Mapping Requirements Specifications into a Formalized Blockchain-Enabled Authentication Protocol for Secured Personal Identity Assurance

Benjamin Leiding[1] and Alex Norta[2]

[1] University of Göttingen, Institute of Computer Science, Göttingen, Germany
benjamin.leiding@cs.uni-goettingen.de
[2] Tallinn University of Technology, Department of Software Systems, Tallinn, Estonia
alex.norta.phd@ieee.org

Abstract. The design and development of novel security and authentication protocols is a challenging task. Design flaws, security and privacy issues as well as incomplete specifications pose risks for its users. Authcoin is a blockchain-based validation and authentication protocol for secure identity assurance. Formal methods, such as Colored Petri Nets (CPNs), are suitable to design, develop and analyze such new protocols in order to detect flaws and mitigate identified security risks. In this work, the Authcoin protocol is formalized using Colored Petri Nets resulting in a verifiable CPN model. An Agent-Oriented Modeling (AOM) methodology is used to create goal models and corresponding behavior models. Next, these models are used to derive the Authcoin CPN models. The modeling strategy as well as the required protocol semantics are explained in detail. Furthermore, we conduct a state-space analysis on the resulting CPN model and derive specific model properties. The result is a complete and correct formal specification that is used to guide future implementations of Authcoin.

Keywords: authcoin, colored petri net, authentication, security, trust, privacy, access control, identity, blockchain, smart contract, formal verification

1 Introduction

Preventing design flaws, security and privacy issues as well as incomplete specifications that pose a risk for its users, is a challenging task during the design and development of new security protocols in the field of computer science [6][7][14][33]. In a best case scenario, issues of a security protocol are inconvenient to users who rely on it, while in other cases, design flaws and errors are fatal. The broken encryption of a wireless network [31] is an example for the first case, whereas a broken security protocol that grants an attacker access to sensible parts of nuclear power plants [4] illustrates a more serious threat.

Formal methods, such as Petri nets [28], π-calculus [22] and communicating sequential processes [16], are utilized for the design, development and analysis of new as well as existing protocols, thereby eliminating, or minimizing the security issues of the targeted protocols [1][9]. Most recently, a formal security analysis on the underlying protocol of the popular Signal Messenger[3] by Cohn-Gordon et al. [8], results in the demonstration of several standard security properties provided by the protocol.

In this work, a formal model of the Authcoin protocol [20] is created using Colored Petri Nets (CPNs) [17][18] in order to detect and eliminate eventual design flaws, missing specifications as well as security and privacy issues. A CPN is a graphical oriented language for the design, specification, simulation as well as the verification of systems and describes the states of a modeled system and the events (transitions) that cause the system to change states. CPN models are represented using a directed bipartite graph that consists of places, transitions, arcs and tokens. Places are denoted as circles and transitions as rectangles. Arcs connect places with transitions or transitions with places and have inscriptions in CPN-ML expressions [3][13][17][18][26]. In this work, CPN-Tools[4] is used for designing, evaluating and verifying the CPN models of Authcoin. The result is a formal specification of the protocol that is used to guide future implementations of Authcoin.

While an in-depth description of the Authcoin protocol is available [20], a gap exists with respect to the process of deriving the corresponding formal CPN model. This work fills the detected gap and investigates the research question of how to formalize the Authcoin protocol using the formalism of Colored Petri Nets? In order to answer this question with a separation of concerns, the main research question is divided into three subquestions: First, what is the top level of the formal CPN model of the Authcoin protocol? Second, what protocol semantics are required? Third, what are the refining CPN models?

The remainder of this paper is structured as follows. Section 2 provides a short introduction to the Authcoin protocol itself. Section 3 outlines the modeling strategy used to derive the top level CPN model of Authcoin, followed by Section 4 that deals with the required protocol semantics. The further refined CPN sub-modules are presented in Section 5. An evaluation of the created formal model is performed in Section 6 which also discusses related work. Section 7 concludes this work and provides directions for future work.

2 Authcoin

We provide a high level overview on Authcoin, its workflow and basic functionalities in order to equip the reader with a basic understanding of the protocol. The content is based on the original Authcoin paper [20] that provides further and more detailed information.

[3] https://whispersystems.org/
[4] http://cpntools.org/

The Authcoin protocol is an alternative concept to the commonly used public key infrastructures such as central authorities and the PGP Web of Trust (WoT). Authcoin combines a challenge response-based validation and authentication process for domains, certificates, email accounts and public keys, with the advantages of a blockchain-based storage system. The blockchain technology provides a publicly available, transparent and fault tolerant mechanism for storing data in a distributed and decentralized manner [25]. Authcoin uses an Ethereum-powered [5][34] blockchain-based storage system. The advantage of Ethereum and similar systems is that they do not only provide a blockchain-based storage but also incorporate Turing-complete programming languages on the protocol-layer on top of a blockchain in order to realize smart-contract capabilities which enables Authcoin users to create customized challenges for their validation and authentication processes.

The Authcoin protocol distinguishes between validation and authentication. In the context of Authcoin, validation aims to prove the following three facts: First, a specific entity has access to a certain account that is under validation, e.g, an email account. Second, a certain entity has access to a specified private- and public key. Third, the specified key pair corresponds to the tested account. The authentication of Authcoin continues the validation procedure by verifying the identity and aims to confirm that the alleged owner is also the actual owner of the public key. As illustrated in Figure 1, users can set up challenges for other entities and ask them to fulfill these challenges. Either the entity fails to do so, or is able to successfully complete the challenge and create a corresponding response. The chosen challenge depends on the use-case scenario, the required level of security and the given threat level of the involved entities. For example, to authenticate Bob, Alice challenges him to send her a photo of him holding his passport and the latest issue of a specific newspaper. Bob decides to fulfill her request, provide the required documents and Alice decides whether the response satisfies her request. Both upload the request and response to the blockchain. Even though Leiding et al. [20] outline a public/private key based solution, it is also possible to abstract from this assumption and use other identifiers, e.g., biometric identifiers. More detailed information on the Authcoin protocol itself, the challenge design, its security and further features are available in [19][20].

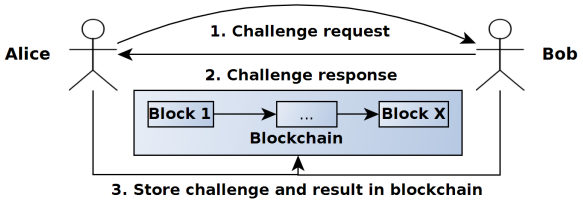

Fig. 1: General validation and authentication procedure (Source: [20])

3 Modeling the Authcoin Protocol

In order to model the Authcoin protocol using CPN, an appropriate modeling strategy is required, mapping the existing descriptions of the protocol, as outlined in Section 2, to the corresponding elements of a CPN model. Mapping the informal descriptions and requirements of the Authcoin protocol to a formal model using Colored Petri Nets (CPNs), results in a sound CPN model. Based on this model it is possible to consider concurrency conflicts, dependability issues and detect and eliminate eventual design flaws as well as security and privacy issues. To do so, Section 3.1 first outlines the modeling strategy used to create the CPN models, followed by Section 3.2 that presents the resulting top level CPN model of the protocol.

3.1 Modeling Strategy

Authcoin organizes and defines the exchange of validation and authentication information between different entities that are modeled as agents. In software engineering, various agent-oriented approaches exist, such as: Tropos [15], Gaia [35], Prometheus [27], MASB [23][24] and MaSE [11]. In [21], Mahunnah et al. introduce a mapping heuristics from agent models to CPN models based on Sterling's and Taveter's [30] methodology for Agent-Oriented Modeling (AOM).

There are two types of AOM models that are necessary to represent the Authcoin protocol, i.e., the goal model and the behavioral model. An AOM goal model is used to capture the functional requirements of a system in the form of goals, as well as non-functional requirements and roles of involved entities. Non-functional requirements represent quality goals of the system [30].

Figure 2 illustrates the top level goal model of Authcoin. A goal is represented in form of a parallelogram, quality goals in the form of clouds and sticky men represent roles. Functional requirements of the goal model are structured in a tree-like hierarchy with the overall objective of the system at the top. The main objective of Authcoin is to provide a secure and reliable validation and authentication protocol. The main goal is further decomposed into multi-layered sub-goals until the lowest atomic sub-goal is reached. In the context of Authcoin, the main goal is further divided into the following sub-goals: Key generation and establishing a binding, validation and authentication processing, mining and revocations. The three quality goals "secure", "correct" and "reliable" are attached to the overall main goal of the goal model, meaning they are relevant to all sub-goals and are inherited.

The AOM behavior model refines the previously developed goal model for specific agents and activities. A behavior model in AOM has two parts: an agent behavior model coupled with a behavior interface model [30]. The former describes the rule-based behavior of an agent, while the latter focuses on identifying activities with associated triggers, preconditions and postconditions [21]. Table 1 presents the behavior interface model of the goals depicted in the top level goal model of Figure 2. Each activity is listed with its corresponding trigger,

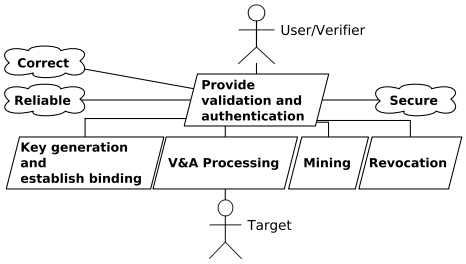

Fig. 2: Authcoin - Top level goal model (Adapted from [19])

optional preconditions and its postconditions. The execution of an activity is either triggered by an event, or by a precondition after the occurrence of an event [21]. The *Mining*-activity in Table 1 is triggered by receiving input transactions for the next block of the blockchain - to do so, the precondition has to be fulfilled. After the activity's execution, the proposed input transactions are either available on the blockchain or an error occurred and triggered a failure message. Complete and detailed listings of all remaining goal models, behavior interface models are available in Appendix A.2 and Appendix A.3. All acronyms, names and abbreviations as well as a description of the token colors of the Auhtcoin CPN model are available in Appendix A.4.

Table 1: Exemplary behavioral interfaces of activities for Authcoin

Activity	**Trigger**	**Precondition**	**Postcondition**
Key generation and establish binding	User wants to create a new key pair	Identifier list, key expiration date, key type, key length	Key pair, EIR on blockchain, EIR
V&A Processing	Received EIRs for V&A	Verifier EIR, target EIR	V&A results on blockchain or failure message
Mining	Received input for blockchain	Input transactions	CR, RR and SR on blockchain and VARs or failure message
Revocation	User wants to revoke an EIR or a SR	KeyPair, EIR, SR, CR, RR, VARs	Revoked EIR or SR and updated information on blockchain

3.2 Mapping the Agent-Oriented Model to CPN Models

Mapping the created AOM models to CPN is the final step to derive the top level CPN model of Authcoin. Table 2 and Figure 3 illustrate the mapping heuristic of AOM goal models to CPN as well as the mapping of behavior interface models to CPN.

Notation	Name
→	Connecting Arc
▭	Sub-Goal or Activity
⬭→	Trigger or Precondition
→⬭	Postcondition
▣	Goal

Table 2: Notation for mapping AOM to CPN (Source: [21])

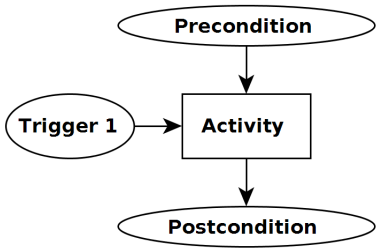

Fig. 3: Mapping a behavior interface model to a CPN model (Source: [19])

Directed arcs connect places and transitions representing the protocol execution through activities. Transitions represent activities, or sub-goals and CPN modules, depicted in the form of double-boarded rectangles, illustrate goals derived from the goal model. We refine the CPN modules into smaller sub-elements of the overall model, mapping to the same relation between goals and sub-goals in AOM. As illustrated in Figure 3, places with outgoing arcs either act as triggers, or represent a precondition, whereas places with incoming arcs represent postconditions of a given activity in AOM [21].

The final result of the mapping process is illustrated in Figure 4, presenting the complete and formalized top level CPN model of Authcoin derived from AOM and implemented using CPN-Tools. The top level CPN model consists of the four sub-modules derived from the four sub-goals of the top level AOM goal model. A detailed explanation of Figure 4, as well as further depictions and the refined implementation of these sub-modules are available in Section 5. The CPN token color sets, names and abbreviations used in the model are introduced in the following Section 4.

4 Protocol Semantics

This section introduces the set of CPN token color sets, names and acronyms used in the process of implementing the CPN models. Token colors represent data structures of Authcoin data objects that are used to illustrate the data flow through the CPN model. Tokens are transfered and manipulated by CPN transitions and ensure that the data objects adhere to the specified data syntax. Exemplary some acronyms, names and description of token colors of Auhtcoin's top level module are presented in Table 3. The first column specifies the module of the first occurrence of a certain token, name or acronym. The second column specifies the name, followed by a short description in column number three. The last column provides information on the data types. A complete listing of all

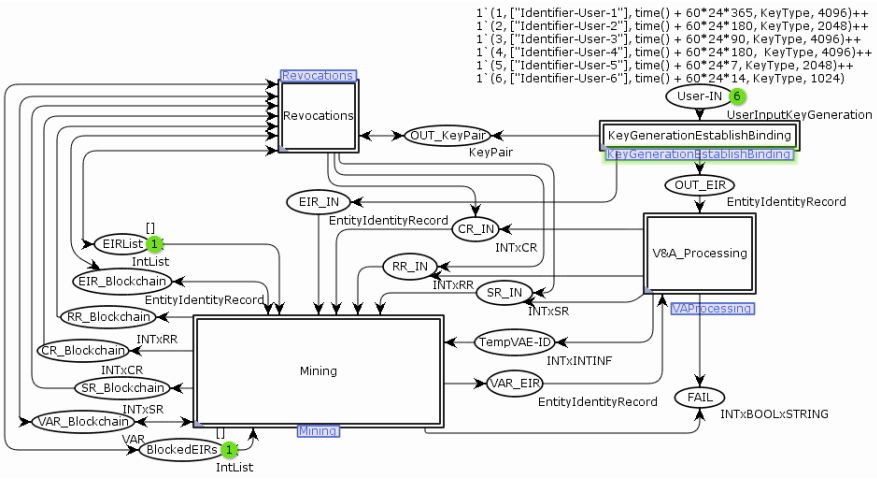

Fig. 4: Authcoin - Top level CPN model (Source: [19])

acronyms, names and abbreviations as well as a description of the token colors of the Auhtcoin CPN model is available in Appendix A.4.

Table 3: Exemplary acronyms, names and description of token colors of Auhtcoin's top level module.

Module	**Token color**	**Description**	**Type**
Top Level	PublicKey	Public key	(KeyFingerprint, Key, ExpirationDateUTC, KeyType, KeyLength)
Top Level	PrivateKey	Private key	String
Top Level	ChallengeRecord	Contains all information of a V&A challenge	(CR_ID, VAE_ID, Timestamp, ChallengeType, Challenge, VerifierEIR_ID, VerificationTargetEIR_ID)
Top Level	ResponseRecord	Contains all information regarding a V&A response	(RR_ID, VAE_ID, Timestamp, CorrespondCR_ID, Response)

5 Refined CPN Models of Authcoin

We presents the further refined sub-modules of the main modules of the top level CPN model presented in Figure 4. The sub-module refinements are derived from the AOM model using the same process as outlined for the top-level CPN model in Section 3. Due to page limitations, only the first level of module refinements is presented and explained in detail. The top level CPN model in

Figure 4 consists of four sub-modules. Each of the following sections focuses on one of these modules and provides further explanations. Section 5.1 focuses on the *KeyGenerationEstablishBinding* module, Section 5.2 on the *V&A-Processing* module, Section 5.3 on the *Mining* module and Section 5.4 on the *Revocations* module.

For further information on the remaining refined sub-modules and additional explanations, we refer the reader to [19]. In addition, the CPN-Tool source file of the presented CPN model is available in Appendix A.1.

5.1 *KeyGenerationEstablishBinding* Module

The first sub-module of the Authcoin top level CPN model is the *KeyGenerationEstablishBinding* module. The module is illustrated in Figure 5 and describes the process of generating a new key pair and establishing a binding between the key pair and the owning entity. The user provides a list of identifiers, an expiration date, a key type and the desired key length as input. The input is processed and results in a new key pair and an EntityIdentityRecord (EIR) that is posted to the blockchain. An EIR contains all identity related information of an entity.

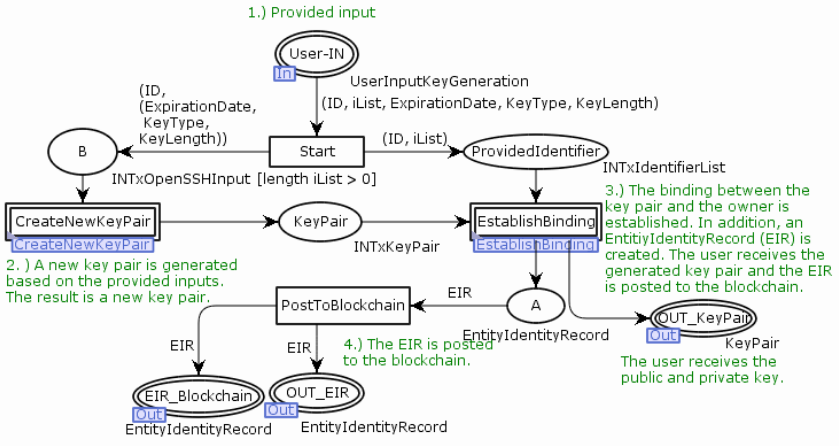

Fig. 5: CPN model of the *KeyGenerationEstablishBinding* module (Source: [19])

5.2 *V&A-Processing* Module

Figure 6 illustrates the *V&A-Processing* module in more detail. A set of EIRs is provided as an input, one for the target and one for the verifier of a validation and authentication (V&A) procedure. Both EIRs are further processed to create a VAE (V&A Entry) that consists of an ID for this specific V&A process and the target as well as the verifier EIR. The VAE is then further processed in

the *FormalValidation* module. The formal validation procedure is executed for both EIRs. It is checked if the public key of each EIR is well formed, has a sufficient key length and has not been expired or revoked yet. If one test fails, the V&A processing fails. Otherwise the VAE is further processed in the *V&A* module. During the V&A process, the verifier and target exchange challenges (CR - ChallengeRecord), with each other and create the corresponding responses (RR - ResponseRecord). Both entities evaluate the received responses and create corresponding signatures (SR - SignatureRecord) depending on whether they are satisfied with the received response or not. All information is posted to the publicly available blockchain. If any of these steps fail, the whole V&A process stops and the specific VAE is marked as failed.

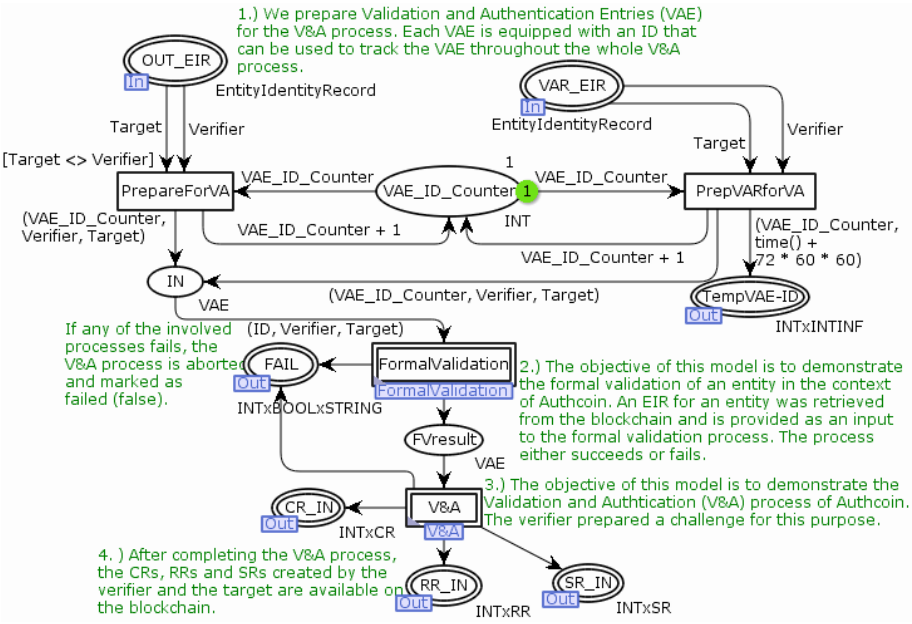

Fig. 6: CPN model of the *V&A-Processing* module (Adapted from [19])

5.3 *Mining* Module

Depending on whether the V&A processing finished successfully, the corresponding information (CRs, RRs and SRs) are posted to the blockchain, as illustrated in Figure 7. The actual blockchain-mining process is implemented in a symbolic way, meaning that each input transaction is directly mined into a new block without actually simulating a blockchain consensus algorithm and further block-building procedures. In the context of this paper, a symbolic implementation is sufficient since Authcoin can be deployed on top of different blockchain

architectures with varying mining-concepts. Furthermore, the process of mining a new block does not affect the protocol itself as long as it guarantees that a transaction posted to the blockchain is mined in a given timespan. It is only relevant for Authcoin that with each new block, a defined number of validation and authentication requests (VARs) is generated.

In our CPN model, every time a new EIR, CR, RR or SR is posted to the blockchain, a new block is mined followed by the creation of new VAR(s) that are also posted to the chain. The creation of new VAR(s) is triggered when a new block is added to the blockchain. The detailed role and functionality of a VAR is out of scope of this work and more detailed information are available in [20]. The presented CPN model is different in two aspects from description of VARs in [20]. First, in the model, users do not chose VARs on their own and instead a user is randomly chosen. Second, the model is limited to process only two VARs in order to avoid an endless loop during the simulation.

The *ProcessVAR* module in the *Mining* module describes the processing steps of a VAR chosen by a user. First, the status of the VAR is updated in order to avoid multiple processing of the same VAR. Afterwards, the EIR of the VAR's target is retrieved. In combination with the verifier's EIR, the V&A process of the *V&A-Processing* module is triggered and executed. The results of the V&A process are used to update the pending VAR and change the status of the VAR to "finished". The updates are posted to the blockchain.

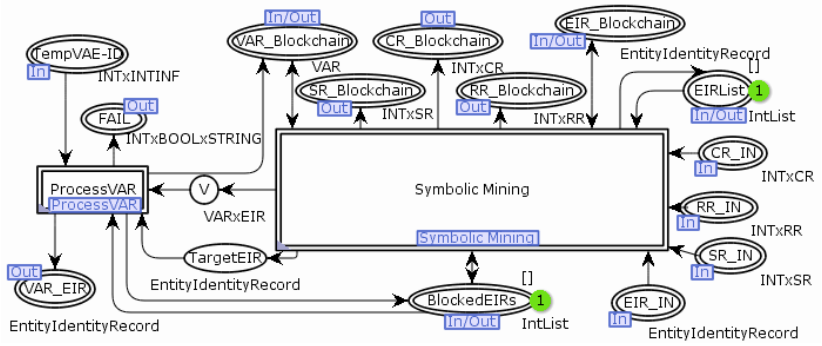

Fig. 7: CPN model of the *Mining* module (Adapted from [19])

5.4 *Revocation* Module

The *Revocation* module is illustrated in Figure 8. In the context of Authcoin, it is possible to revoke either signatures in the form of SRs, or to revoke entity records in the form of EIRs. An EIR can be revoked if it is no longer required, or not trustworthy anymore. SRs can be revoked in case that the signing entity has to remove the expressed trust relationship. The updated revocation information is posted back to the blockchain.

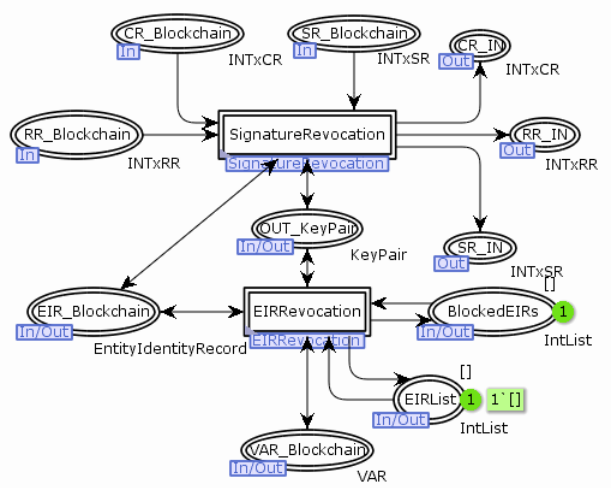

Fig. 8: CPN model of the *Revocations* module (Source: [19])

6 Evaluation

The following section focuses on evaluating the created CPN model by performing state-space analyses. In order to avoid a state-space explosion [18], the model is separated into sub-modules and a full state-space is calculated for each. From the results of the analyses, we derive certain model properties and explain their implications.

During a state-space analysis, all reachable states and state changes of a given CPN model are calculated and represented in a directed graph, where the nodes correspond to the set of reachable markings and the arcs correspond to occurring binding elements [18]. From this graph, it is possible to deduce certain properties of the CPN models and the systems presented by the models. The state-space analysis used in this work is generated using built-in functionalities of CPN-Tools. Besides a full state-space analysis, a SCC graph (Strongly Connected Component) is calculated based on the directed graph of the state- space analysis. The nodes of the SCC graph are obtained by making a disjoint division of the nodes in the state- space such that two state-space nodes are in the same SCC if and only if they are mutually reachable, i.e., there exists a path in the state-space from the first node to the second node and vice versa [18]. The SCC graph is used to deduce further model properties, e.g., if the SCC graph has less nodes than the state-space graph then at least one cycle exists in the state-space graph of the CPN model.

6.1 State-Space Analysis

Since a full computational verification of the whole CPN model is not feasible for this size of models and causes a state-space explosion, all parts of the models

are tested independently. As presented in Table 4, the CPN model is split into six sub-modules, that are tested with prepared input statements that aim to cover as many execution paths as possible without causing a state-space explosion. The *KeyGenerationEstablishBinding* module is depicted in Figure 5, the *FormalValidation* module and the *V&A* module are depicted in Figure 6 and the *Revocations* module in Figure 8. The *VARcreation* module refers to a slightly modified version of the *SymbolicMining* module illustrated in Figure 7. The *ProcessVAR* module consists of the *Mining* module in combination with the *V&A* module, minus the *SymbolicMining* module.

A full state-space is calculated for all modules listed above, followed by the calculation of the SCC graph. During these calculations, all other parts of the CPN models have been disabled. Relevant results and selected properties derived from the analysis are presented in the following Table 4. It is important to keep in mind that the properties of the separated modules might differ from the properties of the whole CPN model itself due to the combination and influences of the different components on each other. Nevertheless, verifying the correct execution and behavior of sub-modules strengthens the assumptions that the protocol performs as intended. The complete results of the state-space analyses are available in Appendix A.5.

Table 4: Selected state-space analysis results of the Authcoin CPN models.

Module	**Loops**	**Home markings**	**Dead markings**	**Dead transitions**	**Live transitions**
Key Generation Establish Binding	No	No	Yes	No	No
Formal Validation	No	No	Yes	Yes	No
Validation & Authentication	No	No	Yes	Yes	No
VAR Creation	No	No	Yes	No	No
Process VAR	No	No	Yes	Yes	No
Revocations	No	No	Yes	Yes	No

As shown in Table 4, none of the tested modules contain any loops. Thus, there are no infinite occurrences of execution paths in the state-space graph which guarantee the termination of each module. Still, it is possible to deduce from the design of the Authcoin protocol that there are loops in the complete model, since the blockchain architecture causes loops when chaining new blocks to the blockchain.

Furthermore, the state-space analysis shows the absence of any home markings. A home marking is a marking that can be reached from any other reachable marking, meaning that it is impossible to have an occurrence sequence that cannot be extended to reach the home marking.

The detected dead markings are caused either by intentionally disabling certain parts of the CPN models or customized input values that prevent a state-space explosion. "A dead marking is a marking in which no binding elements are enabled" [18]. The existence of at least one dead marking guarantees a termination of executable actions at a certain point, thereby preventing infinite runtime. Since all modules contain a dead marking, none of them has a live transition. By definition, "a transition is live if from any reachable marking we can always find an occurrence sequence containing the transition" [18].

All detected dead transitions are caused by the intentional disabling of execution paths and prepared input statements. A transition is considered dead if there is no reachable marking that enables the transition. Since all occurrences of dead transitions are artificially enforced, it means that all transitions of all tested modules can be potentially enabled at a certain point during the protocol execution [18].

Limitations of the evaluation result from the customized input statements used to prevent state-space explosions. Even though the input statements are designed to cover as many executions paths of the state-space graph as possible, certain places and transition with minor relevance had to be left out intentionally in order to avoid a state-space explosion. For the same reason, the number of iterative executions of the *VARcreation* module has been limited artificially in such a manner, that only one VAR is created.

6.2 Related Work

The general feasibility of using CPNs for formalizing, analyzing and verifying existing protocols is demonstrated by, e.g., [3][13][32]. Similar to this work, Vanek and Rohlik [32] demonstrate an implementation and simulation of an authentication protocol and show how CPNs can be used to create and simulate fully functional models. Furthermore, [12][29][36] use CPN to model attacks on security protocols and perform security analyses as well as verifications. Previously mentioned authors also experience issues with state-space explosions and handle them either by model simplification or prepared input statements.

Cohn-Gordon et al. [8] also acknowledge limitations of their formal analysis caused by simplification and intentionally limiting the scope of the model. As mentioned earlier, the derived CPN models of Authcoin also have certain limitations, e.g., a symbolic implementation of the blockchain mining process and artificially limited number of processed VARs. Further limitations concern the socio-technical nature of the Authcoin protocol, resulting in simplification of certain aspects of the models.

7 Conclusion and Future Work

The Authcoin protocol is formalized using Colored Petri Nets and an agent-oriented modeling methodology resulting in the CPN model of Authcoin. The utilized modeling strategy and the required protocol semantics are explained.

Furthermore, the top level model and the refined sub-modules are illustrated and described.

We formalize Authcoin using Colored Petri Nets and an agent-oriented modeling methodology used for creating the goal models as well as the corresponding behavior models. Based on these models, the top level CPN model as well as the refined CPN models are derived and implemented using CPN-Tools. The top level CPN model consists of four sub-modules that are further refined and described. We present the required protocol semantics by defining the necessary token color sets representing the used data structures. Moreover, we acknowledge and discuss the limitations of the CPN models resulting either from simplification of operations and models, or the socio-technical nature of the Authcoin protocol.

For future work, we plan to replace the symbolic modeling and mining of the blockchain with a complete CPN model of a specific blockchain implementation for better integration of the VAR creation process into the Authcoin CPN model. Furthermore, enhancing the security of the designed CPN models by applying security risk-oriented patterns [2] to the CPN models is part of our agenda. We also aim to implement and deploy the Authcoin protocol on the Qtum blockchain [10] that is compatible with Ethereums smart-contract system and therefore also enables a seamless integration of Authcoin with other Qtum-powered applications such as Agrello[5].

A Appendix

A.1 CPN Model

https://owncloud.gwdg.de/index.php/s/tfCNiFQtdNJKESG

A.2 Goal Model

https://owncloud.gwdg.de/index.php/s/yUHx8DFN57TqTIn

A.3 Behavior Interfaces of Activities

https://owncloud.gwdg.de/index.php/s/JCMD0ft3tQUbeSp

A.4 Protocol Semantics

https://owncloud.gwdg.de/index.php/s/OS7WcjrUEklSH0T

A.5 State-Space Analysis

https://owncloud.gwdg.de/index.php/s/nQwMLr7mK78z8wk

[5] https://www.agrello.org/

References

1. Abadi, M., Gordon, A.D.: A Calculus for Cryptographic Protocols: The Spi Calculus. In: Proceedings of the 4th ACM Conference on Computer and Communications Security. pp. 36–47. ACM (1997)
2. Ahmed, N., Matulevičius, R.: Securing Business Processes Using Security Risk-oriented Patterns. Computer Standards & Interfaces 36(4), 723–733 (2014)
3. Aly, S., Mustafa, K.: Protocol Verification and Analysis Using Colored Petri Nets (2004)
4. Brook, C.: Nuclear Power Plant Disrupted by Cyber Attack. https://threatpost.com/nuclear-power-plant-disrupted-by-cyber-attack/121216/ (2016), (Accessed April 26, 2017)
5. Buterin, Vitalik: Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform. URL: https://github.com/ethereum/wiki/wiki/White-Paper (2014), (Accessed March 13, 2017)
6. Cam-Winget, N., Housley, R., Wagner, D., Walker, J.: Security Flaws in 802.11 Data Link Protocols. Communications of the ACM 46(5), 35–39 (2003)
7. Carlsen, U.: Cryptographic Protocol Flaws: Know Your Enemy. In: Computer Security Foundations Workshop VII, 1994. CSFW 7. Proceedings. pp. 192–200. IEEE (1994)
8. Cohn-Gordon, K., Cremers, C., Dowling, B., Garratt, L., Stebila, D.: A Formal Security Analysis of the Signal Messaging Protocol. URL: https://eprint.iacr.org/2016/1013.pdf, (Accessed April 26, 2017)
9. Crazzolara, F., Winskel, G.: Events in Security Protocols. In: Proceedings of the 8th ACM conference on Computer and Communications Security. pp. 96–105. ACM (2001)
10. Dai, P., Mahi, N., Earls, J., Norta, A.: Smart-Contract Value-Transfer Protocols on a Distributed Mobile Application Platform. URL: https://qtum.org/uploads/files/cf6d69348ca50dd985b60425ccf282f3.pdf (2017), (Accessed March 13, 2017)
11. DeLoach, S.A.: Analysis and Design using MaSE and agentTool. Tech. rep., DTIC Document (2001)
12. Dresp, W.: Security Analysis of the Secure Authentication Protocol by Means of Coloured Petri Nets. In: IFIP International Conference on Communications and Multimedia Security. pp. 230–239. Springer (2005)
13. Edwards, K.: Cryptographic Protocol Specification and Analysis Using Coloured Petri Nets and Java. Ph.D. thesis, Queen's University Kingston (1999)
14. Fábrega, F.J.T., Herzog, J.C., Guttman, J.D.: Strand Spaces: Why is a Security Protocol Correct? In: Security and Privacy, 1998. Proceedings. 1998 IEEE Symposium on. pp. 160–171. IEEE (1998)
15. Giorgini, P., Mylopoulos, J., Sebastiani, R.: Goal-oriented Requirements Analysis and Reasoning in the Tropos Methodology. Engineering Applications of Artificial Intelligence 18(2), 159–171 (2005)
16. Hoare, C.A.R.: Communicating Sequential Processes. In: The origin of concurrent programming, pp. 413–443. Springer (1978)
17. Jensen, K.: Coloured Petri Nets. In: Discrete Event Systems: A New Challenge for Intelligent Control Systems, IEE Colloquium on. pp. 5–1. IET (1993)
18. Jensen, K., Kristensen, L.M., Wells, L.: Coloured Petri Nets and CPN Tools for Modelling and Validation of Concurrent Systems. International Journal on Software Tools for Technology Transfer 9(3-4), 213–254 (2007)

19. Leiding, B.: Securing the Authcoin Protocol Using Security Risk-oriented Patterns. Master's thesis, University of Göttingen, Germany, DE (2017)
20. Leiding, B., Cap, C.H., Mundt, T., Rashidibajgan, S.: Authcoin: Validation and Authentication in Decentralized Networks. In: The 10th Mediterranean Conference on Information Systems - MCIS 2016. Cyprus, CY (September 2016)
21. Mahunnah, M., Norta, A., Ma, L., Taveter, K.: Heuristics for Designing and Evaluating Socio-technical Agent-Oriented Behaviour Models with Coloured Petri Nets. In: Computer Software and Applications Conference Workshops (COMPSACW), 2014 IEEE 38th International. pp. 438–443. IEEE (2014)
22. Milner, R., Parrow, J., Walker, D.: A Calculus of Mobile Processes, I. Information and Computation 100(1), 1–40 (1992)
23. Moulin, B., Cloutier, L.: Soft computing. chap. Collaborative Work Based on Multiagent Architectures: A Methodological Perspective, pp. 261–296. Prentice-Hall, Inc., Upper Saddle River, NJ, USA (1994)
24. Moulin, B., Brassard, M.: A Scenario-based Design Method and an Environment for the Development of Multiagent Systems. In: Australian Workshop on Distributed Artificial Intelligence. pp. 216–232. Springer (1995)
25. Nakamoto, S.: Bitcoin: A Peer-to-Peer Electronic Cash System. URL: https://bitcoin.org/bitcoin.pdf (2008), (Accessed April 26, 2017)
26. Norta, A., CINCO, C., Computing, I.: Safeguarding Trusted eBusiness Transactions of Lifecycles for Cross-Enterprise Collaboration. University of Helsinki, Department of Computer Science, Helsinki, Finland, Tech. Rep. C-2012-1 (2012)
27. Padgham, L., Winikoff, M.: Prometheus: A Methodology for Developing Intelligent Agents. In: International Workshop on Agent-Oriented Software Engineering. pp. 174–185. Springer (2002)
28. Petri, C.A.: Kommunikation mit Automaten. Ph.D. thesis, Technical University of Darmstadt (1962)
29. Sornkhom, P., Permpoontanalarp, Y.: Security Analysis of Micali's Fair Contract Signing Protocol by Using Coloured Petri Nets. In: 2008 Ninth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing (2008)
30. Sterling, L., Taveter, K.: The Art of Agent-oriented Modeling. MIT Press (2009)
31. Stubblefield, A., Ioannidis, J., Rubin, A.D.: A Key Recovery Attack on the 802.11b Wired Equivalent Privacy Protocol (WEP). ACM Transactions on Information and System Security (TISSEC) 7(2), 319–332 (2004)
32. Vanek, T., Rohlik, M.: Model of DoS Resistant Broadcast Authentication Protocol in Colored Petri Net Environment. In: Proc. 17th Int. Conf. Systems, Signals and Image Processing,(IWSSIP 2010). pp. 264–267 (2010)
33. Vaudenay, S.: Security Flaws Induced by CBC Padding - Applications to SSL, IPSEC, WTLS... In: International Conference on the Theory and Applications of Cryptographic Techniques. pp. 534–545. Springer (2002)
34. Wood, G.: Ethereum: A Secure Decrentralized Generalised Transaction Ledger. URL: http://gavwood.com/paper.pdf (2014), (Accessed March 13, 2017)
35. Wooldridge, M., Jennings, N.R., Kinny, D.: The Gaia Methodology for Agent-oriented Analysis and Design. Autonomous Agents and Multi-agent Systems 3(3), 285–312 (2000)
36. Xu, Y., Xie, X.: Modeling and Analysis of Security Protocols Using Colored Petri Nets. JCP 6(1), 19–27 (2011)