Formal Verification of E-Commerce Protocol with Complex Trading Capabilities of Intermediaries

Cătălin V. Bîrjoveanu[1] and Mirela Bîrjoveanu[2]
[1]*Department of Computer Science, "Al.I.Cuza" University of Iaşi, Iaşi, Romania*
[2]*Vitesco Technologies, Iaşi, Romania*
catalin.birjoveanu@uaic.ro, mbirjoveanu@gmail.com

To formally verify *Protocol with Complex Trading Capabilities of Intermediaries (PCTCI)*, we use the Automated Validation of Internet Security Protocols and Applications (AVISPA) tool (Vigano, 2006), widely used by academic and industry researchers in the field of security protocols. An overview of the AVISPA tool and how to use it to formally verify complex protocols is provided in (Bîrjoveanu and Bîrjoveanu, 2022). Moreover, the results obtained in evaluation from (Lafourcade and Puys, 2016) regarding the most used tools for security protocols verification, led us believe that Cl-AtSe from AVISPA is the most suitable tool for analyzing such a complex protocol as the one we proposed.

1 AVISPA BACKGROUND

The protocol, its security properties, its scenario to verify and the intruder's knowledge are specified in the High Level Protocol Specification Language (HLPSL) (Vigano, 2006). An hlpsl2if translator is used to automatically translate the HLPSL specification into an intermediate format (IF), which is the input to the Cl-AtSe (Constraint-Logic based Attack Searcher) model checker (Turuani, 2006). Cl-AtSe considers a Dolev-Yao intruder (Dolev and Yao, 1983) that controls the communication channels under the *perfect cryptography assumption*.

As HLPSL is a role based language, it offers some special type of roles: *basic* and *composed*. A basic role defines the actions of a participant in a protocol session using transitions. The general form of a transition is described below:

$State = 1 \wedge Rcv(M1) = | > State' := 2 \wedge Snd(M2)$

The state of the role is specified by $State$ variable. The agents instantiating the basic roles communicate using parameters *Snd* and *Rcv*. The transition means: if the value of $State$ is 1 and the message $M1$ is received on *Rcv* channel, then the new value of $State$ is 2 and the message $M2$ is sent on the *Snd* channel.

A composed role is used to specify a protocol session by parallel composition of the basic roles in a *composition* section. A top-level composed role is the *environment role* that specifies the initial knowledge of the intruder and the protocol scenario to be verified. The protocol scenario is a composition of one or more protocol's sessions, where the intruder may play some protocol's roles as an honest agent.

The security requirements are modeled in the *goal section* using *goal facts*. A goal fact is specified in a basic role as outcome of a specific transition. Confidentiality and authentication are the only security goals that can be directly specified in HLPSL.

2 PROTOCOL SPECIFICATION IN HLPSL

The formal proof that *PCTCI* satisfies the security requirements demands the formal proof that each of the sub-protocols *ATP (Aggregate Transaction sub-protocol)*, *OTP (Optional Transaction sub-protocol)* and *CTP (Chained Transaction sub-protocol)* of *PCTCI* satisfies the corresponding security requirements. The specification of *PCTCI* in HLPSL involves the specification of each its sub-protocol in HLPSL.

For specification and formal verification of *PCTCI*, we will use the protocol scenario from Fig. 1. The blue colored intermediary B_1 plays *ATP*, the red one B_4 plays *OTP* and the magenta B_2 and green ones B_3, B_5 and B_6 play *CTP*.

In the following, we will give some details regarding the specification and verification of *ATP* in HLPSL. The participants involved in *ATP* are:

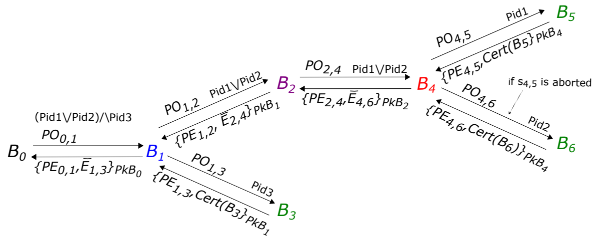

Figure 1: *PCTCI* scenario.

the customer $C(B_0)$, the intermediaries B_1, B_2, the provider B_3 and the payment gateway *PG*. B_1 receives a purchase request $PO_{0,1}$ from C to buy the aggregate product $(Pid1 \vee Pid2) \wedge Pid3$ and to fulfill it, he initiates an aggregate transaction $at_{1,3}$. B_1 sends $PO_{1,2}$ to B_2 for buying the optional product $(Pid1 \vee Pid2)$, and $PO_{1,3}$ to B_3 for buying $Pid3$.

Fig. 2 illustrates the first transition of broker 1 basic role played by B_1 and the first transition of payment gateway role played by *PG*. First transition of the broker 1 role corresponds to the reception of $PO_{0,1}$ from C, checking the novelty of the received order information by verifying the list $OIList0$ of order information received until then, and sending $PO_{1,2}$ to B_2. We encode any product in HLPSL by using the concatenation operator (denoted by .), and/or for the aggregate/optional product, and bg/en for open/closed brackets. Thus, the product $(Pid1 \vee Pid2) \wedge Pid3$ is encoded as follows: $bg.Pid1.or.Pid2.en.and.Pid3$.

Because the payment data like card number $Cn12$ that B_1 uses in $s_{1,2}$ is a sensitive data, it must be known only by B_1 and *PG*. To specify the confidentiality requirement for card number $Cn12$, we use *secret* goal fact in the form *secret(Cn12,scn12,{B1,PG})* in the first transition of broker 1 role. A similar confidentiality requirement is also added for card number $Cn13$ that B_1 uses in $s_{1,3}$. From the modeling point of view, we use two different card numbers for B_1 to be able to model and analyze various resolution scenarios.

An essential security requirement is the authentication of B_1 to *PG* with agreement on payment information. This requirement is specified by strong authentication of B_1 to *PG* that involves adding two goal facts, as follows:

- The goal fact *witness(B1,PG,pg_b1_pi1,B1.Cn12.Otp12.N01'.N12'.Am1'.or.Am2'.B2)*, in the first transition of the broker 1 role, meaning that B_1 wants to authenticate itself to *PG* on the payment information *B1.Cn12.Otp12.N01'.N12'.Am1'.or.Am2'.B2*;
- The goal fact *request(PG,B1,pg_b1_pi1,B1.Cn12'.Otp12'.N01'.N12'.Am1'.or.Am2'.B2)*, in the first transition of the payment gateway role (only after *PG* checks that B_1's payment information are authorized by checking his card information list $CIList1$), meaning that *PG* accepts the authentication of B_1 on the payment information.

ATP ensures strong authentication of B_1 to *PG* on payment information, if for an unique goal fact *request(PG,B1,pg_b1_pi1,B1.Cn12'.Otp12'.N01'.N12'.Am1'.or.Am2'.B2)*, an unique corresponding *witness(B1,PG,pg_b1_pi1,B1.Cn12.Otp12.N01'.N12'.Am1'.or.Am2'.B2)* was previously emitted. The strong authentication corresponds to the injective agreement from (Lowe, 1997). For a complete *ATP* verification we also specify the following security requirements (full specification can be found at (Bîrjoveanu and Bîrjoveanu, 2023)): confidentiality of C's card number, strong authentication of C to B_1, of B_1 to B_2, of B_1 to B_3 on corresponding order information, strong authentication of C to *PG* on his payment information, strong authentication of *PG* to C, of *PG* to B_1 on payment evidences generated by *PG*. As can be observed, we do not require the strong authentication of *PG* to B_2, *PG* to B_3 on payment evidences because this will be required in *CTP* played by B_2 and respectively B_3. Strong fairness in *ATP* is modeled by strong authentication of *PG* to B_1 and of *PG* to C on payment evidences, together with the fact that B_1 accepts strong authentication of *PG* on $PE_{1,2}$ and $PE_{1,3}$ in the same time with accepting strong authentication of *PG* on $PE_{0,1}$, evidences being either all successful or all aborted. Our specification from (Bîrjoveanu and Bîrjoveanu, 2023) includes also use cases that require application of *Res1* (e.g. $at_{1,3}$ is successful and $s_{0,1}$ is aborted) and *Res2* (e.g. B_1 does not receive the corresponding payment evidence from $s_{1,3}$) resolution sub-protocols.

Our AVISPA models from (Bîrjoveanu and Bîrjoveanu, 2023) incorporate the specification of *OTP* played by B_4 for buying the optional product $Pid1 \vee Pid2$, the specification of *CTP* played by B_2 for buying $Pid1 \vee Pid2$ and the specification of *CTP* played by the provider B_3 for buying $Pid3$. All these specifications contain also the resolution scenarios. Security requirements for *OTP* and *CTP* are modeled in the same manner as for *ATP*.

The *PCTCI* scenario we model is relevant and significant for verification because includes all three subprotocols (*ATP*, *OTP* and *CTP*) that can be played by the customer, intermediaries or providers. Our HLPSL specification of *PCTCI* needs a total number of 120 transitions in all roles from all sub-protocols.

We can extend the specification to any other protocol scenario by the appropriate parametrization of the basic roles from the corresponding sub-protocols (*ATP*, *OTP* and *CTP*) depending on the number of subtransactions belonging to the aggregate/optional

```
1. State = 0 /\ Rcv({X'.C.B1.bg.Pid1'.or.Pid2'.en.and.Pid3'.N01'.bg.Am1'.or.Am2'.en.and.Am3'.{H(C.B1.bg.Pid1'.or.Pid2'.en.and.Pid3'.N01'.bg.Am1'.
   or.Am2'.en.and.Am3')}_inv(PkC)}_Kcb1'.{Kcb1'}_PkB1) /\ not(in(C.B1.bg.Pid1'.or.Pid2'.en.and.Pid3'.N01'.bg.Am1'.or.Am2'.en.and.Am3'.{H(C.
   B1.bg.Pid1'.or.Pid2'.en.and.Pid3'.N01'.bg.Am1'.or.Am2'.en.and.Am3')}_inv(PkC), OIList0))
   =|>
   State' :=1 /\ OIList0' :=cons(C.B1.bg.Pid1'.or.Pid2'.en.and.Pid3'.N01'.bg.Am1'.or.Am2'.en.and.Am3'.{H(C.B1.bg.Pid1'.or.Pid2'.en.and.Pid3'.N01'.
   bg.Am1'.or.Am2'.en.and.Am3')}_inv(PkC) , OIList0) /\ N12' :=new() /\ Kb1b2' :=new()
   /\ Snd({{B1.Cn12.Otp12.N01'.N12'.Am1'.or.Am2'.B2.{H(B1.Cn12.Otp12.N01'.N12'.Am1'.or.Am2'.B2)}_inv(PkB1)}_PkPG.B1.B2.Pid1'.or.Pid2'.
         N01'.N12'.Am1'.or.Am2'.{H(B1.B2.Pid1'.or.Pid2'.N01'.N12'.Am1'.or.Am2')}_inv(PkB1)}_Kb1b2'.{Kb1b2'}_PkB2)
   /\ secret(Cn12,scn12,{B1,PG}) /\ witness(B1,PG,pg_b1_pi1,B1.Cn12.Otp12.N01'.N12'.Am1'.or.Am2'.B2)
   /\ witness(B1,B2,b2_b1_oi1,B1.B2.Pid1'.or.Pid2'.N01'.N12'.Am1'.or.Am2')
1. State = 0
   /\ Rcv({{B1.Cn12'.Otp12'.N01'.N12'.Am1'.or.Am2'.B2.{H(B1.Cn12'.Otp12'.N01'.N12'.Am1'.or.Am2'.B2)}_inv(PkB1)}_PkPG.Pid1'.N24'.{H(Pid1'.
         N01'.N12'.N24'.B1.B2.Am1')}_inv(PkB2)}_Kb2pg'.{Kb2pg'}_PkPG) /\ not(in({B1.Cn12'.Otp12'.N01'.N12'.Am1'.or.Am2'.B2.{H(B1.Cn12'.
         Otp12'.N01'.N12'.Am1'.or.Am2'.B2)}_inv(PkB1)}_PkPG.Pid1'.N24'.{H(Pid1'.N01'.N12'.N24'.B1.B2.Am1')}_inv(PkB2), PRList2))
   /\ in(B1.Cn12'.Otp12' , CIList1)
   =|>
   State' :=1 /\ PRList2' :=cons({B1.Cn12'.Otp12'.N01'.N12'.Am1'.or.Am2'.B2.{H(B1.Cn12'.Otp12'.N01'.N12'.Am1'.or.Am2'.B2)}_inv(PkB1)}_PkPG.
   Pid1'.N24'.{H(Pid1'.N01'.N12'.N24'.B1.B2.Am1')}_inv(PkB2) , PRList2)
   /\ Snd({y.B1.B2.N01'.N12'.Pid1'.{H(y.B1.B2.N01'.N12'.Pid1'.Am1')}_inv(PkPG).y.N01'.N12'.Pid1'.{H(y.N01'.N12'.Pid1')}_inv(PkPG).y.N01'.N12'.
         N24'.Pid1'.{H(y.N01'.N12'.N24'.Pid1')}_inv(PkPG)}_Kb2pg') /\ request(PG,B1,pg_b1_pi1,B1.Cn12'.Otp12'.N01'.N12'.Am1'.or.Am2'.B2)
   /\ witness(PG,B1,b1_pg_pe12,y.B1.B2.N01'.N12'.Pid1'.H(y.B1.B2.N01'.N12'.Pid1'.Am1').H(y.N01'.N12'.Pid1'))
```

Figure 2: *ATP* specification-First transition of broker 1 role played by B_1 and first transition of payment gateway role played by *PG*.

transaction and on the type of the required product.

3 PROTOCOL VERIFICATION RESULTS

Our results regarding the formal verification of *ATP*, *OTP* and *CTP* using Cl-AtSe in SPAN (Security Protocol Animator) (Span, 2017) are summarized in Table 1. Cl-AtSe uses constraint solving to find any protocol attack for a bounded number of protocol's sessions.

For the sub-protocol's verification we considered five concurrent sessions for *ATP* and *OTP*, and four concurrent sessions for *CTP*. In each case, first protocol's session is played only by the honest participants, and each next session considers the intruder playing a different basic role (excluding the payment gateway role). To simulate the resolution scenarios we consider different card information lists *CIList*s on the *PG*'s side containing authorized payment data.

The verification is done using typed model option, in which all variables and constants are typed. For each sub-protocol, we present the security requirements that are analyzed, the number of analyzed states, the result of the verification (SAFE) and the computation time. Also, we provide the number of transitions and the number of code lines (LoC) needed to specify each sub-protocol.

The verification results prove that Cl-AtSe did not find any attack and all specified security requirements are ensured. As we can see in Table 1, the complexity of each sub-protocol is reflected in the number of analyzed states and the needed computation time.

Strong fairness, non-repudiation, and integrity are obtained from strong authentication goals. For example, strong fairness in *ATP* is obtained from the strong authentication of *PG* to *C* on $PE_{0,1}$ and the strong authentication of *PG* to B_1 on $PE_{0,1}$, $PE_{1,2}$ and $PE_{1,3}$ in which either all payments evidences are successful, or all are aborted. Non-repudiation in *ATP* regarding *C* is a result of the strong authentication of *C* to *PG* on *C*'s payment information. Non-repudiation in *ATP* regarding B_1 is ensured by the strong authentication of B_1 to *PG* on B_1's payment information and by strong authentication of B_1 to *PG* on the corresponding payment request / abort request (for abortion cases). Integrity of $PE_{0,1}$ in *ATP* is ensured from the strong authentication of *PG* to *C* on $PE_{0,1}$ and the strong authentication of *PG* to B_1 on $PE_{0,1}$.

Results obtained for *CTP* in which B_1 sends the buying request directly to a provider are not included in Table 1 because they are similar with the results obtained for *CTP* when B_1 sends the buying request to an intermediary.

Next we discuss how *PCTCI* derives its security requirements based on security requirements guaranteed in each of its sub-protocols.

Confidentiality of participant's card numbers in *PCTCI* is guaranteed because each sub-protocol *ATP*, *OTP*, and *CTP* ensures their participant's card number confidentiality. Moreover, every message from *PCTCI* is transmitted hybrid encrypted with the public key of the authorized receiver in the sub-protocol to which it belongs.

Strong fairness in *PCTCI* is ensured if after its execution, any instance of *CTP*, *ATP* and *OTP* ensures strong fairness and all of them provides the same fairness level (either all aggregate, optional and chained

Table 1: Cl-AtSe verification results of security requirements

Sub-protocol (Participants)	Security requirement	Trans. No.	Result	Analyzed States	Time	LoC
ATP (C,B_1,B_2,B_3,PG)	SF, CONF for card numbers of C and B_1 SA of C to B_1, of B_1 to B_2, of B_1 to B_3 on OIs SA of C to PG, of B_1 to PG on their PIs SA of $B1$ to PG on PR/AR SA of PG to C, of PG to B_1 on PEs N-R regarding C and B_1, INT of PIs and PEs	31	SAFE	378572692	8h11m33s	495
CTP (B_1,B_2,B_4,PG)	SF, CONF for card numbers of $B1$ and B_2 SA of B_1 to B_2, of B_2 to B_4 on OIs SA of $B1$ to PG, of B_2 to PG on their PIs SA of $B2$ to PG on PR/AR SA of PG to $B1$, of PG to B_2 on PEs N-R regarding B_1 and B_2, INT of PIs and PEs	33	SAFE	63429279	2h34m12s	454
OTP (B_2,B_4,B_5,B_6,PG)	SF, CONF for card numbers of B_2 and B_4 SA of B_2 to B_4, of B_4 to B_5, of B_4 to B_6 on OIs SA of $B2$ to PG, of B_4 to PG on their PIs SA of $B4$ to PG on PR/AR SA of PG to $B2$, of PG to B_4 on PEs N-R regarding $B2$ and B_4, INT of PIs and PEs	42	SAFE	138020243	6h36m36s	578

SF=strong fairness CONF=confidentiality SA=strong authentication N-R=non-repudiation INT=integrity OIs=order information PIs=payment information PEs=payment evidences PR=payment request AR=abort request LoC=lines of code

transactions are successful, or all are aborted). As we formally proved, strong fairness in each sub-protocol played by a participant B_j guarantees that all transactions in which B_j is involved are either successful or aborted. The same level of fairness for all sub-protocols applied in *PCTCI* is ensured by way in which any intermediary behaves as a receiver in one sub-protocol and initiator in the next sub-protocol and also by the way the resolution sub-protocols are applied. For example, in the scenario considered in Fig.1, the behavior of B_4 as initiator in *OTP* and as receiver in *CTP* ensures that both instances of *OTP* and *CTP* have the same level of fairness. To be more concrete, we make an analysis according to the reverse way of completing the transactions in *PCTCI*. From the results obtained by formal proof, after the provider B_5 plays *CTP*, either $s_{4,5}$ is successfully completed or aborted. If $ot_{4,6}$ is aborted (meaning that $s_{4,5}$ and $s_{4,6}$ are aborted), then in the next step backwards, after B_4 plays *OTP*, both $s_{2,4}$ and $ot_{4,6}$ become aborted. If $ot_{4,6}$ is successful (meaning that $s_{4,5}$ is successful, or $s_{4,5}$ is aborted and $s_{4,6}$ is successful), then in the next step backwards, after B_4 plays *OTP*, either both $s_{2,4}$ and $ot_{4,6}$ are successful, or both aborted. The result of the last case of abortion is ensured because in the formal verification of *OTP* (and also *CTP* and *ATP*), we specify transitions that model the resolution sub-protocols in which the abortion of a subtransaction leads to cascading abortion of the following transactions already successfully completed. In the similar way, the behavior of B_2 ensures the same level of fairness both in *CTP* where he acts as initiator and in *ATP* where he acts as receiver.

We continue this reasoning, enlarging step by step the strong fairness result, until we obtain strong fairness for the sequence of transactions starting with the one initiated by customer. In conclusion, *PCTCI* ensures strong fairness, meaning either all aggregate, optional and chained transactions from the complex transaction are successful, or all are aborted.

Effectiveness If every party involved in *PCTCI* behaves honestly and there are no network communication errors or delays, then each instance of *ATP*/*OTP*/*CTP* corresponding to aggregate/optional/chained transactions from the complex transaction ensures effectiveness (meaning that each instance of *ATP*/*OTP*/*CTP* is successfully completed without *TTP* intervention). Therefore, effectiveness in *PCTCI* is obtained by a similar analysis with the one from strong fairness in which all transactions are successful.

Timeliness Each sub-protocol *CTP*, *ATP*, *OTP*, *Res1* and *Res2* from *PCTCI* ensures timeliness, because any intermediary who initiates a chained, aggregate or optional transaction, (and also the customer) waits a certain finite period of time to receive the corresponding payment evidences. If the period of time passes and C or an intermediary does not receive the appropriate payment evidences, he initiates *Res2* which in its turn provides timeliness because resilience of the communication channels used between customer/an intermediary/a provider and *PG*. Thus, *PCTCI* guarantees timeliness.

Non-repudiation As we obtained in the formal analysis, each instance of *ATP*/*OTP*/*CTP* applied in *PCTCI* guarantees non-repudiation by strong authentication requirements. For example, in *ATP*, non-repudiation regarding C is a result of the strong authentication of C to *PG* on C's payment information. Also, non-repudiation regarding B_1 in *ATP* is ensured by the strong authentication of B_1 to *PG* on B_1's payment information and also strong authentication of B_1 to *PG* on the corresponding payment request / abort request (for abortion cases). So, non-repudiation in *PCTCI* is ensured.

REFERENCES

Bîrjoveanu, C. V. and Bîrjoveanu, M. (2022). *Formal Verification of Multi-party Fair Exchange E-Commerce Protocols. In: Secure Multi-Party E-Commerce Protocols. SpringerBriefs in Computer Science*. Springer.

Bîrjoveanu, C. V. and Bîrjoveanu, M. (2023). *Formal Verification of PCTCI*. https://github.com/PCTCIDevelopers/PCTCI-Verification Last accessed May 22, 2023.

Dolev, D. and Yao, A. (1983). On the security of public-key protocols. *IEEE Transactions on Information Theory*.

Lafourcade, P. and Puys, M. (2016). Performance evaluations of cryptographic protocols verification tools dealing with algebraic properties. In *Lecture Notes in Computer Science*. Springer.

Lowe, G. (1997). A hierarchy of authentication specifications. In *10th CSF Workshop*. IEEE.

Span (2017). *SPAN, a Security Protocol ANimator for AVISPA*. http://people.irisa.fr/Thomas.Genet/span/.

Turuani, M. (2006). The cl-atse protocol analyser. In *17th International Conference on Rewriting Techniques and Applications*. Springer.

Vigano, L. (2006). Automated security protocol analysis with the avispa tool. *Electronic Notes in Theoretical Computer Science*.