q[s_{24}, [s_{28}, s_{30}], request] &= q(s_{24}, s_{28}) + q(s_{24}, s_{30}) = 0 + r_q = r_q \\
&\vdots \\
q[s_1, [s_2, s_3], request] &= q(s_1, s_2) + q(s_1, s_3) = r_q + r_q = 2r_q = q[s_1, (s_2, s_3), request] \\
&\vdots \\
q[s_{18}, [s_{25}, s_{26}], sendBob] &= q(s_{18}, s_{25}) + q(s_{18}, s_{26}) = r_B + r_B = 2r_B = q[s_{18}, (s_{25}, s_{26}), sendBob] \\
&\vdots \\
q[s_{32}, [s_{37}, s_{38}], sendAlice] &= q(s_{32}, s_{37}) + q(s_{37}, s_{38}) = r_A + r_A = 2r_A = q[s_{32}, (s_{37}, s_{38}), sendAlice] \\
&\vdots \\
q[s_{41}, [s_{43}, s_{44}], confirm] &= q(s_{41}, s_{43}) + q(s_{41}, s_{44}) = r_c + r_c = 2r_c = q[s_{41}, (s_{43}, s_{44}), confirm] \\
&\vdots \\
q[s_{45}, [s_{21}, s_{24}], usekey] &= q(s_{45}, s_{21}) + q(s_{45}, s_{24}) = r_u + r_u = 2r_u = q[s_{45}, (s_{21}, s_{24}), usekey]
\end{aligned}
$$

Aggregated graph

Equivalence classes:

$[1], [2, 3], [4, 5], [6, 7], [8, 10], [9, 11],$
$[12, 15], [13, 16], [14, 17], [18], [19, 22],$
$[20, 23], [21, 14], [25, 26], [27, 29], [28, 30],$
$[31, 33], [32], [34, 35], [36, 39], [37, 38],$
$[40, 42], [41], [43, 44], [45]$

45 states to 25 states



External behaviour

Macro-states

2	(generate, 1.0).(response, 1.0).KDC	(response, inf).Alice1	Bob	Alice	Bob
3	(generate, 1.0).(response, 1.0).KDC	Alice	Bob	(response, inf).Alice1	Bob

Single-states

18	KDC	Alice1	Bob	Alice1	Bob
45	KDC	Alice2	Bob1	Alice2	Bob1

Steady state probabilities - comparison

45 states

1	KDC	Alice	Bob	Alice	Bob	0.019093521178978577
2	(gener...	(respon...	Bob	Alice	Bob	0.04020761887107237
3	(gener...	Alice	Bob	(respon...	Bob	0.04020761887107239
4	(respo...	(respon...	Bob	Alice	Bob	0.060802669385485196
5	(respo...	Alice	Bob	(respon...	Bob	0.060802669385485224
6	KDC	Alice1	Bob	Alice	Bob	0.03974418389100009
7	KDC	Alice	Bob	Alice1	Bob	0.039744183891000096
8	(gener...	Alice1	Bob	(respon...	Bob	0.019872091945500044
9	KDC	(sendAli...	(sendA...	Alice	Bob	0.028325145982018726
10	(gener...	(respon...	Bob	Alice1	Bob	0.019872091945500048
11	KDC	Alice	Bob	(sendAli...	(sendA...	0.028325145982018723
12	(respo...	Alice1	Bob	(respon...	Bob	0.009936045972750022
13	(gener...	(sendAli...	(sendA...	(respon...	Bob	0.024098618963759387
14	KDC	(confir...	(confir...	Alice	Bob	0.022170729446658634
15	(respo...	(respon...	Bob	Alice1	Bob	0.009936045972750024
16	(gener...	(respon...	Bob	(sendAli...	(sendA...	0.024098618963759387
17	KDC	Alice	Bob	(confir...	(confir...	0.022170729446658634
18	KDC	Alice1	Bob	Alice1	Bob	0.009936045972750022

25 states

1	P1	0.019093521178978577
2	P2_3	0.08041523774214476
3	P4_5	0.12160533877097042
4	P6_7	0.07948836778200019
5	P8_10	0.039744183891000096
6	P9_11	0.05665029196403745
7	P12_15	0.019872091945500048
8	P13_16	0.048197237927518774
9	P14_17	0.044341458893317275
10	P18	0.009936045972750024

Analysis

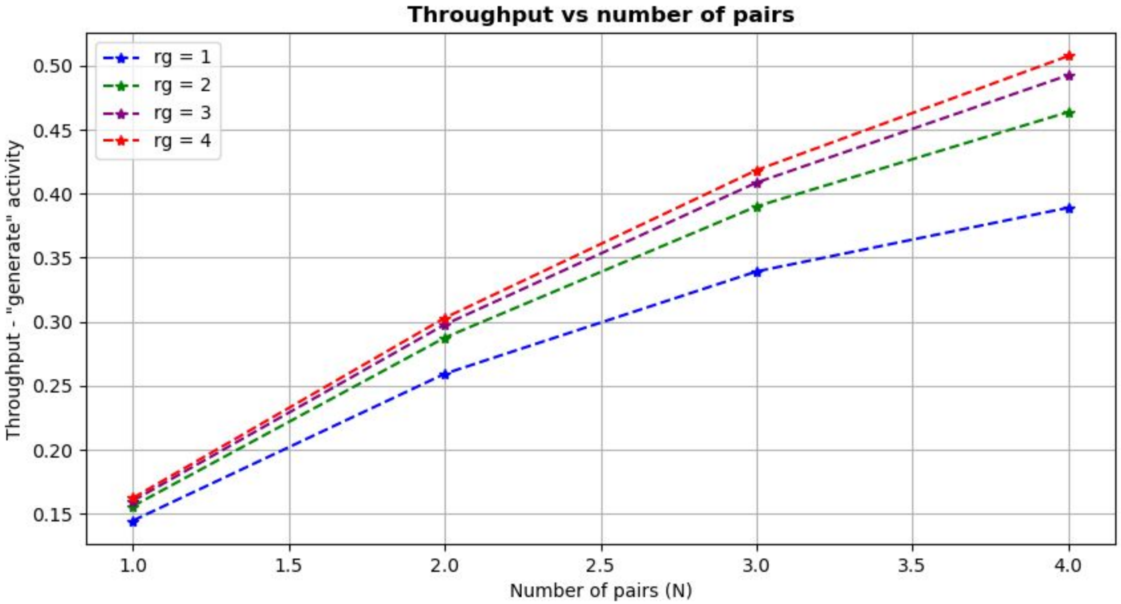
Modified model - PEPA Eclipse Plug-in

```
rp = 1.0;
rq = 1.0;
rB = 1.0;
rc = 1.0;
rA = 1.0;
ru = 1.1;
rg = 1.0;

KDC = (request, T).(generate,rg).(response, rp).KDC;

Alice = (request, rq).(response, T).Alice1;
Alice1 = (sendBob, rB).(sendAlice, T).(confirm, rc).Alice2;
Alice2 = (usekey, ru).Alice;

Bob = (sendBob, T).(sendAlice, rA).(confirm, T).Bob1;
Bob1 = (usekey, T).Bob;

KDC <request, response> ((Alice <sendBob, sendAlice, confirm, usekey> Bob) || (Alice <sendBob, sendAlice, confirm, usekey> Bob))
```

Original model

```
KDC = (request, T).KDC + (response, rp).KDC;
```

Throughput (I)



Throughput: comparison

$$\rho_A = \rho_B = \rho_c = \rho_g = \rho_p = \rho_q = 1.0, \quad \rho_u = 1.1 \quad \text{and } N = 1$$

Original model

Action	Throughput
confirm	0.16923076923076924
request	0.16923076923076924
response	0.16923076923076924
sendAlice	0.16923076923076924
sendBob	0.16923076923076924
usekey	0.16923076923076924

Modified model

Action	Throughput
confirm	0.14473684210526316
generate	0.14473684210526316
request	0.14473684210526316
response	0.14473684210526316
sendAlice	0.14473684210526316
sendBob	0.14473684210526316
usekey	0.14473684210526316

Utilisation - comparison

Original model

Utilisation | Throughput | Population

- KDC (1 local states)
  - KDC = 1.0
- Alice (6 local states)
  - (confirm, 1.0).Alice2 = 0.16923076923076924
  - (response, inf).Alice1 = 0.16923076923076924
  - (sendAlice, inf).(confirm, 1.0).Alice2 = 0.16923076923076924
  - Alice = 0.16923076923076924
  - Alice1 = 0.16923076923076924
  - Alice2 = 0.15384615384615385
- Bob (4 local states)
  - (confirm, inf).Bob1 = 0.16923076923076924
  - (sendAlice, 1.0).(confirm, inf).Bob1 = 0.16923076923076924
  - Bob = 0.5076923076923077
  - Bob1 = 0.15384615384615385

Modified model

Utilisation | Throughput | Population

- KDC (3 local states)
  - (generate, 1.0).(response, 1.0).KDC = 0.14473684210526316
  - (response, 1.0).KDC = 0.14473684210526316
  - KDC = 0.7105263157894737
- Alice (6 local states)
  - (confirm, 1.0).Alice2 = 0.14473684210526316
  - (response, inf).Alice1 = 0.2894736842105263
  - (sendAlice, inf).(confirm, 1.0).Alice2 = 0.14473684210526316
  - Alice = 0.14473684210526316
  - Alice1 = 0.14473684210526316
  - Alice2 = 0.13157894736842105
- Bob (4 local states)
  - (confirm, inf).Bob1 = 0.14473684210526316
  - (sendAlice, 1.0).(confirm, inf).Bob1 = 0.14473684210526316
  - Bob = 0.5789473684210527
  - Bob1 = 0.13157894736842105