Initial clauses

- Clause 0: attacker(true)
  (The attacker applies function true.)
- Clause 1: attacker(*v*) -> attacker(spk(*v*))
  (The attacker applies function spk.)
- Clause 2: attacker(*v*) && attacker(*v_1*) -> attacker(sign(*v*,*v_1*))
  (The attacker applies function sign.)
- Clause 3: attacker(*v*) && attacker(*v_1*) -> attacker(sencrypt(*v*,*v_1*))
  (The attacker applies function sencrypt.)
- Clause 4: attacker(sencrypt(*x*,*y*)) && attacker(*y*) -> attacker(*x*)
  (The attacker applies function sdecrypt.)
- Clause 5: attacker(*v*) -> attacker(pk(*v*))
  (The attacker applies function pk.)
- Clause 6: attacker(sign(*m_1*,*k_2*)) -> attacker(*m_1*)
  (The attacker applies function getmess.)
- Clause 7: attacker(false)
  (The attacker applies function false.)
- Clause 8: attacker(*v*) && attacker(*v_1*) -> attacker(encrypt(*v*,*v_1*))
  (The attacker applies function encrypt.)
- Clause 9: attacker(encrypt(*x*,pk(*y*))) && attacker(*y*) -> attacker(*x*)
  (The attacker applies function decrypt.)
- Clause 10: attacker(0)
  (The attacker applies function 0.)
- Clause 11: attacker(*v*) -> attacker(*v* + 1)
  (The attacker applies function +.)
- Clause 12: attacker(*v* + 1) -> attacker(*v*)
  (The attacker applies function - 1.)
- Clause 13: attacker(*v*) && attacker(*v_1*) && attacker(*v_2*) -> attacker((*v*,*v_1*,*v_2*))
  (The attacker applies function 3-tuple.)
- Clause 14: attacker((*v*,*v_1*,*v_2*)) -> attacker(*v*)
  (The attacker applies function 1-proj-3-tuple.)
- Clause 15: attacker((*v*,*v_1*,*v_2*)) -> attacker(*v_1*)
  (The attacker applies function 2-proj-3-tuple.)
- Clause 16: attacker((*v*,*v_1*,*v_2*)) -> attacker(*v_2*)
  (The attacker applies function 3-proj-3-tuple.)

- Clause 17: attacker(*v*) **&&** attacker(*v_1*) **->** attacker((*v*,*v_1*))
  (The attacker applies function 2-tuple.)

- Clause 18: attacker((*v*,*v_1*)) **->** attacker(*v*)
  (The attacker applies function 1-proj-2-tuple.)

- Clause 19: attacker((*v*,*v_1*)) **->** attacker(*v_1*)
  (The attacker applies function 2-proj-2-tuple.)

- Clause 20: attacker(*v*) **->** attacker((*v*))
  (The attacker applies function 1-tuple.)

- Clause 21: attacker((*v*)) **->** attacker(*v*)
  (The attacker applies function 1-proj-1-tuple.)

- Clause 22: mess(*v*,*v_1*) **&&** attacker(*v*) **->** attacker(*v_1*)
  (The attacker can listen on all channels it has.)

- Clause 23: attacker(*v*) **&&** attacker(*v_1*) **->** mess(*v*,*v_1*)
  (The attacker can send messages it has on all channels it has.)

- Clause 24: attacker(fail-any_type)
  (Initial knowledge of the attacker.)

- Clause 25: attacker(c[])
  (Initial knowledge of the attacker.)

- Clause 26: equal(*v*,*v*)
  (Definition of equal.)

- Clause 27: attacker(*new-name_1*)
  (The attacker can create new names.)
  Abbreviations:
  - *new-name_1* = new-name[!att = *v*]

- Clause 28: attacker(spk(skA[]))
  (The message spk(skA[]) may be sent to the attacker at output {3}.)

- Clause 29: attacker(pk(skB[]))
  (The message pk(skB[]) may be sent to the attacker at output {6}.)

- Clause 30: b-inj-event(beginBparam(*pk2_1*),*@occ10_1*) **&&** attacker(*pk2_1*) **->** attacker(encrypt(sign((spk(skA[]),*pk2_1*,*k_2*),skA[]),*pk2_1*))
  (If the message *pk2_1* is received from the attacker at input {9},
  event beginBparam(*pk2_1*) is executed at {10},
  then the message encrypt(sign((spk(skA[]),*pk2_1*,*k_2*),skA[]),*pk2_1*) may be sent to the attacker at output {12}.)
  Abbreviations:
  - *k_2* = k[pk2 = *pk2_1*,!1 = *@sid*]

  - *@occ10_1* = @occ10[pk2 = *pk2_1*,!1 = *@sid*]

- Clause 31: attacker(encrypt(sign((spk(skA[]),pk(skB[]),*k_2*),skA[]),pk(skB[]))) **->** attacker(sencrypt(secretB[],*k_2*))
  (If the message encrypt(sign((spk(skA[]),pk(skB[]),*k_2*),skA[]),pk(skB[])) is received from the

attacker at input {17},
then the message sencrypt(secretB[],*k_2*) may be sent to the attacker at output {20}.)

- Clause 32: attacker(encrypt(sign((spk(skA[]),pk(skB[]),*k_2*),skA[]),pk(skB[]))) -> inj-event(endBparam(pk(skB[])),*@occ21_1*)
  (If the message encrypt(sign((spk(skA[]),pk(skB[]),*k_2*),skA[]),pk(skB[])) is received from the attacker at input {17},
  then event endBparam(pk(skB[])) may be executed at {21} in session *@sid*.)
  Abbreviations:
  - *@occ21_1* = @occ21[km = encrypt(sign((spk(skA[]),pk(skB[]),*k_2*),skA[]),pk(skB[])),!1 = *@sid*]