University of Zurich[UZH]

BACHELOR THESIS – Communication Systems Group, Prof. Dr. Burkhard Stiller

A Web-Based Interface for a Blockchain-based Cyber Insurance Approach

Imami, Florian
Zurich, Switzerland
Student ID: 15-731-276

Supervisor: Muriel Franco, Eder Scheid
Date of Submission: August 26, 2021

University of Zurich
Department of Informatics (IFI)
Binzmühlestrasse 14, CH-8050 Zürich, Switzerland

ifi

Bachelor Thesis
Communication Systems Group (CSG)
Department of Informatics (IFI)
University of Zurich
Binzmühlestrasse 14, CH-8050 Zürich, Switzerland
URL: http://www.csg.uzh.ch/

Abstract

Cyber incidents are increasing in numbers and are becoming a problem that can not easily be ignored by companies that rely on technology to operate. Even with high cyber security investments a residual risk remains, mostly due to the dynamic and evolving nature of software and new attack techniques coming up. Hence, to mitigate the costs of cyber attacks, Cyber Insurance has been in discussion for a while now. Besides that, blockchain-based Cyber Insurance approaches are receiving increasing attention as the technology brings several advantages along the way, for example immutability. One of such systems is the recently proposed SC4CyberInsurance which deploys Cyber Insurance agreements as Smart Contracts. However, the system is yet in a state where provided functionalities are not executable in an intuitive and simple manner. This thesis therefore aims to improve the feasibility when it comes to addressing real-world scenarios and the ability to use the system interactively as a tool between insurers and customers, by designing and developing Web-based Interfaces and improving the underlying processes. The core functions are adapted and upgraded to meet stakeholder requirements.

Cyber Angriffe geschehen zunehmends häufiger mit Erfolg, und entwickeln sich zu einer immer grösseren Bedrohung für Unternehmen, dessen Geschäftsprozesse auf Technologien gestützt sind. Auch mit hohen Investitionen in Cyber Security laufen diese Gefahr, nicht jede bestehende Sicherheitslücke schliessen zu können. Dies beruht vor allem auf dem stetigen Wandel von Software und auf neuen Angriffsmethoden. Auf der Suche nach Möglichkeiten, um sich vor den resultierenden Kosten erfolgreicher Cyber Angriffe zu schützen, wird das Augenmerk vermehrt auf Cyber Versicherungen gerichtet. Unterdessen erhalten auch auf Blockchain basierende Cyber Versicherungslösungen immer mehr Aufmerksamkeit, da diese zusätzliche Vorteile, wie Schutz vor Vertragsmanipulationen, mit sich bringen. Eine dieser Cyber Versicherungslösungen ist das kürzlich entwickelte SC4CyberInsurance; ein System, welches basierend auf den Vereinbarungen zwischen Kunde und Versicherungsunternehmen einen Smart Contract erstellt. Die Handhabung des ursprünglich entwickelten Systems gestaltet sich komplex und die Interaktionen sind nur schwierig ausführbar. Ziel dieser Bachelorarbeit ist es, das ursprüngliche System zu verbessern und zu erweitern, so dass es für realitätsnahe Fälle einsetzbar wäre, indem es eine funktionierende Schnittstelle zwischen Versicherungsunternehmen und Kunden darstellt. Dies wird erreicht, indem ein web-basiertes Interface für jede einzelne Nutzergruppe gestaltet und entwickelt wird, und bestehende Funktionalitäten verbessert werden. Die Funktionen des Systems, wie zum Beispiel Verhandlungen von Schadensfällen, werden für das Interface angepasst, so dass die Anforderungen der Anspruchsgruppen erfüllt werden.

Acknowledgments

First and foremost, I would like to express my sincere gratitude to my supervisor, Muriel Franco, for his continuous support, interesting discussions, and instructive and insightful guidance in the process of this thesis. It has been a really great pleasure to work with someone as clever and empathetic as Muriel.

Also, I would like to thank my co-supervisor, Eder Scheid, for his inputs and support.

Finally, I would also like to sincerely thank Prof. Dr. Burkhard Stiller, head of the Communication System Research Group (CSG) at the University of Zurich, for giving me the opportunity to write my bachelor's thesis about such an interesting topic.

Contents

Chapter 1

Introduction

Cybersecurity has become a concern that cannot easily be ignored by companies whose services rely on technological infrastructure and digital connectivity. Criminal activities involving business networks and governments as a target are becoming more frequent as the digital world is evolving. Technological innovations and the digitalization lead, beside indisputable benefits, to more attack surface for cybercriminals [10]. According to a recently published article [1], the global damage caused yearly by cyberattacks is illustrating a rising trend and it is predicted that it will achieve a total of 6 trillion US$ in 2021. Analysts are also expecting an average increase of 15 percent annually in the next five years. Subsequently, global damage caused by cybercrime will be one of the highest business risks existing and would even exceed the profit of international drug trafficking. In a publication of the World Economic Forum [5], cybercrime is considered to be one of the highest global business risks.

The risk of being successfully targeted by cyberattacks can be reduced by investments [38, 39] in cybersecurity, however, the evolving technology makes it difficult for companies to adapt to the newest attack techniques used by cybercriminals [10]. To recover from the damage inflicted by this kind of attacks, Cyber Insurance represents a possibility of transferring these risk [7]. As stated from N. Kshetri [8], Cyber Insurance is still in its infancy. In the same publication it was also mentioned that Cyber Insurance encounters numerous challenges regarding the risk, cost estimation and the definition of coverage. Nevertheless, Cyber Insurance is growing continuously and fast. The total spending on Cyber Insurance is going up and is expected to reach 20 billion US$ by 2025 [9]. Standard & Poor's Corp. predicts a growth of premiums by 20% to 30% annually [6]. In order to provide more efficient services, insurance companies can take the opportunity to involve modern technologies in their processes [11]. However, the integration of blockchain and Smart Contracts (SC) has also been discussed in recent research publications [4]. It was stated, that these relatively young technologies have potential to provide benefits when integrated in the Cyber Insurance sector. A recent example for one blockchain-based system for Cyber Insurance would be BlockCIS [12].

1.1 Motivation

The Cyber Insurance sector is continuously growing and receives more attention as cyberattacks are becoming a frequent occurrence [1][10]. As the Cyber Insurance sector is in the process of evolving and establishing, it is experiencing changes due to new technologies such as blockchain and SCs [4]. The involvement of these technologies may bring several significant advantages. This thesis focuses on the system introduced in SC4CyberInsurance [13]. As a blockchain-based Cyber Insurance system, SC4CyberInsurance introduces the creation and deployment of insurance contracts as a SC, and therefore integrates the benefits of using blockchain technology into Cyber Insurance agreements. The SC is able to run payments and acts as an intermediate between the involved actors, rendering third parties unnecessary. The SC4CyberInsurance model provides a prototype with functionalities that support the creation, management and interaction between the stakeholders and the SC. However, the existing prototype misses a visual interface that would make the execution of all provided functions and management of data more presentable and usable. With the implementation of Web-based Interfaces for all actors involved and modifications to adapt real-world scenarios, the usage of this tool can be simplified as well as the user experience can be improved.

1.2 Description of Work

The goal of this thesis is the development of a Web-based interface for the blockchain-based Cyber Insurance model in [13]. For evaluating the user interface, the provided model has to be used to address real-world-based scenarios in the form of case studies. Both, customers and insurance companies, should be able to interact with the platform to define contract agreements, coverages and execute all functions that are part of the SC4CyberInsurance model. The platform should guarantee the deployment of trustworthy and automated contracts on the blockchain.

To achieve the thesis goal, the work is divided into several steps. The first, initial step consists of an extensive literature review regarding Cyber Insurance and blockchain-based Cyber Insurance approaches in order to achieve the necessary theoretical background. Research for work related to the thesis topic is conducted to illustrate the state of the art and show characteristics of various related systems. Furthermore, knowledge of the technical aspects related to the implementation of the SC4CyberInsurance solution [13] has to be acquired.

The second step includes the definition of all actors involved in the SC4CyberInsurance model as well as the definition of all interactions. Use cases have to be illustrated in order to develop in a front-end solution, as a basis for designing the system's Web-interface layout and functions. To define how to configure and store important aspects of the contract agreements, a data structure needs to be specified.

After defining the solution, the technologies that are to be used for the prototype need to be selected. The existing SC4CyberInsurance prototype has to be modified and improved,

in order to be thoroughly integrated with the developed front-end. Thus, the data flow has to be ensured, such that the users of the platform can create and execute the SC functions in an intuitive manner. Involved actors should be able to define, send or request information that is part of the blockchain-based Cyber Insurance model. To communicate with the deployed insurance contract in the form of a SC, the front-end should use the provided scripts.

In a last step, the developed solution is evaluated and several case studies are conducted to show the feasibility and completeness of the implemented Web-interfaces. The case studies include scenarios that simulate real-world demands and use cases.

1.3 Thesis Outline

The rest of this thesis is organized as follows. Chapter 2 provides a theoretical background of the topics covered in this thesis is provided. This includes a brief analysis of Cyber Insurance and its core concepts. In Chapter 3 the provided SC4CyberInsurance prototype of [13] is thoroughly explained and illustrated. Also, two related systems to this topic are analyzed and characterized. In Chapter 4 the approach of developing the Web-based Interface for a modified SC4CyberInsurance system is specified. Chapter 5 presents the technical details of the implementation. Evaluations on the feasibility of the developed solution are executed in Chapter 6. In the end, Chapter 7 concludes this thesis with a brief summary and future work.

Chapter 2

Background

The following chapter provides an overview of the theoretical background this thesis is based on. Starting with an overview of Cyber Insurance in Section 2.1 and breaking it down into its underlying processes and core concepts, the reader will be introduced to the topic. Also, the challenges this type of insurance faces are mentioned and premium estimation in Cyber Insurance will be discussed. The technologies blockchain and Smart Contracts are briefly explained in Section 2.2.

2.1 Cyber Insurance

Compared to traditional insurance types, Cyber Insurance has its peculiarities. Companies can purchase this relatively new type of insurance to transfer their cyber residual risk [20]. Even with significant investments in IT security, companies can not entirely exclude the existence of residual risk. The reasons are, among others, that even though the protection techniques against cyberattacks are improving, the attack techniques used by cybercriminals are evolving as well [10]. As a result, we can observe an upwards trend in the relatively young sector of Cyber Insurance [9].

2.1.1 Cyber Insurance Process

The Cyber Insurance process involves two parties, an underwriter and a policyholder, who is demanding the insurance product. Fig. 2.1 visualizes the Cyber Insurance process and its different action flows. The following illustration of a Cyber Insurance process is based on the publication of Dambra et al. and their explanation of such a process [14].

In the underwriting phase, similarities to the traditional insurance process can be observed. The insurer collects information in order to draw the client's risk exposure, therefore identifying the assets to be insured, the threats, and existing vulnerabilities in the IT system [17]. In Cyber Insurance there are several challenges, such as the existence of asymmetric information [20]. These challenges will be discussed further and in more

detail in Section 2.1.4. After the risk identification, the likelihood and impact of possible incidents are calculated. In the end, the risk survey should result in an illustration of the threats, internal and external, that the company is confronted with. When both parties agree on the contract specifications, an insurance contract is established that consists of the coverage definition and the premium calculated. The pricing of the premium is, in contrast to traditional insurance, difficult to be assessed accurately by using actuarial data due to of the lack of data in the first place. Companies often try to avoid the public disclosure of attacks because of negative public reactions. Also, the dynamic character of cyberattacks [15] reduces the benefits of following a strict actuarial pricing approach.

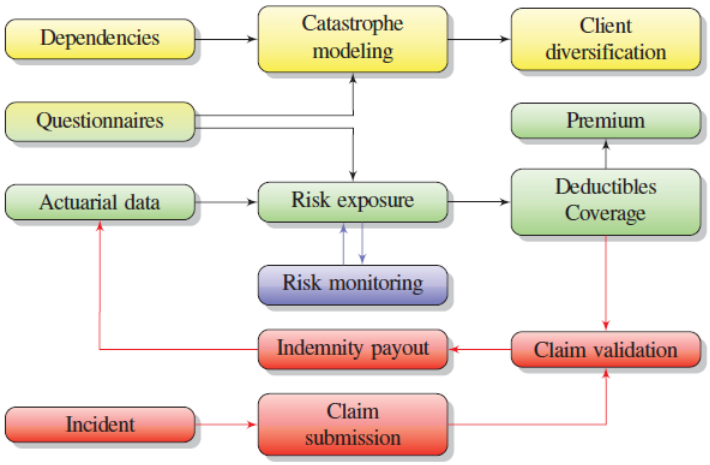

Figure 2.1: Classic Insurance Process Workflow extended in a Cyber Scenario (Yellow: Portfolio Management, Green: Underwriting, Blue: Post binding, Red: Claiming) [14]

The portfolio management phase encompasses the whole insurance contract life cycle. In contrast to traditional insurance processes, the independent probability of claims is difficult to achieve in Cyber Insurance due to the networking of companies. The goal for the insurer in this phase is to manage the portfolio of insurance contract in a way that prevents claims for the same incident to be submitted by multiple policyholders at the same time. Regarding the insurance of cyber risks, achieving this goal is made more difficult by the limited number of different software and hardware products [16]. The usage of external services expands the attack surface of companies and therefore increases the risk exposure [14]. Also, software products are usually utilized by a large number of operators and open vulnerabilities can thus lead to large-scale cyberattacks [22]. Reinsurance as an opportunity for Cyber Insurers to protect themselves from these overarching threats is yet not provided sufficiently, as the Cyber Insurance sector is still in its nascent stage.

A significant peculiarity of Cyber Insurance is the existence of a post binding phase, including periodical risk assessments as a so called risk monitoring. This phase targets to solve the problems Cyber Insurance is facing. Where traditional insurers end the direct interaction with the other party after the underwriting phase has been completed, Cyber Insurance underwriters may periodically interact with the customer over time to

adapt the contract. The risk exposure of companies may change over time, due to the dynamic character of the cyber domain. This changes the risk exposure of customers that has been collected in an initial step. To solve this problem, the risk assessment may be repeatedly updated to adjust the risk exposure defined at an earlier stage. For example when a company invests in cybersecurity to increase protection against cyberattacks and decrease the likelihood of being successfully targeted by one. Therefore, in this scenario, the premium that the customer has to pay can be adjusted downwards after the contract specifications has been updated, because the risk exposure is lower. In the end, both parties profit from periodical risk assessments. It also reduces the risk of moral hazard, where the policyholder behaves in an incautious way, knowing that the underwriter will cover the damages incurred.

In the case of an incident, the insurer has to handle the claim submission. In Section 2.1.2, the coverage commonly specified in Cyber Insurance contracts is listed and described, including insurable and uninsurable risks. Usually, the claim needs to be validated. However, there also exist special occurrences, where the attack is hard to capture because of technical infrastructure.

2.1.2 Coverage

The contractual agreements, which are defined during the underwriting process, lead to a protection of the insured against specified cyber risks. Cyber Insurance generally differentiates between two categories of costs, namely first-party and third-party costs [8]. First-party costs are a direct result from the incident, whereas third-party costs are claims made from external parties that suffer a loss resulting from the cyberattack executed on the insured party [18]. An example for third-party costs is the accusation of the insured company that was targeted, because of third-party data public disclosure [21][23]. While first party costs can be simply defined as costs that directly affect the business of the insured party, third party costs can be characterized as liability insurance. Cyberattacks can for example spread via the targeted business network to other companies and clients in the same network. Third party insurance focuses on minimizing these costs.

According to Petratos et al. [20], cyber risks can be divided into the categories "*insurable*", and "*uninsurable, or only insurable with constraints*". The separation is shown in the following.

Insurable Cyber Risks

- **Privacy events:** Companies with privacy or information risks can insure the customer and employee information in cases where data breaches publicly disclose sensitive data [2].
- **Crime and fraud:** A very common policy in the Cyber Insurance domain is the insurance against cyberattacks and fraud. Also, professional advice on preventing these kind of incidents may be included here.

- **Network security liability:** The coverage of the network security is one of the most important policies for companies [2]. This category includes incidents that lead to a network security failure. An accurate and continuous risk assessment increases the efficiency of mitigation strategies, as well as cost estimation.
- **Software and data damage:** Also, the damage on company data and software is insurable. In order to prevent worst case scenarios, insurers can require the policyholder to perform data backups on a regular basis.

Uninsurable, or only insurable with constraints

- **Reputational loss:** It is possible to insure losses related to public reputation damage, although the costs for this kind of loss are difficult to estimate for insurance companies.
- **Network business interruption:** Denial-of-Service (DoS) attacks lead to network interruption. When multiple customers are impacted by one DoS attack, for example due to their business networks being connected to each other, the interdependent probability of claims is critical. The consideration of this worst-case scenario leads to difficulties in estimating the costs.
- **IP theft or espionage:** As company secrets are of utmost importance for companies, and therefore priceless, the value of a company secret is entirely lost if publicly disclosed due to a cyber incident. These kind of incidents often occur in relation to crimes between different states.
- **Physical asset damage:** The interconnection between the physical world and the cyber domain is yet not well understood by insurers.
- **Death and bodily injury:** Cyber incidents may also lead to physical harm, e.g. when hospitals or other medical environments are targeted. The continuous functionality of medical devices is obviously important for the life and health of the patients, and therefore incidents may lead to serious consequences. Contrary to Cyber Insurance, where this specific kind of policy is not yet entirely adopted, the traditional insurance sector offers insurance products for this kind of incidents.

2.1.3 Premium Estimation

In insurance it is generally important to calculate a fair premium. Unfair premiums reduce the incentive of the potential customer. Only a fair premium will lead to a contract agreement between two parties involved. In order to calculate the premium in Cyber Insurance, the risk assessment has to be executed properly, as mentioned in Section 2.1. To counter unreasonably high priced premiums, state regulations were created. However, as cyber incident related actuarial data is not sufficiently available in order to estimate loss expectations accurately, insurance companies have to consider different methods for pricing their products. The research of Romanosky et al. [18] discussed several methods for pricing Cyber Insurance premium by empirically analyzing several US Cyber Insurance policies. These approaches will be explained in the following.

Flat rate pricing approach

This is one of the simplest methods observed. The price for first and third party coverage is fixed for all insured entities. Mostly premiums of small companies are calculated using the flat rate approach. It is based entirely on the estimated frequency of incidents a company is expected to experience during the contract lifetime, and the loss expected. Other factors, such as the size of the company and the yearly revenue, have been excluded. Also, a variation of this method has been observed, which is extended by a single factor: hazard groups. A hazard group is assigned according to the business type of the company insured. Usually companies whose core services highly rely on technological infrastructure and/or sensitive data are assigned to a higher hazard group.

Base rate pricing approach

The base rate pricing approach includes the yearly revenues or assets of policyholders to estimate the premium. The industry sector of the customer is also an important factor, which is additionally included in this approach. First, this calculation process uses the revenue or assets of the company to estimate the base premium. Second, the standard insurance factors are involved. These factors simply consist of the maximal indemnity and deductible. Also, the duration of the contract and the personal attack history are involved in this premium estimation approach. The third group of factors used in this approach are part of the industry classification. However, insurance policies analyzed by Romanosky et al. [18] were not systematically showing any consensus on what industries are the ones with most risk included. So whether or not the industry is involved as a factor for the premium estimation, and what industry is considered to be the riskiest one, is up to the insurer.

The most observed approach by Romanosky et al. [18] is a variation of the base rate pricing approach, which is called ”base rate with security questions”. Here, the company security is assessed and involved in the premium calculation. The security level of companies define the adjustments to the premium calculated. It is considered to be the approach with the best trade off between complexity and practical usability compared to other methods observed. Ranking the system security of the insured company regarding different categories results in a more accurate premium estimation, as well as in a reusable instrument to be used on with future customers.

2.1.4 Challenges

There are several challenges Cyber Insurance is facing currently. Most of the mentioned challenges originate from the dynamic character of the cyber domain and the newness of Cyber Insurance as insurance type. The first group of challenges include externality effects which have a big impact on this specific type of insurance. Externality effects discourage companies to purchase Cyber Insurance [8]. Interconnectivity in the cyber domain creates a high amount of externalities due to the interaction between several heterogeneous players in the same ecosystem [20]. The security system of one company often depends on others.

For example, if one policyholder insures herself against a specific type of cyber risk, other companies connected to the network of the policyholder will automatically be insured also. Since an attack would directly damage the insured company in the first place, the companies in the same network will probably also experience a negative impact due to the interconnectivity that exist [3]. This would lead to the "Free-Riding problem" [17] that reduces the incentive for companies to invest in their security or to purchase Cyber Insurance products, as it seems to be sufficient to insure only one node in the same network.

The Cyber Insurance market is also struggling with the lack of prior experience and standardization [8]. As an insurance buyer, it is important to understand the cyber risks the company is facing in order to find the appropriate type of insurance and coverage. Without a sufficient amount of knowledge, the coverage is difficult to be clearly defined. Additionally, the continuous evolution of IT systems makes it hard for companies and insurers to keep up with new emerging cyber risks and new attack techniques [17]. In [8] it is also mentioned that the value chain of Cyber Insurance is yet not well developed and that insurance brokers, as well as insurance agents, often lack of knowledge and understanding.

Insurers have uncertainties in pricing the premiums and estimating the costs. Data on cyberattacks and related losses is relatively new and very scarce. In [8] it is mentioned that there exists a lack of statistical data on cyber incidents, resulting in difficulties for specifying insurance policies accurately. A possible reason for that may be the habit of companies to keep cyber incidents private and not to publish them, in order to avoid reputation damage and lawsuits [17]. To tackle that problem, regulations have been created. However, there is also no complete database of cyber incidents. For example, a recently analyzed database of breaches consisted of approximately 1000 entries, recorded between the years of 1971 and 2009 [19]. Other factors that lead to difficulties in premium pricing are insuring risks that correlate with the risks of other insureds. Especially in the cyber space this is a serious problem for the insurance company. As there is still a lack of re-insurance [20], large-scale attacks can lead to serious losses for the insurance company. Re-insurance is a solution to the problem of interdependent probability of claims, however, the small number of re-insurers is growing.

2.2 Blockchain and Smart Contracts

This thesis is based on the recently developed SC4CyberInsurance system [13], a blockchain-based Cyber Insurance approach that uses blockchain technology for the creation and maintenance of Cyber Insurance contracts. The insurance contract is deployed as a SC. In order to understand the functionality and the additional benefits of using blockchain technology for Cyber Insurance, the following Section characterizes the technology.

2.2.1 Blockchain Technology

The idea behind blockchain, according to the whitepaper of Bitcoin [26], was to create a fully peer-to-peer electronic payment system that reduces the need for a trusted intermediary between the two parties participating in a payment. Therefore, the exchange of currency would not require to be monitored by financial institutions or other third parties. Instead of having to trust a third-party, blockchain introduces a cryptographic proof [40].

The so called blocks that store information about transactions are linked together. These transactions include information about a cryptocurrency transfer, or transaction, between two parties [4]. A blockchain usually consists of multiple blocks and is extended with additional transactions by adding a new block to the end of the chain. Therefore, the blockchain acts as a ledger that consists of the transaction history [27]. Every block in the chain points to the previous block, by referencing to the cryptographic hash value that was calculated on the "parent" block [28]. Also, a hash value is computed on every new block so that each block will have its own unique identifier.

Instead of using a centralized database to store the transaction data contained in all blocks, the blockchain is stored decentralized and redundantly at the network nodes. Each of these nodes stores a copy of the entire blockchain, as explained in [4]. The publication also illustrates that nodes can not only passively store a copy of the blockchain, but also take action by mining for a new block. Mining is an important part of the verification process. So called mining nodes, also called miners, continuously check if the amount that one party wants to spend in a transaction is valid and they also solve a computational problem to add new blocks. In case of Bitcoin [26] this computational problem is called the Proof-of-Work (PoW). The process of adding a new block will be explained later in this Section. The computational problem makes manipulation of the blockchain almost impossible since a specific computational power is needed to solve it. Therefore, a malicious subject must have control over more than half the amount of all network nodes. Furthermore, due to the fact that all blocks apart from the very first block reference to the hash of the previous block, manipulation of the chain is extremely hindered, since the manipulator must then also modify all the blocks that were added after the targeted block. In order to create an incentive for nodes to be take the role of a miner, they receive a monetary reward for successfully solving the computational problem and adding a new block to the chain.

Taking Bitcoin as an example, the transfer of coins is explained in the following [26]. A transaction is created and executed by nodes of the blockchain network [29]. A transfer of a coin for example is executed by digitally signing the transaction with the private key of the sender. This guarantees that the sender can be verified with its public key. New transactions are added to the end of the existing blockchain.

When a new transaction has been executed and needs to be added at the end of the blockchain, it is broadcast to all network nodes [26]. The nodes add this new transaction to their stored blockchain duplicate as a block. Also, they start solving the computational algorithmic problem for this block and the first node to solve the algorithmic problem broadcasts the validated block again to all other nodes in the network. The new block is only accepted by all network nodes if the amount that is transferred by that specific sender is valid and does not exceed its existing balance. Nodes that accept the new block, and

therefore entirely accept the new transaction, start working on extending the blockchain by beginning the process again from the beginning. If other nodes were already working on another, earlier block, they stop working on the old block and start creating a new one. However, nodes that do not accept the new block continue working on another, older block. That would lead to more than one blockchain being processed at the same time. In that case, the longest blockchain, therefore the one with the most blocks, is considered to be the valid one. The longest chain is usually processed using the most computational power worked up by the majority of nodes which leads the network to be protected against malicious subjects.

The key characteristics of blockchain are explained in the following list [28]:

- **Decentralization:** There is no need for an intermediary trusted party to validate the executed transactions. This leads to a reduction of the server costs and performance bottlenecks compared to traditional centralized systems.
- **Persistency:** Manipulating the transactions that had been recorded and stored in the blockchain is extremely hindered. Every new transaction needs to be accepted by most of the other network nodes that are continuously checking the validity.
- **Anonymity:** A specific address is assigned to every user in the blockchain network. However, a user is not limited to having one address. For example, users can use multiple addresses to interact with the blockchain, in order to up their identity. As private user information is not stored and controlled by any trusted intermediary, a certain amount of anonymity is provided. Nevertheless, an absolute anonymity is yet hard to achieve.
- **Auditability:** Every transaction that is recorded in the blockchain includes a timestamp and has to be validated first. Therefore, it is possible to track transactions in the records. This leads to transparency of the data stored in the blockchain.

2.2.2 Smart Contracts

Blockchain technology can not only be used to store transactions, but also to provide executable code [4]. Program code can be stored in the blockchain as a so called Smart Contract (SC). SCs are self-executing contracts which contain defined contract agreements between two parties that do not need a trusted intermediary to be carried out. The functions and characteristics of a SC are programmed and therefore defined. The idea of SC originally was created by N. Szabo in 1997 [30]. It was meant to be a computerized transaction protocol that is able to automatically execute the terms of a contract between two parties. Certain events will trigger the execution of functions. For example, when a condition is met, currency is send to the according party automatically [28].

The most popular blockchain that supports SCs is Ethereum. It provides a turing-complete machine, which is decentralized, called the Ethereum Virtual Machine (EVM) [31]. This enables the execution of SC code. The currency used for transactions in this case is called ether. In order to execute transactions, for example to update the existing contract agreements that have been specified initially, fees have to be paid [32].

Chapter 3

Related Work

This thesis is based on the SC4CyberInsurance prototype [13]. In the following chapter, the developed prototype will be analyzed and explained. The core functions as well as the key characteristics of SC4CyberInsurance will be briefly discussed, so that the reader is provided with an adequate comprehension of the system. This Chapter further illustrates two different related systems where blockchain was integrated in the context of Cyber Insurance and Cybersecurity.

3.1 SC4CyberInsurance

SC4CyberInsurance is a blockchain-based Cyber Insurance approach that has been developed by Noah Berni [13]. As blockchain is considered to be a promising technology for the insurance sector [4], this approach introduces a system that creates and deploys SCs for Cyber Insurance. Due to the definition of the contract agreements in form of a SC, the agreements are immutable and transparent, leading to a trustworthy contract agreement between customer and insurer. This system integrates several advantages of blockchain technology while focusing on Cyber Insurance use cases. The customer and insurer can both interact with the system to create and deploy a Cyber Insurance contract by using its core functions provided. These functions include the creation of customized Cyber Insurance contracts, the update of contract agreements initially defined, the ability to pay premiums and securities and the possibility to report claims. They will be explained in more detail later in this Section. The payments are executed via the SC and therefore the transferred currency is ether.

There are several benefits resulting of the combination of blockchain and SCs. The system [13] leads to a simplified way of creating Cyber Insurance contracts by reducing the complexity and automatizing several typical tasks. The usage of blockchain leads to a trustworthy and immutable agreement definition. By anonymizing data that should not be publicly available on the blockchain, no private business data is published on the blockchain [13]. The SC4CyberInsurance architecture of the provided prototype is visualized in Fig. 3.1.

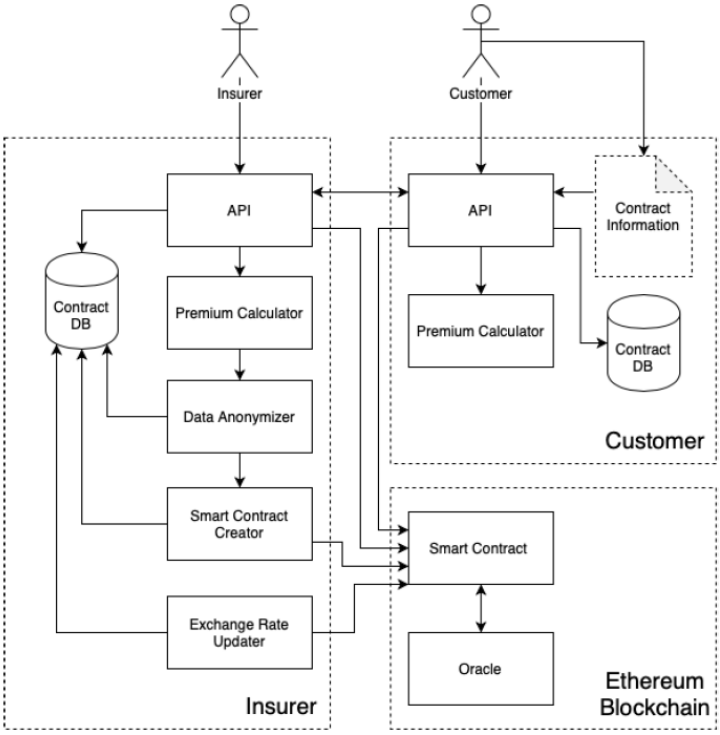

Figure 3.1: System Architecture of the Original SC4CyberInsurance Prototype [13]

In the original SC4CyberInsurance architecture [13], both actors interact with the system via an API. The customer has to enter the necessary information to be able to create a Cyber Insurance contract. In the original system, this was done by filling out a predefined file and sending it to the system using an API. The customer can also let the system compute the according premium for the specified contract agreements. When the customer wants to create a contract by using the predefined file, the insurer side automatically processes the forwarded contract agreements. Private customer information is anonymized and stored on the blockchain with the contract specifications. Both parties store the entire contract information including private business data in separate databases. A SC is then deployed on the Ethereum blockchain and its address is also stored in both databases. The contract has a lifetime which is lengthened when the customer pays the premium. During the lifetime of a contract the customer can access all the necessary functions the system provides. However, before the customer can interact with other functions of the system, a security must be paid to validate the contract. This protects the insurer from the scenario in which the customer cancels the contract by not paying the premium anymore. The security has to be paid after the contract has been created and before the customer can execute other contract functions. In the end of the contract lifetime, the security gets refunded. However, if a customer does not pay the premium anymore before the contract duration is over, the security is kept by the insurer. The premium is transferred from the account of the customer to the SC and the currency is ether.

In the following, the core functions of SC4CyberInsurance [13] are briefly explained. Note that these functions will be modified and upgraded in the context of this thesis. The

following explanation should only provide a basic comprehension of the most important core functions that the SC4CyberInsurance prototype of Noah Berni provides.

- **Contract creation and annulation:** In the beginning of each Cyber Insurance contract lifecycle, the customer has to provide all the necessary information. This information includes business information, contract constraints including the duration and payment frequency, the financial conditions of the company as well as technological information about the company. Also, the attacks and impacts to be covered have to be specified. Further, the insurer can terminate a contract if the customer did not pay the premium on time.

- **Handling contract agreement updates:** It is possible for the customer to modify the initially defined contract agreements. These adjustments are important in scenarios where the insured company improved its cybersecurity or when the technologies have changed. Also, coverages can be changed. Update requests will be analyzed by the insurer who can either accept or decline the changes. This leads to a continuous risk assessment.

- **Security/Premium payment:** In order to be able to report damages and cover the cyber residual risk, the insured company has to regularly pay a premium. The payments of premium and security are executed in a simple way; by transferring ether to the SC manually if needed.

- **Claim Reporting and negotiation:** The insured company reports damages to insurer - similar to other insurance types. Here, the customer has to specify the type of attack and the amount to be covered. Also, a logfile must be sent for evidence. The content of the logfile is also anonymized when shared in a damage report, meaning that only the hash of it will be stored in the blockchain. This function provides a certain amount of interaction between the customer and insurer. The insurer can either accept or decline a damage and has the possibility to send a counteroffer to the customer. Therefore, the customer can afterwards accept or decline the counteroffer. In case of an agreement, the payment is triggered automatically and the necessary amount of ether is transferred between the accounts. When there is no agreement, both parties need to resolve that dispute externally.

- **Updating the Exchange Rate:** The exchange rate of ether can be updated automatically. In the provided prototype, the exchange rate is updated after some time has passed and when the rate has changed significantly.

However, the original blockchain-based Cyber Insurance system does not include an intuitive visual interface and is difficult to use for real-world based scenarios. State changes need to be fetched manually and some parameters need to be specified manually in order to execute the functions. Upgrading the provided system by improving the usability and develop a Web-based interface is the goal of this thesis.

3.2 BlockCIS

BlockCIS [12] is a blockchain-based continuous monitoring and processing system that is used in the field of Cyber Insurance. It addresses the challenges Cyber Insurance is confronted with, such as the lack of standardization and actuarial data. Also, it focuses on the problem that the cybersecurity level of companies may change over time, and therefore also the risk exposure. The system includes four different groups of actors: insurers, customers, auditors and third party services. The actors are connected via blockchain and the insurance contract is deployed as a smart contract.

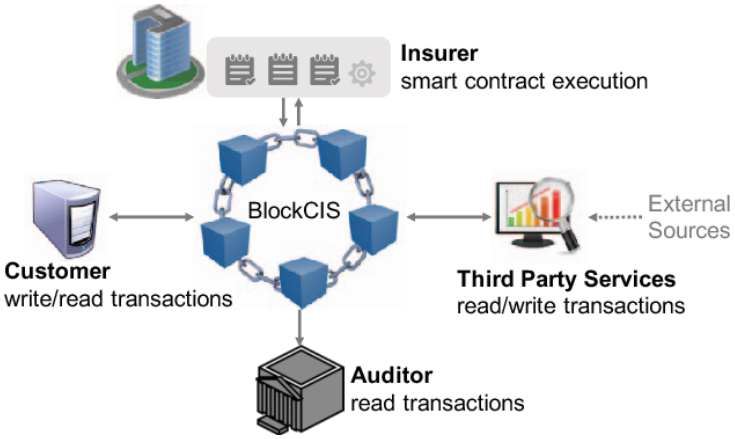

Figure 3.2: BlockCIS Overview. [12]

There is a node deployed for each of the actors involved. At the customer side, the blockchain collects analytical data to assess the cyber risk level continuously. In this manner, the contract information is always up-to-date. In Cyber Insurance, the risk assessment is a very problematic component, as the assessment that has been executed today may not be valid tomorrow. To solve this problem, BlockCIS [12] introduces an automated, real-time immutable feedback cycle. The blockchain supports the dynamic and continuous risk assessment so that the premium can be priced according to the needs of the customer. Like this, the insurer is able to have a precise portrayal of the cyber risks the customer is facing. This leads to tailored premiums and contract coverage. To precisely estimate the likelihood and costs of cyberattacks, third party services can be accessed. Therefore, the integration of third party services supports the overall Cyber Insurance process with useful data. In the case of a dispute between the insurer and customer that cannot be solved among themselves, auditors can be involved. This third party investigates such use cases and helps to resolve disputes. BlockCIS is a supporting tool and does not work as standalone. Also, it does not provide a visual interface to ensure simplified usability. Nevertheless, it successfully tackles the problem that software and cyberattacks have a dynamic character and are evolving over time.

3.3 SECONDO

Another related system which integrates blockchain in cybersecurity and the Cyber Insurance sector is called SECONDO [33]. It is a framework that supports cybersecurity investment decisions and Cyber Insurance pricing. This framework addresses the difficulties that companies are facing in order to estimate the optimal amount of cybersecurity investments. Also, it supports companies to quantify the existing cyber risks. The cyber residual risk is determined with the help of this framework, which also supports the premium estimation based on what strategy the insurer is following. The latter should lead to reduced information asymmetry, as the insurer standard is taken into account.

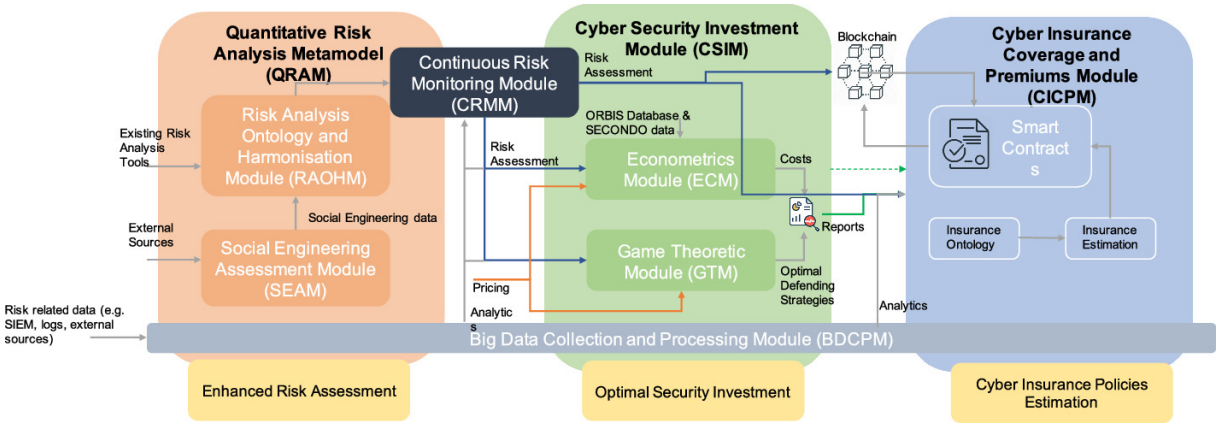

Figure 3.3: Architectural Components and Integrated Modules for SECONDO [33]

Generally, this framework consists of four different components [33]:

1. **Quantitative risk assessment and data analytics:** This component targets the identification of relevant threats a company is facing, the vulnerabilities that the company's assets are confronted with, and the value estimation of the assets that are potential targets of a cyberattack. Also, the likelihood of such an attack is estimated. In the end, the estimated risk is the amount of loss that would occur after an attack on the identified vulnerable assets.

2. **Cybersecurity investments:** The second component uses the data resulting from the risk assessment to support cyber security investment decisions. The Cyber Security Investment Module (CSIM) is responsible for estimating the optimal cyber security investment strategy. It uses several specialized modules that calculate the cost estimates of cyberattacks and security controls. Furthermore, this modules also provides optimal defending strategies.

3. **Continuous risk monitoring and blockchain:** The risk assessment data is stored on a blockchain. The blockchain updates the risk levels dynamically and therefore always the most up-to-date cybersecurity information about the company is stored. The SC is therefore updated regularly also. The insurer and customer will get notified

when the insurance terms have been violated or when events lead to the activation of the contract functions.

4. **Cyber Insurance and smart contracts:** This component supports the assessment of the cyber residual risk. The coverage and premiums are estimated, based on the insurance policy defined by the insurer. Therefore, this component introduces a certain level of standardization due to the specific Cyber Insurance ontology being used. This reduces the information asymmetry which is often a problem in Cyber Insurance sector.

This framework aims for assisting companies cybersecurity investment strategies and Cyber Insurance agreements [33]. Generally, this framework provides efficient methods for assessing the risks and finding optimal investments in cybersecurity by considering a limited budget. Furthermore, the Cyber Insurance contract is here also deployed as a smart contract. It can be observed that there is a rising trend for the usage and integration of blockchain technology in the Cyber Insurance sector.

Chapter 4

Approach

This work provides a significant upgrade to the original SC4CyberInsurance prototype [13], such as the introduction of a visual Web-Interface and an improvement of the feasibility of the core functions provided. The goal of this thesis is to guarantee an intuitive and simplified execution of the SC4CyberInsurance core functions by introducing Web-Interfaces for both user groups: insurers and customers. This chapter explains the solution developed to achieve this goal. The Web-Interface has to ensure proper payment execution, trigger smart contract functions, and continuously provide both parties an overview of the current contract state including existing damage claims. Both parties can use different functionalities to interact with the system, therefore two separate user interfaces have been designed to fulfill this requirement. However, the interfaces will have similar visual features to clarify the connection between them.

Both Web-Interfaces improve the usability of the original SC4CyberInsurance system. To achieve that, the design focuses on simplicity and on interactivity. As there are two group of actors (*i.e.,* insurers and customers) interacting with each other, it had to be guaranteed that the right data is shared at the right time, and that always the most up-to-date contract state is shown for both actors. Also, the functions available have to be able to still be executed properly. The overall design focuses on usability and interactivity with the system's functionalities.

In order to upgrade the existing system to address real-world scenarios, the risk assessment has been improved. The calculation of the premium can recognize that different metrics results have to processed individually. For example, a high result in one metric may show that the customer company is overall confronted with lower cyber risks, while another metric result would show that the company is facing a lot of cyber risks. This improves the processing of the risk assessment, and therefore the premium calculation is adjusted to ensure appropriate pricing results. The extensibility of this solution is provided, so that more current and future risk assessment metrics can be added to the calculation. Also, the contract creation is upgraded, so that the insurer can finally decide whether or not to create a contract.

As this tool is based on blockchain technology, the users should be able to use their own addresses in order to execute SC functions. The assignment of the addresses and

the payment executions are now executed dynamically by introducing the option to log in with the own credentials. Therefore, the usability of this tool is improved and the addresses are not assigned statically anymore. In the rest of this chapter, an overview of the approach and its improvements is provided as well as the different components and features discussed.

4.1 Overview

In a first step, the architecture of the existing SC4CyberInsurance prototype [13] had to be adjusted. In the original architecture both parties had to access the core functions of the prototype via their APIs. The new architecture introduces Web-Interfaces that make the execution of the function easier and more intuitive. The events triggered by the Web-Interface also differ from the original prototype, so that the feasibility is improved and the solution leads to an overall improvement of the functionality. The new adapted architecture is shown in Figure 4.1. This architecture illustrates the system from a customer's perspective. However, both actors will have their own separate interfaces to interact with the system, such that the insurer will be able to also use an interface to access the contract functions.

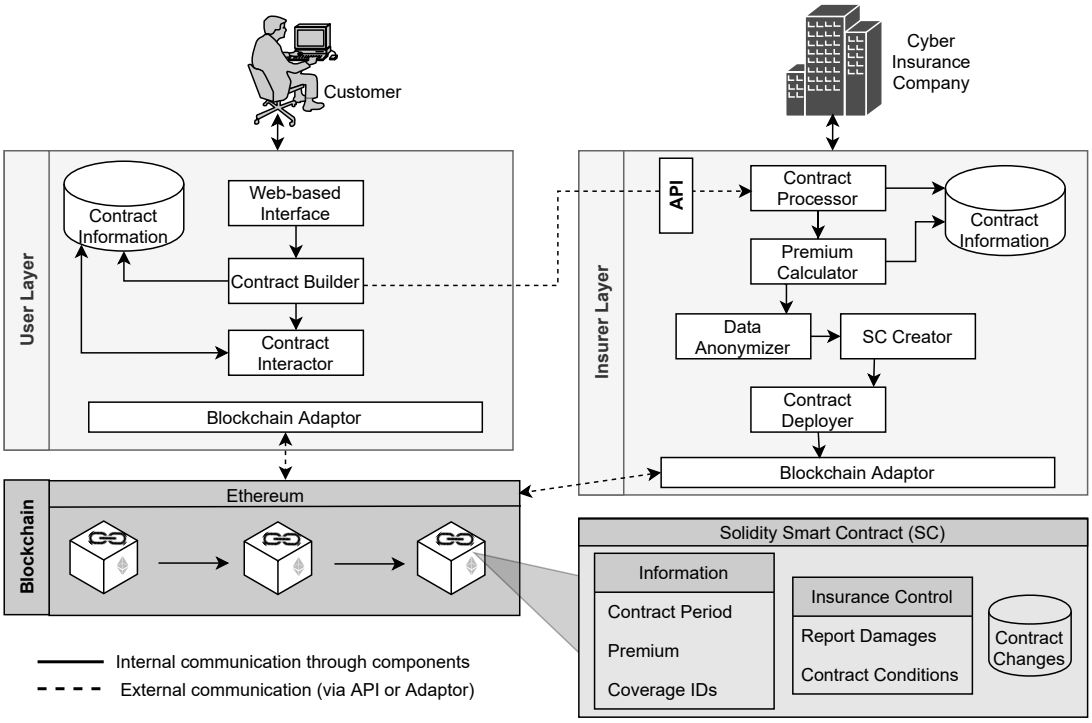

Figure 4.1: Extended SC4CyberInsurance Architecture [35]

From the customer perspective, the interface enables the usage of all necessary functionalities, including an updated contract creation functionality and the continuous interaction with the contract. Also, to ensure the right deployment of the contract agreements as

a SC, both parties still are connected to the Ethereum blockchain. The SC functions will now be executable in an easier manner. The developed front-end always specifies the parameters necessary for executing SC functions automatically by ensuring that the data used is always up-to-date. Hampering is hindered by regularly checking that all users have access on the same data. Therefore, the SC4CyberInsurance system is improved in terms of simplicity and feasibility, and the function handling is automatized on a higher level.

The different components of the SC4CyberInsurance system need to be visible for all users at the same time. A customer has to be able to see if the reported damage claim was accepted or not, or if a counteroffer was sent. Also, the customer and insurer should always be able to see the state of contract updates, so that there will not be any discrepancy of the data stored. Overall, the design should simplify and automatize the system to a certain amount, such that the users do not have to worry about technical details or lost data. In order to design such an interface, the layout had to be defined. A cockpit-based layout was selected in order to be able to always observe the different states and events occurring. Figure 4.2 shows a scenario of the final design from the customers perspective. The cockpit layout ensures that the user can see as much information as possible at the same time.

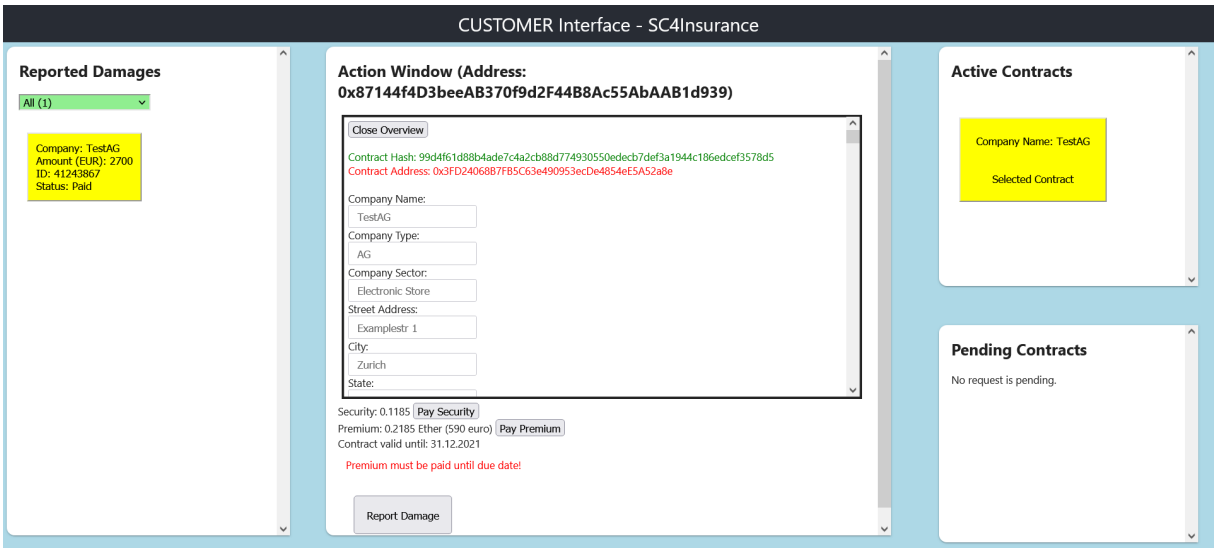

Figure 4.2: Screenshot of the Proposed Web-Interface Layout

The developed UI (User Interface) is able to handle all functionalities defined in the original SC4CyberInsurance [13] system as well as some additional ones, such as an improved contract creation process and claim handling. Also, the interaction between the involved parties is ensured and happens in real-time.

4.2 Contract Creation

Before deploying a Cyber Insurance contract in form of a SC, an agreement between the two involved parties must be made. Therefore, the customer company has to provide and submit necessary information in order to create a contract agreement with the insurer. The customer can provide the necessary information by filling out a dynamic form. As Cyber Insurance contracts are complex in terms of risk assessment, the customer company has to provide detailed information about the history of attacks and the technology used. Also, the customer can define the contract coverage using a predefined drop-down list, that is currently based on [37]. The damage types per attack are based on the list of [13], and both lists are extendable. The submitted information then needs to be processed and a premium must be calculated.

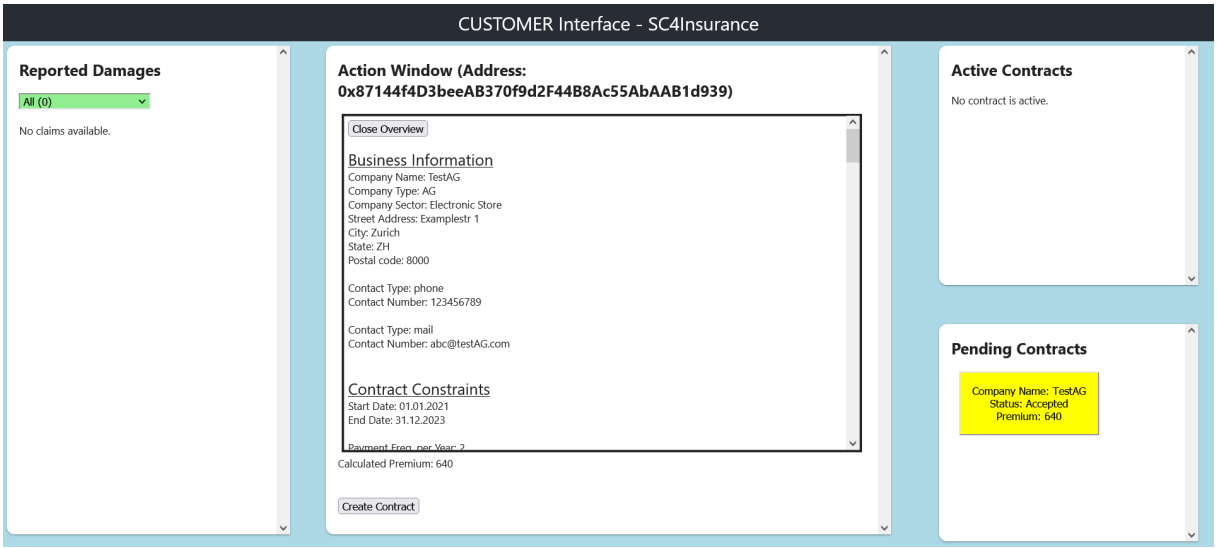

Figure 4.3: Visual Overview of the Request Information and State

The solution introduces a new functionality that should support the contract creation. In the original prototype, the contract was directly created when the customer triggered it. From now on the customer has to send a request to the insurer, who then can decide whether to continue with the contract creation or not. If any inconveniences exist, the insurer can simply decline the request. If the insurer decides to create the contract, the premium can be calculated by mentioned party. Therefore, the insurer is in charge of calculating the premium, which is based on the information the customer provides. The premium is calculated based on a predefined algorithm which can be adapted easily. By accepting the request, the customer afterwards has to finally decide whether or not to trigger the contract creation. Before that final decision, the customer can see the calculated premium. If the customer does not agree with that premium, the insurer can be contacted externally to cancel the request and therefore delete it. Both actors can always see the actual state of the request and, if available, see the premium that has been calculated. Figure 4.3 illustrates a scenario, where the premium has been calculated by

the insurer and the customer only has to finally trigger the contract creation. Both parties can see the same request state and can open an overview.

4.2.1 Premium Calculation

A very important part of the contract creation is the premium calculation. As mentioned in Subsection 2.1.3, there are different possible ways to calculate a premium based on customer information. In this solution, the pricing approach is followed, where a base rate is adjusted depending on what information is provided by the customer. The attack history, the technological infrastructure, and risk metrics are included. The approach is extendable and new risk assessment metrics can be added. In this Subsection, the pricing approach is explained briefly, and upgrades of the original SC4CyberInsurance system are made clear.

The customer provides following information [13]:

- **Business Information:** This includes identifying information about the customer itself.
- **Contract Constraints:** The customer defines the preferred start and end date of the contract, as well as the yearly payment frequency that has to be followed.
- **Company Conditions:** This includes information about the revenue and size of the company. Also, the percentage of total yearly revenue that is estimated to be due to technological infrastructure is defined here.
- **Company Security:** This information is essential to calculate the premium accurately. Besides executed risk assessment metrics, which are summarized in a predefined, but extendable list, the customer must also provide information about attacks in the past, security software usage, and executed security training.
- **Company Infrastructure:** Here, all the technological aspects of the company are included.
- **Contract Coverage:** This includes the attacks and impacts that the customer wants to cover with the Cyber Insurance contract.

The premium calculation in this solution considers different aspects and therefore targets to provide an accurate pricing approach. First, the base rate is defined. A base rate of 580 is predefined in the calculation, but that can be adapted easily. Then, the yearly revenue that is based on the usage of technology is calculated. This leads to a premium, which is only based on the company performance and a base rate.

The result is then adapted, by considering the company's security and technological infrastructure. The available list of risk assessment metrics that is included in the prototype can be easily extended adding new ones. For the prototype, two cybersecurity metrics have been chosen, based on the list of [36]. The metrics chosen are *"Number of systems*

with known vulnerabilities" and *"Volume of data transferred using the corporate network"*. However, the metrics can be adapted easily and more can be added. Besides the two metrics mentioned, the *"Level of Cybersecurity education"* is assessed too. Depending on what metric result is provided, the premium is adapted differently. For example, a low level of cybersecurity education means that the company has a higher risk level. On the other hand, a low number of transferred data leads to a lower the risk level. The history of attacks is also considered in the premium calculation. The list of attacks is predefined and based on the list provided by [37]. Here, the number of attacks is important to estimate the premium.

The company infrastructure also plays an important role for the premium calculation. Depending on the number of connected devices and technologies used, the premium is adjusted. For example, if an operating system used has not been updated, the premium is higher than with installed updates. Also, the more devices are connected with the company's infrastructure, the higher the premium will be, because of higher cyber risks. On the other hand, installed security software, as well as employee training, decreases the premium.

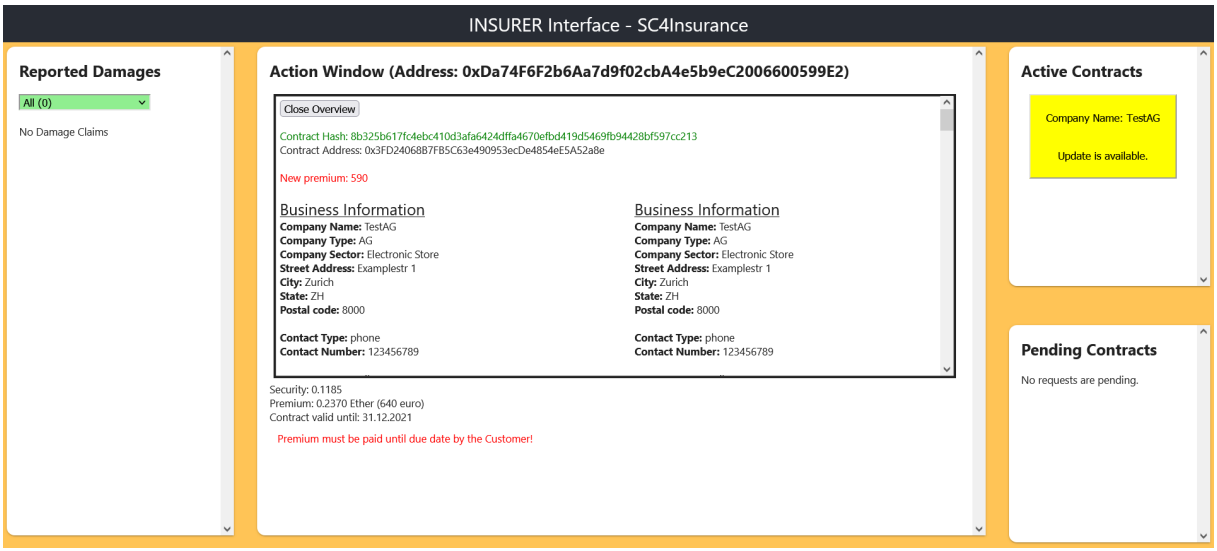

Figure 4.4: Contract Update Overview from the Insurer's Perspective

4.3 Contract Updates

The proposed solution also provides the possibility handle contract updates interactively and automatically update the state of all users involved. After a contract has been created, and the customer paid the security, the contract information can be updated. As shown in Figure 4.2, the contract overview of the customer already includes a form. The form fields are already filled out with the existing contract information. The customer can change the fields available and send an update request to the insurer. Changing contract information

might lead to a different premium, which is also shown to both parties, if available. After sending an update request, the customer is able to open an overview that includes the existing contract, as well as the update request information. The insurer can always see if an update is available for a specific contract due to the application continuously fetching data. The contract overview shown in Figure 4.4 includes the existing information as well as the update information. The insurer can therefore decide whether or not to update the contract conditions. Accepting the update request automatically lets the new contract information take the place of the old one. Therefore, all SC attributes are updated, too.

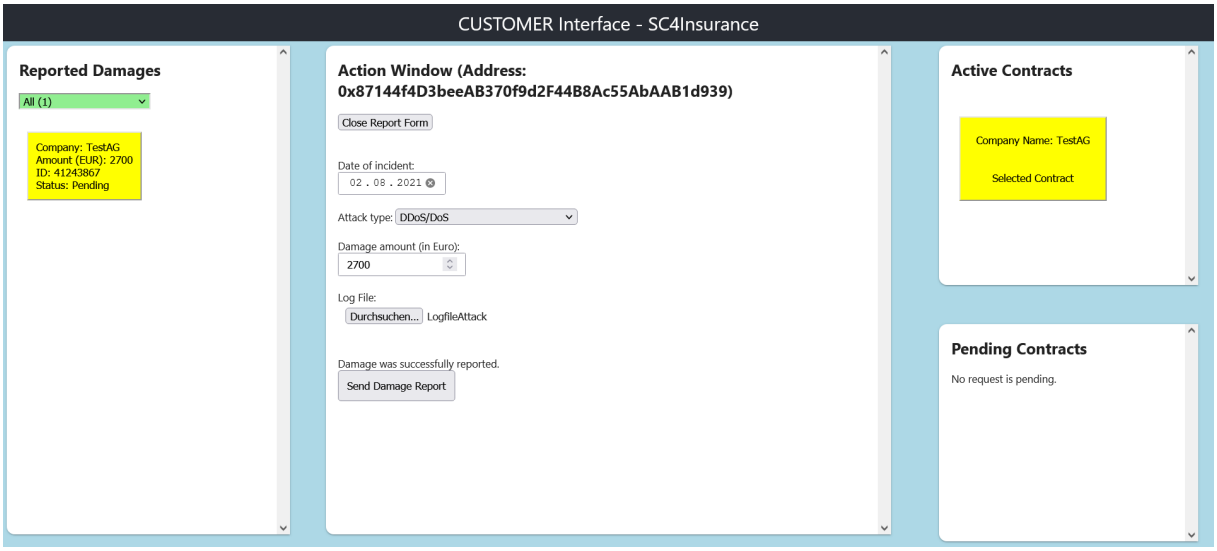

Figure 4.5: Damage Report Overview from the Customer's Perspective

4.4 Claim Handling

When it comes to claim handling, the solution provides a very interactive and simplified manner of dealing with it. Figure 4.5 shows the customer interface when a damage has been reported. The customer can simply open the report form by first opening the contract information overview and then navigating to the form. Automatically, the right contract address is specified and the damage report would refer to the chosen contract. For reporting a damage, the customer has to provide information about the date of occurrence, the type of attack, which is also a predefined drop-down list based on [37], the amount of damage incurred, and a log file. The log file submission is mandatory for evidence and the damage report cannot be sent to the insurer without uploading a log file via this application. The customer has to be sure that the damage amount reported does not exceed the according coverage. To achieve that, the customer can simply check the existing contract information.

The insurer on the other hand, can always see all existing claims. By selecting a reported claim, the insurer opens up a damage overview, as seen in Figure 4.6. All necessary details

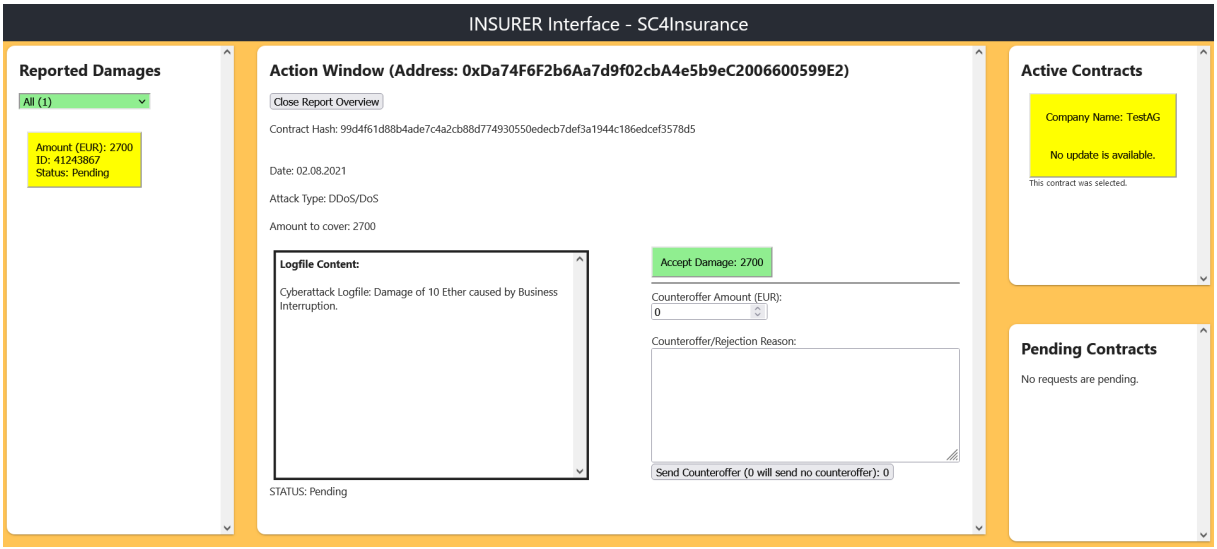

Figure 4.6: Claim Overview from the Insurer's Perspective

are shown, including the log file content. Also, the existing contract is marked, so that it is always clear what damage claim belongs to what contract. This ensures that the insurer can check the contract agreements that the claim belongs to and to decide how to handle that claim.

The claim states are defined as follows, based on [13]:

- *Pending*: The claim has not been accepted or declined by the insurer yet. No counteroffer has been sent, as well as no payment has been executed yet. This is the assigned state directly after the customer reports a damage.
- *Cancelled*: The claim has been cancelled by the customer, thus the insurer has to take no further action.
- *Paid*: Both parties found an agreement. The payment has been executed and the case is closed.
- *Under Investigation*: The insurer declined the initial damage report, and either sends a counteroffer or not. A report with this state needs the customer to take action by either accepting or declining the insurer's decision.
- *Dispute*: When the customer does not agree with the counteroffer or rejection of the claim, the state is changed to "Dispute". Now the insurer can either send a better counteroffer or both parties can resolve the dispute by involving third parties.
- *Resolved*: After finding an agreement externally, the state changes to "Dispute".

The users of this Web-Interface can either decide to show all claims that have been reported, or to filter for the states above. In this manner, both parties can keep track of

all changes. Also, the detailed overview of the damage reports enables the users to see the initially demanded amount, and the finally paid amount. Therefore, no information is lost during negotiation.

4.5 Payment Execution

The solution provides a simplified way to execute payments between the parties involved and to the SC itself. In the original prototype, the users had to manually specify how much ether to transfer to the SC and to the other party. With the introduction of the Web-based Interfaces, the payment handling is much simpler. The interface shows the security and premium amount for each contract selected, as shown in Figure 4.2 on the left part of the screen. The amount of ether and euro is shown right in the contract overview. Hence, to execute premium and security payments the customer can now see all necessary information and trigger the payment by simply pressing a button. The specific amount to transfer does not have to be set by the users manually, as the amount of ether contained in the SC is always considered. This is especially important when the insurer sends a counteroffer. For example, when the SC already contains enough ether to execute the payment with the counteroffer amount, the insurer does not send any ether to it. Otherwise, the insurer sends only as much ether as needed.

4.6 Address Configuration

The possibility to use a specific account address has been introduced with this solution. In the beginning of each session, the users are requested to configure their account address by using their private keys using a log-in screen. This is one way for ensuring dynamic address assignment, instead of assigning the addresses statically. All further transactions of a user will then use the defined address. Another option would be to use credentials such as email and password, and to set the wallet address and private key manually as a part of registration. The account address would then be used every time when a transaction needs to be signed. This solution provides the former method to simulate dynamic address configuration.

Chapter 5

Implementation

Chapter 4 showed the functionalities of the solution and the upgrades done. In the following sections of this chapter, the implementation process is explained in detail. Subsequently, Chapter 6 visualizes the implemented processes by executing various case studies based on a simulation set up.

To improve the feasibility of the SC4CyberInsurance system, a Web-based Interface was implemented in the scope of this thesis. The source-code of the implemented prototype is available publicly at `https://github.com/fimami/SC4CyberInsurance-WebInterface.git`. As insurers and customers are two user groups with different needs, it was decided to develop two separate user interfaces. Both interfaces need to be able to execute the core functions provided. In order to ensure a proper usage of the solution, the original system including its Python Flask API and scripts had to be adapted to execute the processes efficiently.

5.1 Implementation Overview

The Web-Interfaces have been implemented using React version 17.0.2, a Javascript library. It has been chosen, because it provides a simple way to break down complex UI into reusable, smaller components. The solution has to provide numerous functionalities and to show the users as much information as possible at the same time, without being too complex; React supported the development of such an intuitive, cockpit-based design. Additionally, the solution has to guarantee that all users are able to see real-time based information, making periodical fetching for data necessary. Reacts Virtual DOM supports that functionality, as it enhances the performance by only triggering changes in the front-end when the data changes.

The server-side of this solution is implemented with Python and the underlying API is Python Flask. It has been decided to use the same technologies for the back-end as in [13] to ensure the feasibility of all SC functions. The existing API has been adjusted and extended entirely in order to execute all functionalities that the interfaces provides. Also, SC functions have been added, in order to receive the premium, security amount,

and the validation status. The database used is SQLite, which enables the testing of all functionalities that the solution must provide.

Both interfaces have similar visual aspects. As already shown in Chapter 4, the interfaces consists of several containers. In the following subsections, the interactions between the interfaces and the API are shown, as well as the data flows are briefly explained.

5.1.1 Contract Agreement Definition

To define the contract agreements the customer has to fill out a simple, dynamic form. As the information needed for Cyber Insurance contracts is relatively complex in terms of technological details and coverage, the form enables the customer to add and/or delete form fields dynamically, where needed.

The customer has to initially define, which attack type to be insured against. For every attack type defined, the customer can select one or more impact/damage types to cover. The form fields are mapped to allow the input of nested value pairs. The customer is able to simply send the form data to the server, which then stores the contract information to the relevant databases as a "pending contract". No SC is deployed at this point, as the insurer yet has to calculate the premium, and to accept the request. This protects the insurer from malicious intents, where a customer could repeatedly create and deploy SCs, and the insurer would repeatedly pay transaction costs. Also, the customer can not directly calculate the premium via the customer interface.

After the request has been stored on the relevant databases, it appears in the front-end. For every instance fetched from the database, a clickable item is rendered in a specified container called "Pending Contracts". The items always show the state of the request, which initially is defined as "New", and the company name of the customer. The contract request can then be processed as explained in Section 4.2. The status, company name, and a calculated premium is always visible in the pending contract list item.

In Figure 5.1, the adapted process of contract creation and its API calls are briefly shown. The customer triggers the creation of a contract request by submitting the contract form. This event stores the pending contract to all relevant databases. In order to show all existing contract requests, both interfaces continuously fetch for data. In this case, the company name, premium, status and the hash of the request content are fetched. The insurer can simply open up an overview of the contract request by clicking on the item shown in the "Pending Contracts" list. A click on such an item triggers the `getPendingContractInformation` call, which gets the data with the according hash from the database. The customer can also open up an overview the same way. The insurer can calculate a premium for the request information provided by the customer, and then either accept or decline the request, as discussed in 4.2. In case of declining the contract request, the pending contracts simply are deleted from all relevant databases. Accepting the request calls `acceptPendingContract`, which triggers an update of the contract request changing the status to "Accepted" and updating the calculated premium in all relevant databases. Afterwards, the customer is able to trigger the definitive contract creation, which then deploys the insurance contract as a SC. Here, the pending contracts are deleted

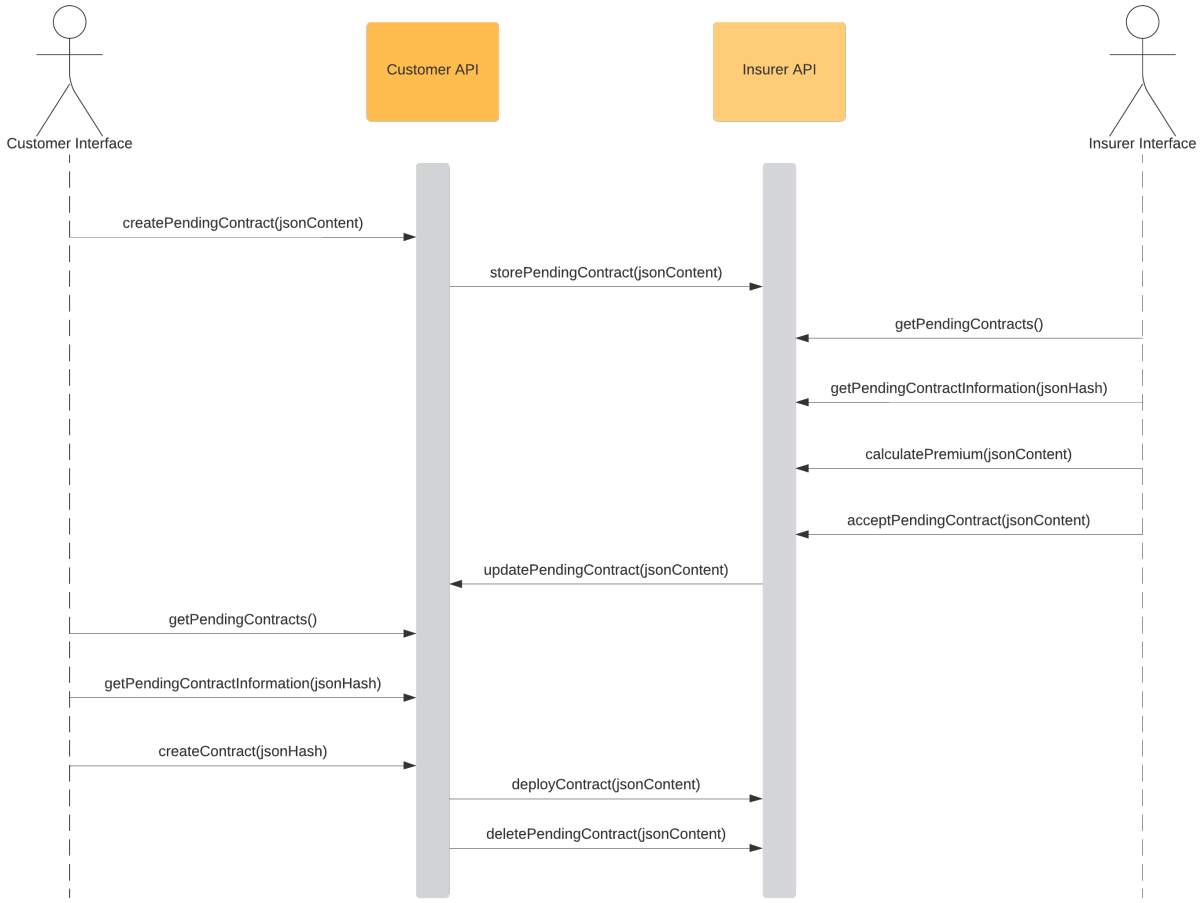

Figure 5.1: Contract Creation Process

also, because the definitive contract is stored on both databases and it is not needed to store a copy of it. The sole process of deploying the contract as a SC is not thoroughly explained here, as it does not differ much from the original SC4CyberInsurance prototype [13].

As the customer is limited to having one insurance contract, it is not possible to fill out another form after a contract has been definitively deployed. But on the other hand, the insurer can have multiple customer at the same time. The interfaces are developed in a way, that ensure the feasibility in such a case, allowing the insurer to manage multiple contracts at the same time.

5.1.2 Automatized Data Fetching

After a contract has been finally deployed, both user groups have access to additional functions, including opening an overview of the existing contracts that are fetched periodically. Figure 5.2 illustrates the fetching for data, that is executed passively, although only when the interfaces are being used. Deployed contracts as well as pending contracts

are fetched periodically. In order to ensure that both user groups can have a proper interaction when the customer wants to update contract conditions, they will always get the according data from the back-end. At the customer side, this is handled by using `checkForNewProposal`. As the customer can only have one insurance contract, this API call checks if there is an update available. If an update proposal is available, the customer can additionally open an overview of the update request. The insurer interface on the other hand repeatedly triggers an API call `getNewProposalByHash` passively, which returns the hash of the update entry and its stored message. The message is either *No update is available* or *Update is available*, and is shown in the contract list for every active contract item. Also, both interfaces periodically call for reported damages, but only when active contracts are available, so that the performance remains high.

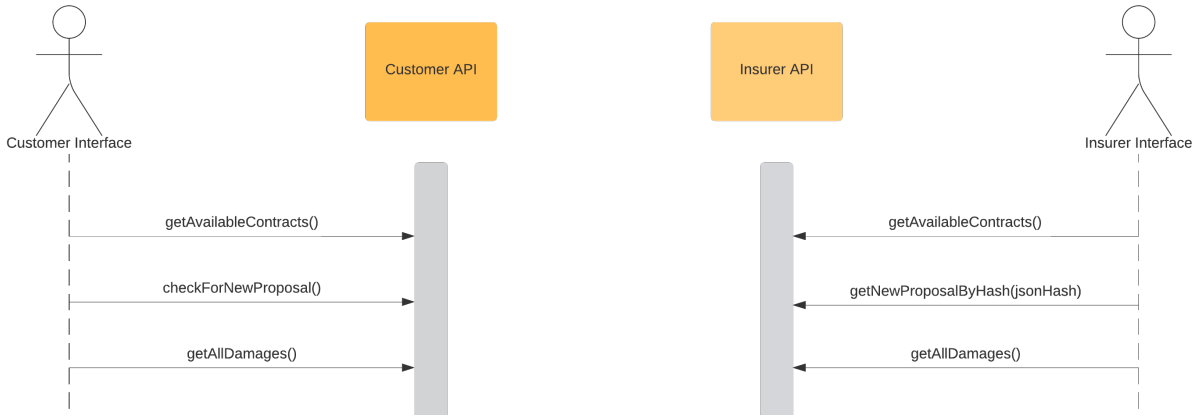

Figure 5.2: Passive Fetching for Data

5.1.3 Security and Premium Payment

In order to report a damage claim or to update contract agreements, the customer must pay the security first and validate the contract. Figure 5.3 illustrates, in what order the calls are executed, triggered from actions in the front-end. First, the customer has to select the deployed contract showed in the contract list. By clicking on the contract item, the `useContract` call is triggered, which uses the hash of the shown contract as parameter. The SC and hash of the selected contract are then set in the back-end, so that future SC functions and database updates are enabled. Clicking on the contract item shown in the front-end opens up an overview and triggers multiple calls to fetch the necessary data. The calls `getValidUntil` and `getValidBool` get the date of validity return if the contract is valid or not. The contract is valid after the security has been paid. The date of validity is lengthened every time the customer pays a premium, according to the contract agreements. To show the customer an overview of the premium and security in ether as well as in euro, the calls `getPremium`, `getPremiumInEther` and `getSecurity` directly access the SC. As always when a user opens up a contract overview, the `getContractInformation` call sends a request for the information stored under a specific hash. To pay the security and premiums, the customer can simply click a button shown next to the amounts fetched

before. The calls `paySecurity` and `payPremium` take the fetched amounts as parameters and execute the payments automatically. To avoid multiple payments of the security and the payment of the premium after the contract lifetime, the front-end fetches the relevant overview data repeatedly after any payment execution.

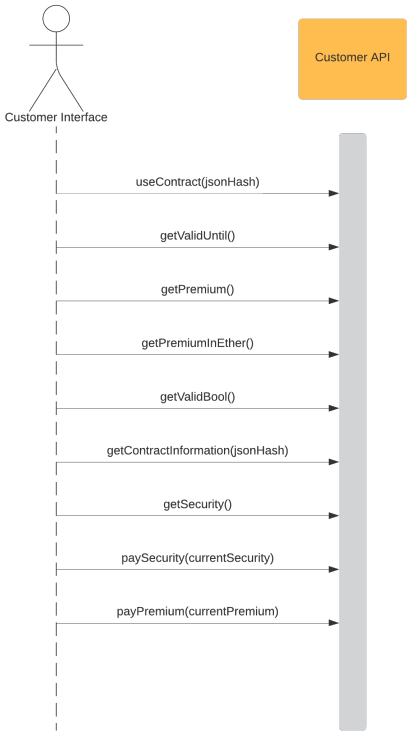

Figure 5.3: Premium and Security Payment Process

5.1.4 Contract Update Handling

When the security is paid, the customer is able to send an update request to the insurer. For that, the customer simply can access the existing contract information and manipulate various fields in a predefined form. As earlier shown in Figure 4.2, the contract overview of the customer includes a form, instead of only showing the information text-based, as it is the case for the insurer. The usage of a form enables the customer to simply change the contract information directly. Also, the form includes input validation, as several fields are not able to be changed. The customer can adapt the fields as preferred, and send an update request to the insurer by simply clicking a button at the bottom of the form. This triggers `proposeToUpdateContract`, which takes the information defined in the update form as a parameter. The insurer on the other hand can compare the original contract with the update request as already shown in Figure 4.4. Declining the update request triggers `declineToUpdateContract`, which deletes the contract updates in all relevant databases directly and also according variables stored in the SC. Accepting the update trigger `agreeToUpdateContract`, which deletes the old contract agreement in the

database and definitely sets the new, updated agreement as the valid one. Also, the SC is adjusted.

5.1.5 Damage Report and Negotiation

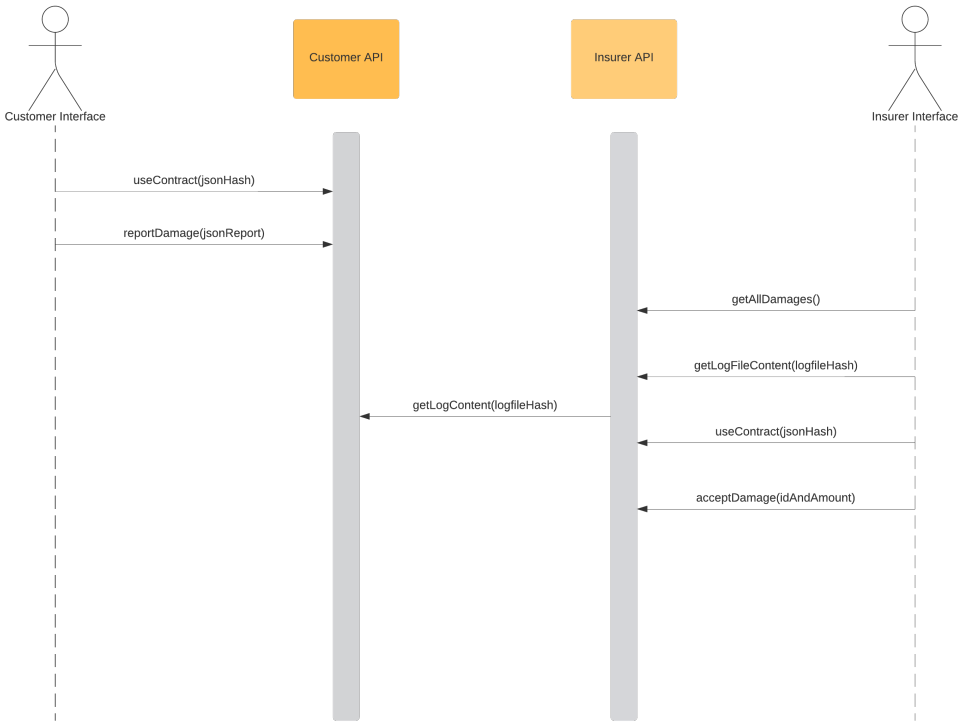

Figure 5.4: Process of Claim Handling where the Insurer accepts the Claim

The process of damage reporting and claim handling is one of the most important functionalities implemented. In the case of an incident, the damage reporting must work perfectly. Figure 5.4 visualizes the interactions this process consists of. By selecting an active contract in the contract list, the customer triggers the `useContract` call, as previously explained. Here the customer can simply navigate to open a ”damage report” form, by clicking on the according button in the contract overview, as shown in Figure 5.5. The ”Report Damage” button is only available after the customer has paid the security, and therefore after the contract has been validated.

Clicking mentioned button opens up a new form, where the customer has to enter all necessary information about the incident. That includes the date of incident, the attack type, the amount of damage, and a log file for evidence, as shown in Figure 5.6. Sending the damage report to the insurer triggers `reportDamage`, which sends the damage information inserted to the back-end. The information is stored in the SC as a damage report, and the log file is stored in the database of the customer. To ensure, that the log file will not be manipulated, the hash of the log file is stored in the SC also. The contract must at

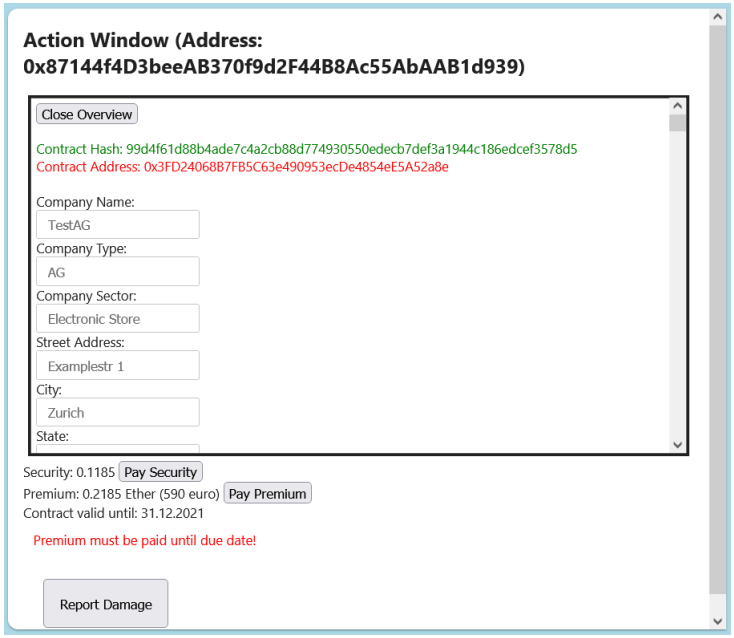

Figure 5.5: Contract Overview of the Customer after Security Payment

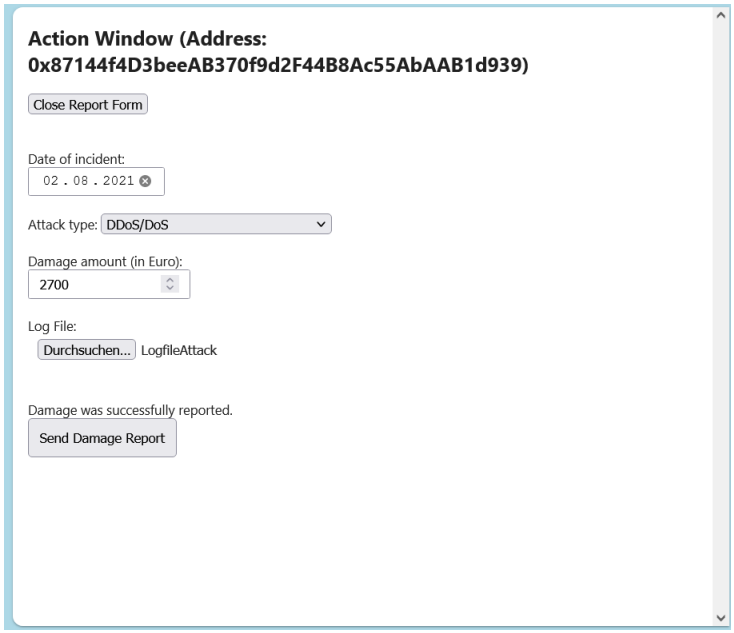

Figure 5.6: Damage Report Form

least be valid until the date of incident, and the log file is mandatory to upload, in order to send a damage report successfully.

The payments are always executed by transferring ether between insurer and customer. For example, when the customer pays the premium and the security, the ether is trans-

ferred from the customer to the SC. The payments are executed entirely automatized and the users only need to press one button to start the process. During the claim handling process, events and actions need to consider the account balance of customer, insurer, and SC in order to avoid unnecessary movement of currency. When the insurer accepts a claim with the status "New", the damage amount is transferred to ether and a transaction of currency is triggered from the insurer to customer account. However, if the insurer declines a new damage report without specifying a counteroffer, no currency is transferred. When the insurer specifies a counteroffer amount, then it is checked how much ether is currently stored in the SC. If the amount of ether stored in the SC is large enough to execute the payment of the counteroffer, the insurer does not transfer ether to it. Otherwise, the insurer transfers only as much ether to the SC, as it would be needed to transfer the counteroffer amount from the SC to the customer. Additionally, the selection of a damage list item triggers a state change in the front-end, which then shows the insurer what contract belongs to the report selected.

Chapter 6

Evaluation

This chapter provides, by conducting three case studies, evidences of the feasibility of the developed prototype. All functionalities of the solution are executed based on one defined scenario. The case studies show all different possible interactions between an insurer and a customer, thus highlighting all features developed. Therefore, the usage of the tool will be thoroughly explained. The first case study focuses on the contract creation and the interactions between customer and insurer during the process. The second one shows the interactions of the customer with the deployed contract, and discusses a contract update process. In the third case study, the claim negotiation is briefly shown, including the discussion of all possible interactions. In the end, a discussion of the solution's advantages and limitations is provided.

6.1 Case Study No. 1 - Contract Request and Creation

For this case study, it is assumed that the insurer and customer both have blockchain addresses, whose contain Ether (*i.e.*, Ethereum's cryptocurrency). The customer is a company named MacrosoftIT, that operates in the software development industry. Due to the increasing number of successful cyberattacks against other industry competitors, the company played with the thought of purchasing Cyber Insurance products. After MacrosoftIT experienced a cyberattack themselves they decided to purchase such a product in order to be protected from impacts in the future, as the past investments in cybersecurity could not impede the recent cyberattack. To set up a contract, that entirely fulfills the needs of the company in terms of Cyber Insurance, they use the developed tool, as it assesses a high amount of information to provide a tailor-made insurance product. To sign up with their blockchain address and be able to use the tool they log in using their private key, as shown in Figure 6.1.

The insurer also has to log in the same way to use the tool. After the customer is able to access the tool, he/she opens the predefined form to enter their information and the coverage needed. The customer then provides the company information and contact information, so that the insurer is able to get in touch with the customer. The customer

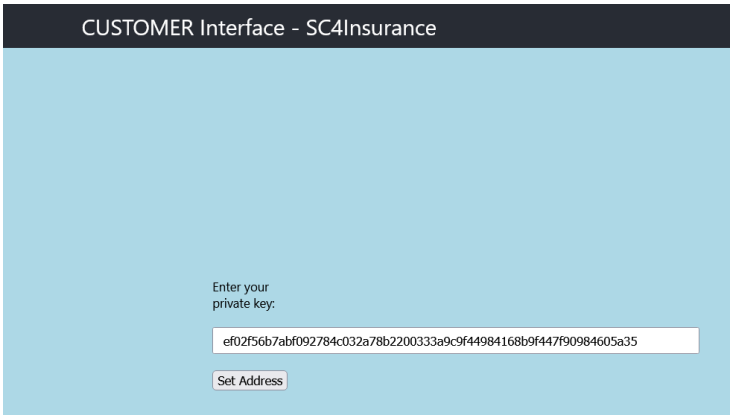

Figure 6.1: Customer Assigns Its Address Using the Private Key

wishes to start the contract in 2021 with a running time of three years, and a payment frequency of twice a year. The percentage of yearly revenue of the last year, that has been earned with he usage of technology, also has to be provided. Also, the company already executed three risk assessments in order to estimate the cyber risks it is facing. The metrics used and the results of those are also provided by the company, as well as attacks experienced in the past, and technology information. The company wants to purchase a Cyber Insurance product for protection against DDoS, Data Breach, and Ransomware, as they currently are the most observed attacks by the IT department. Various impact types for each of those attacks are selected out of a predefined list, as well as the coverage ratio, deductible, and maximal indemnification are specified. This is at least a job for the IT project leader of the company, as the definition of the different aspects requires a certain level of cybersecurity know-how. Figure 6.2 shows, how the customer is able to specify a coverage.

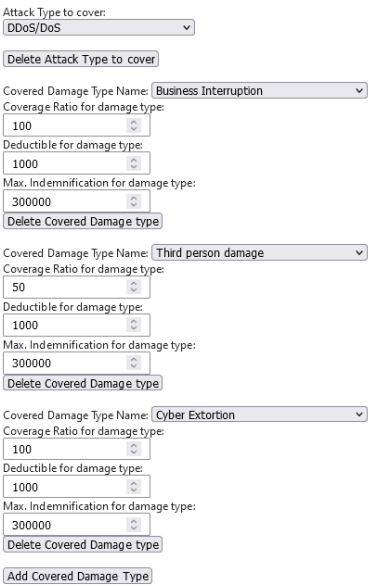

Figure 6.2: Definition of a Coverage using the Form

After the customer finished to fill out the predefined form, they send a contract request to the insurer. Every contract requests automatically appears in the interface of the insurer.The insurer has to handle the request by clicking on the list item. That interaction opens up an overview of the contract request. The insurer already clicked on the according button to calculate the premium, and therefore the premium is shown on the bottom of the overview, as it can be seen in Figure 6.3.

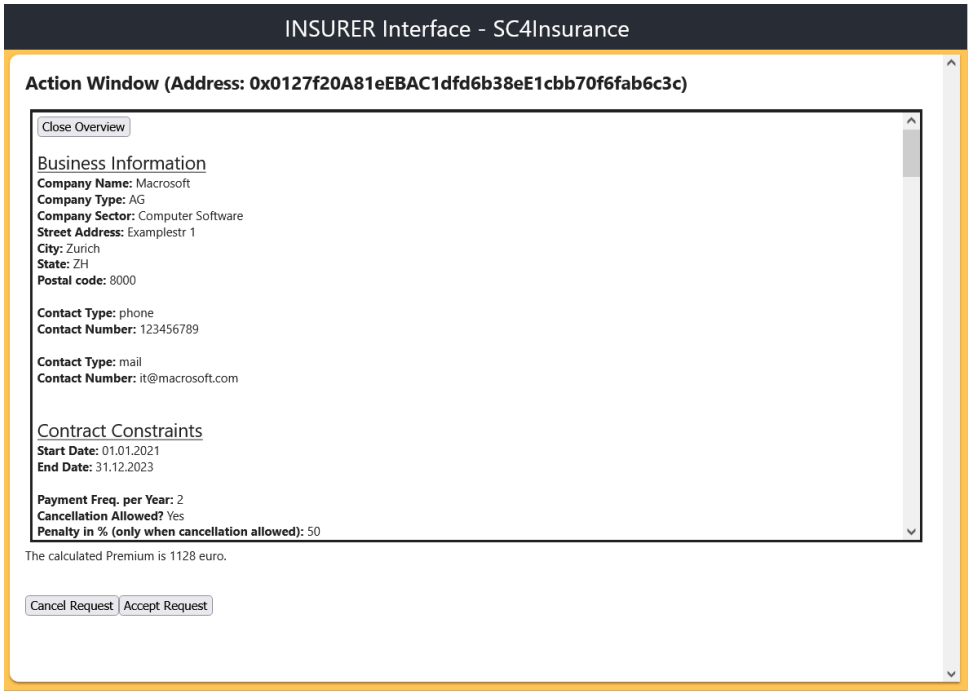

Figure 6.3: Contract Request Overview of Insurer

The insurer could always cancel the contract request. This would be the case, if the information provided by the customer is insufficient. However, a more probable case of canceling the contract request would be after the insurer accepts the request, and the customer then does not agree with the calculated premium. In this scenario, the insurer decides to accept the request and set the premium. The customer can also see when the contract is ready to be created in the bottom right corner. Figure 6.4 shows how the accepted request looks like.

The customer clicks on the shown item to open a similar overview as in Figure 6.3, only now it is possible to directly create the contract. MacrosoftIT agrees with the calculated premium, and therefore decides to create the contract and become a definitive customer.

6.2 Case Study No. 2 - Premium Payment and Update Request

The contract between MacrosoftIT and the Cyber Insurance company has been created. As required, the customer directly goes ahead to pay the security and the first premium.

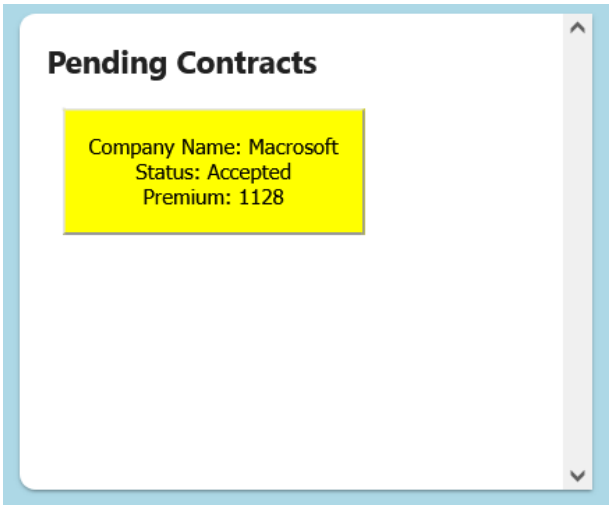

Figure 6.4: Accepted Contract Request Item

For doing that, the contract overview has to be opened, by clicking on the active contract list item. The overview now differs a lot from the request overview (*cf.* Figure 6.5. The customer pays the security by clicking on the ”Pay Security” button, and the premium right afterward.

After three months of contract duration, MacrosoftITs infrastructure has changed, making an update of the contract necessary. They installed updates for all their software used, and also executed a security training for the employees. Therefore, they update the contract agreements by editing the original information directly in the overview. The form makes it possible to adapt the contract information quickly. After changing the information to be up to date, the customer sends the update request to the insurer. The new premium is calculated after the customer has sent the request, in order to avoid malicious cost optimizations. The new premium would now be 100 euros less than before. The insurer is then notified by the update. Thus, the insurer clicks on the shown list item to open an overview of the contract and the updates. Finally, the insurer decides to accept the update, as the changes seem to be valid.

6.3 Case Study No. 3 - Damage Report Negotiation

During the first period (*i.e.*, first half of the year) of the contract lifetime, no incidents happened. Except for the contract update, no actions were executed. Therefore, the validity needs to be increased by the customer, and the premium needs to be paid. After that, during the second period, the customer company is the target of a cyberattack. A DDoS attack was successfully executed and the business was interrupted for a short period of time. Although the attack did last only a couple of hours, the company experienced a big loss due to the business disruption. The customer can report the damage, but first needs to check the contract conditions. As the impact type of the attack is business interruption and the attack type DDoS, the amount requested from the insurer must be adapted according to the coverage specified. Let us assume that the customer defined for

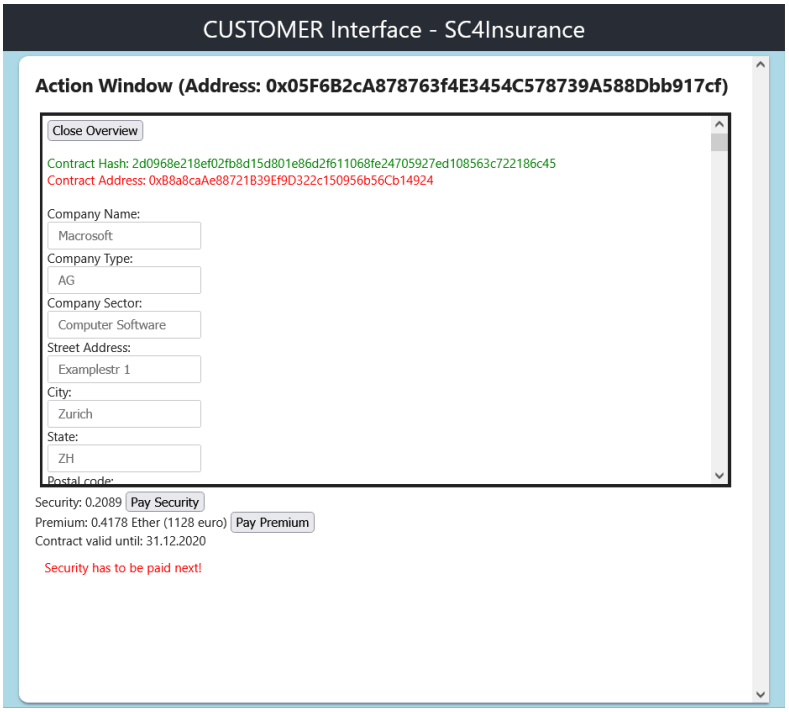

Figure 6.5: Customer Overview of the Active Contract

the damage type of that specific cyberattack a coverage ratio of 100%. The deductible was defined as 1000 euro, and the maximal indemnification is 300'000 euro. The attack costed the company about 27'000 euro. Therefore, the indemnification that can be requested would finally be 27'000 euro minus the deductible, resulting in 26'000 euro. The customer therefore can report a damage of maximum 26'000 euro to the insurer. The customer interface for reporting damage is presented in Figure 6.6.

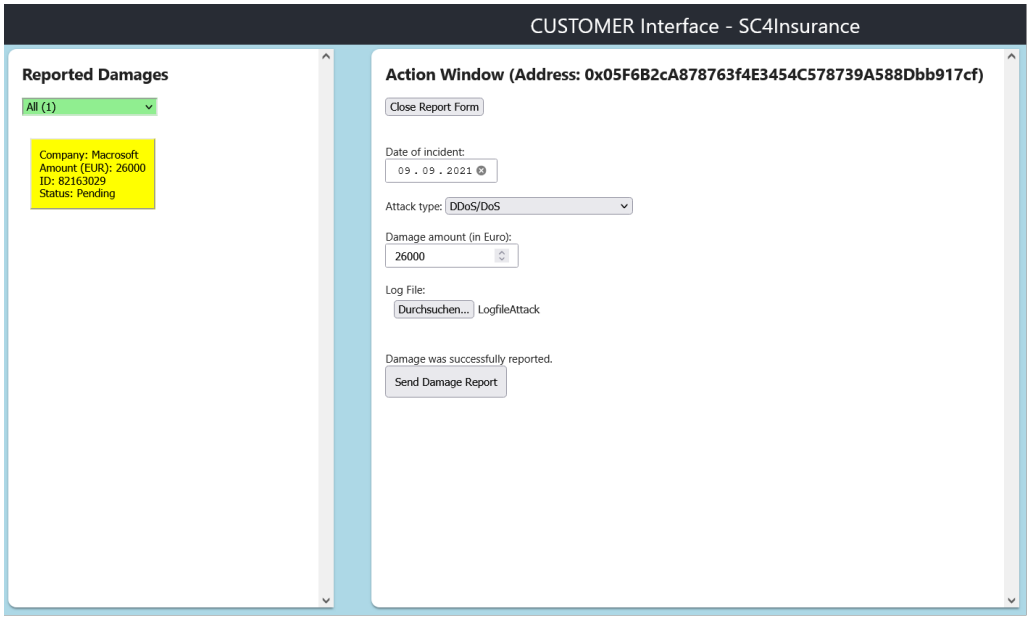

Figure 6.6: Customer Interface after Reporting DDoS Attack

The insurer then automatically receives the request and all of its information in the Reported Damages container on the left side of the insurer interface. Next, it is needed to check the damage report information by clicking on shown item. That opens up an overview, as shown in Figure 6.7. The insurer checks the log file content and the reported amount. Also, he/she has to open the according insurance contract on the top right corner of the interface, to check if everything is covered. It can be seen, that the damage report is valid and can be accepted. Therefore, the insurer accepts it, by clicking on the "Accept Damage" button.

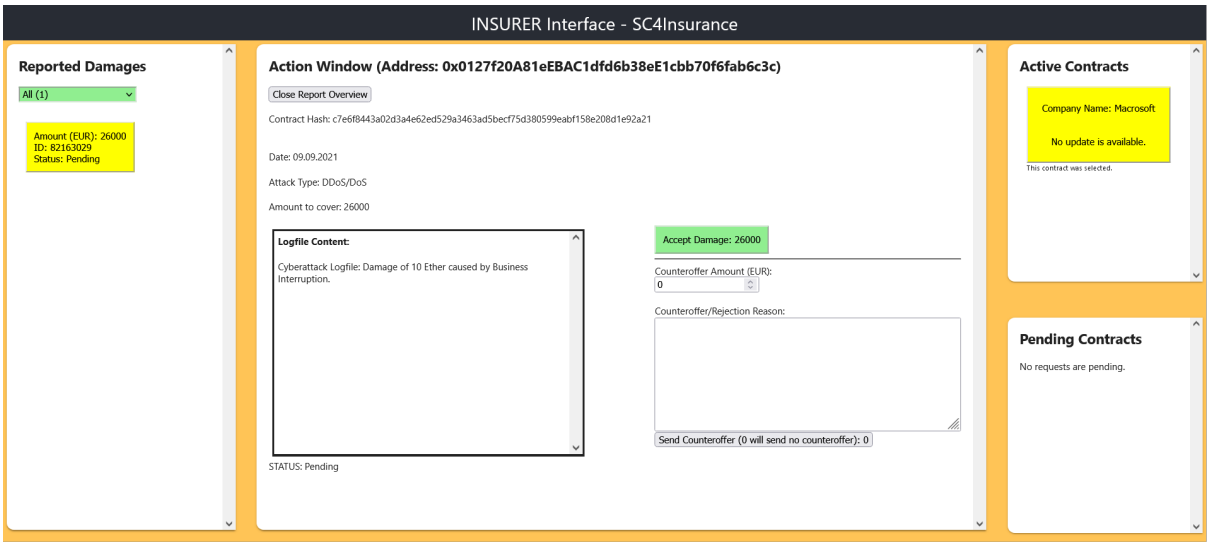

Figure 6.7: Insurer Overview of DDoS Damage Report

Soon after the last cyberattack, MacrosoftIT is targeted by another one. Now, the company was a victim of a data breach. The company is receiving much attention from the media, as it has been the second cyberattack in a short period of time. Also, the data breach impacted a lot of other businesses connected with the customer. To mitigate the damage, the customer reports it to the insurer. The claim includes the third-party damage caused, as well as reputational damage. Although the damage amount of reputational damage is not included in the coverage specification for a data breach, the customer demands an indemnification for it, as MacrosoftIT considers it to be due to privacy law violation. If that were the case, all the reported damage would be covered, as privacy law violation is included in the coverage specification. However, cyberattacks can result in numerous kinds of costs, which can be hard to differentiate. In this case, half of the estimated damage amount reported by the customer is due to reputation damage, and the other half should cover the damage caused by third-party damage. The third-party damage amount includes costs caused due to the private data of connected businesses being in the hands of the attackers. The customer classified the reputational damage as a data privacy law violation instead. MacrosoftIT expects that the public resonance of the attack would lead to regulatory fines having to be paid.

Finally, by considering that the cyberattack led to estimated damage of 2,000 euro, the damage report form would look like Figure 6.8. The insurer recognizes the damage type as reputational damage and, therefore, not covered after exchanging information with

the customer. A counteroffer is sent, which includes indemnification of only 50% of the claim. As the customer communicated with the insurer right after the claim has been sent initially, the counteroffer is accepted and leads to an exchange of ether between the two parties. If both actors disagree about the amount to cover the occurred damage, they can negotiate well with the usage of the tool and the functionality provided.

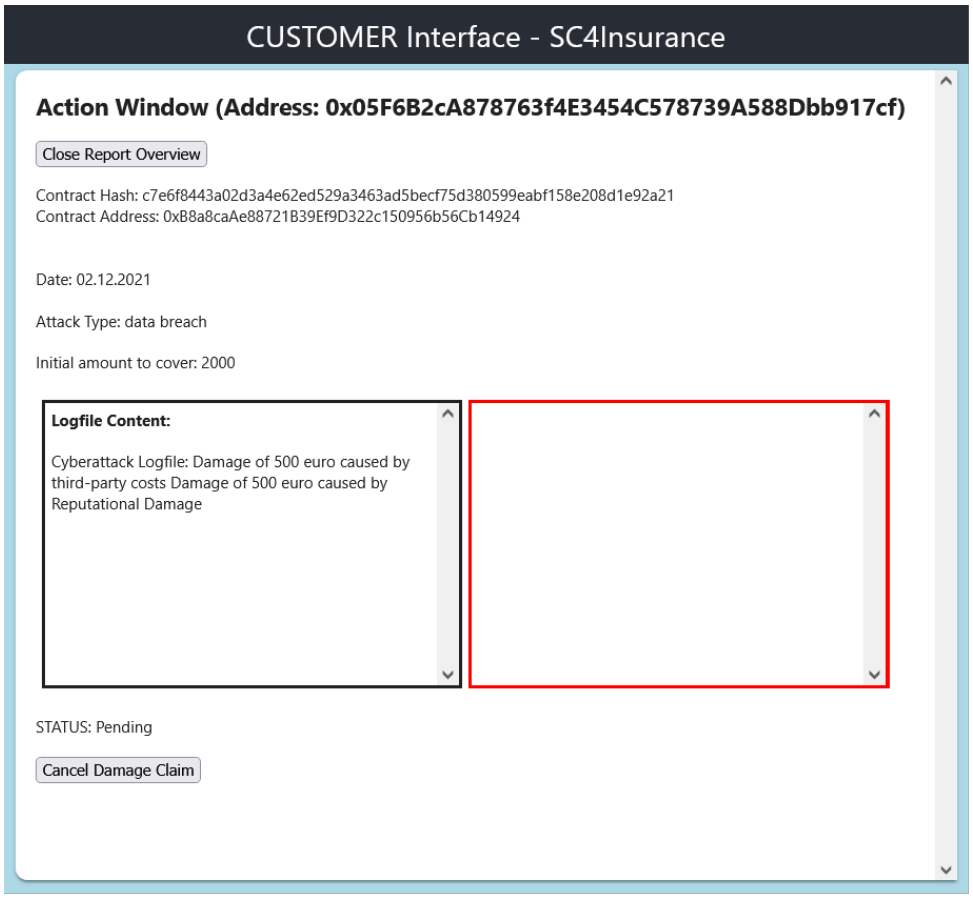

Figure 6.8: Damage Report due to Data Breach

6.4 Discussion

The three case studies illustrate an example of the usage of the tool. All functionalities are accessible and executable in a simplified manner, where the customer and insurer interact with each other seamlessly. It has been shown that during the lifecycle of one contract, including the contract creation, the customer and insurer can update the contract conditions, report damages, and negotiate about the amount of indemnification. Therefore, all functions implemented can be executed with the same contract. The more complex tasks, such as sending ether to the SC in case of a claim, are hidden from both users and handled automatically.

However, the tool has its limitations. One limitation is that the created contracts and the claims are visually not grouped very tightly. The more claims have been made, the more complicated it gets for the insurer to sort out the ones that a specific customer has made. At the current state, it can sort for the claim status, but not for the customer specifically. Although it is shown to which customer the selected claim belongs, the insurer still has to navigate through the existing claims.

The premium calculation of the solution can still be improved more. Certain adaptions have been made in the scope of this thesis, but estimating an accurate premium is still an open issue, which can be solved in the future. There exists several pricing approaches that can be followed for that, but also statistical data on cyber security and attacks can be included in the system, *e.g.* market trends.

Another limitation of the tool is report validation. The customer has to check the contract specifications before reporting damage in order to define an appropriate damage amount. The report form does not validate the provided information by itself. Additional validation would improve the user experience.

The system benefits from the privacy and immutability due to the blockchain implementation. Integrating blockchain in the Cyber Insurance process overall increases the automatization of processes and security in terms of preventing manipulation. As in this solution the hashes of contents are evaluated and compared with each other, it still provides an immutable agreement for all involved actors. However, besides the advantages that blockchain provides, there exist several limitations. To execute SC functions, the specific user address needs to pay a transaction fee. The amounts that have to be paid can be that high, that the incentive for using the system is decreased significantly. Although the blockchain implementation automatizes the processes and makes third parties unnecessary, it adds more complexity to the system.

Chapter 7

Conclusions and Future Work

The goal of this thesis was to implement a Web-based Interface that improves the feasibility and overall usage of the proposed blockchain-based Cyber Insurance approach. The stakeholders including customers and insurance companies have to be able to use separate interfaces in order to interact with the system's functions. Thus, the existing system called SC4Cyberinsurance [13] had to be adapted to work seamlessly with the implemented solution.

A theoretical background is provided by surveying existing literature on Cyber Insurance and blockchain-based Cyber Insurance approaches. The SC4CyberInsurance system, as it has been provided in the beginning of this thesis, is thoroughly discussed by characterizing different aspects. The functionalities that the system consists of are analyzed in order to separate the underlying processes. To map the proposed approach, two related systems that use blockchain technology in the sector of Cyber Insurance and cybersecurity have been analyzed and discussed. Hereby, a conclusion was able to be drawn for the current state-of-art considering blockchain-based Cyber Insurance systems and their lack of visual interfaces.

To design Web-Based Interfaces for SC4CyberInsurance, the functionalities had to be completely analyzed as well as the SC functions and the provided scripts completely understood. A layout had to be defined, which simplifies the system's usage immensely without reducing its benefits. As the two user groups (*i.e.*, insurers and customers), use different functions to interact with the system, it has been decided to implement two similar but yet different interfaces. The layout of both interfaces is cockpit-based, as the system includes a high potential for data actualization. Therefore, to avoid unnecessary user tasks and a struggling user experience, it was decided to define a simple layout that focuses on user interaction and information transparency.

After the development phase, the solution's feasibility has been proved with case studies. The case studies focused on simulating an insurer-customer relationship as realistic as it could be. All case studies have been executed in the same session and without the need to adjust or adapt anything, showing that the solution is feasible for that kind of use cases.

Although the developed tool proved its functionality and its simplicity in terms of usage, it has its limits and can evolve in different dimensions as future work. The introduced

dynamic address configuration is yet a simple proof-of-concept that the solution is capable of allowing individual address assignment. It can be improved in terms of permanent credentials and user settings by using a registration process. Also, the system struggles with estimating an accurate premium with an ongoing risk assessment. The update functionality enables a relatively dynamic risk assessment, which can be triggered by the customer when needed but could be improved. With the integration of actuarial databases or by using trends in terms of cyber risks and attacks, the premium calculation could only profit. However, the solution provides extensibility in terms of risk and premium estimation and further extensions could significantly improve the tool, making it more usable to address numerous kinds of potential scenarios and more complete. Additionally, machine learning techniques could be considered as future work to improve the cost estimation of the solution, thus allowing for a more precise premium calculation. Also, other blockchain implementations can be considered in order to reduce the cost of the solution while also increase the performance and security.

Bibliography

[1] Steve Morgan: Cybercrime To Cost The World $10.5 Trillion Annually By 2025, Cybercrime Magazine, https://cybersecurityventures.com/hackerpocalypse-cybercrime-report-2016/, November 13, 2020.

[2] Dan Burke: Cyber 101: Understand the Basics of Cyber Liability Insurance, WOODRUFF SAWYER, https://woodruffsawyer.com/cyber-liability/cyber-101-insurance-coverage-2021/, November 2, 2020.

[3] R. Pal, L. Golubchik, K. Psounis, P. Hui: Will Cyber-Insurance Improve Network Security? A Market Analysis, IEEE Conference on Computer Communications, 2014.

[4] V. Gatteschi, F. Lamberti, C. Demartini, C. Pranteda, V. Santamaria: Blockchain and Smart Contracts for Insurance: Is the Technology Mature Enough?, Future Internet, February 20, 2018.

[5] Emilio Granados Franco: These are the world's top 10 business risks - by region, World Economic Forum, https://www.weforum.org/agenda/2019/11/top-business-risks-2019-regional-cyberattacks-instability-energy/, November 1, 2019.

[6] Yakir Golan: The Next Five Years: Cyber Insurance Predictions Through 2025, Forbes, https://www.forbes.com/sites/theyec/2021/01/19/the-next-five-years-cyber-insurance-predictions-through-2025/, January 19, 2021.

[7] BUSINESS WIRE: $20+ Billion Cyber Insurance Market - Global Forecast to 2025 - ResearchAndMarkets.com, October 30, 2020.

[8] Nir Kshetri: The Economics of Cyber-Insurance, University of North Carolina, November/December, 2018.

[9] Markets and Markets: Cyber Insurance Market - Global Forecast to 2025, https://www.marketsandmarkets.com/Market-Reports/cyber-insurance-market-47709373.html, October, 2020.

[10] JÃ¼rgen Reinhart and Martin Kreuzer: Cyber insurance: Risks and trends 2020, Munich RE, https://www.munichre.com/topics-online/en/digitalisation/cyber/cyber-insurance-risks-and-trends-2020.html, April 14, 2020.

[11] Jeff Wargin: 8 Insurance Company Technology Trends Transforming the Industry in 2020, Duck Creek Technologies, https://insuretechtrends.com/8-insurance-technology-trends-transforming-the-industry-in-2020/, August 27, 2020.

[12] G. Ciocarlie, K. Eldefrawy, T. Lepoint: BlockCIS - A Blockchain-based Cyber Insurance System, IEEE International Conference on Cloud Engineering, 2018.

[13] Noah Berni: SC4CyberInsurance: Automated Cyber-Insurance Contracts, University of Zurich, Communication Systems Group, Department of Informatics, January 06, 2021.

[14] S. Dambra, L. Bilge, D. Balzarotti: SoK: Cyber Insurance - Technical Challenges and a System Security Roadmap, IEEE Symposium on Security and Privacy (SP), 2020.

[15] Ulrik Franke: The cyber insurance market in sweden, Computers & Security, 2017.

[16] Bruce Schneier: The dangers of a software monoculture, Information Security Magazine, 2010.

[17] A. Marotta, F. Martinelli, S. Nanni, A. Orlando, A. Yautsiukhin: Cyber-insurance survey, 2017.

[18] S. Romanosky, L. Ablon, A. Kuehn, T. Jones: Content analysis of cyber insurance policies: how do carriers price cyber risk?, Journal of Cybersecurity, 2019.

[19] C. Biener, M. Eling, J. H. Wirfs: Insurability of Cyber Risk: An Empirical Analysis, The Geneva Papers, 2015.

[20] P. Petratos, A. Sandberg, F. Zhou: Cyber Insurance, Handbook of Cyber-Development, Cyber-Democracy, and Cyber-Defense, 2018.

[21] Andrew Granato: The growth and challenges of cyber insurance, Chicago Fed Letter, 2019.

[22] Brian Carlson: The Microsoft Exchange Server hack: A timeline, CSO Online, [https://www.csoonline.com/article/3616699/the-microsoft-exchange-server-hack-a-timeline.html, May 6, 2021.

[23] Business Insurance Center: WHAT ARE FIRST-PARTY AND THIRD-PARTY COSTS IN A CYBER ATTACK?, https://www.businessinsurancecenter.com/what-are-first-party-and-third-party-costs-in-a-cyber-attack/, July 29, 2019.

[24] S. Panda, A. Farao, E. Panaousis, C. Xenakis: Cyber-Insurance: Past, Present and Future, Encyclopedia of Cryptography, Security and Privacy, 2021.

[25] Jameela Al-Jaroodi and Nader Mohamed: Industrial Applications of Blockchain, IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC), 2019.

[26] Satoshi Nakamoto: Bitcoin: A Peer-to-Peer Electronic Cash System, 2009.

[27] M. Nofer, P. Gomber, O. Hinz and D. Schiereck: Blockchain, Business & Information Systems Engineering 59, 2017.

[28] Z. Zheng, S. Xie, H. Dai, X. Chen and H. Wang: Blockchain challenges and opportunities: A survey, International Journal of Web and Grid Services, 2018.

[29] Karl WÃ¼st and Arthur Gervais: Do you need a Blockchain?, 2017.

[30] N. Szabo: The Idea of Smart Contracts, 1997.

[31] W. Zou, D. Lo, P. S. Kochhar, X. D. Le, X. Xia, Y. Feng, Z. Chen, B. Xu: Smart contract development: Challenges and opportunities, IEEE Transactions on Software Engineering, 2019.

[32] Arun Rajeevan: Tokens, Gas and Gas limit in Ethereum, `https://arunrajeevan.medium.com/tokens-gas-and-gas-limit-in-ethereum-f07790f56d8f`, February 11, 2019.

[33] A. Farao, S. Panda, S. Menesidou, E. Veliou, N. Episkopos, G. Kalatzantonakis, F. Mohammadi, N. Georgopoulos, M. Sirivianos, N. Salamanos, S. Loizou, M. Pingos, J. Polley, A. Fielder, E. Panaousis and C. Xenakis: SECONDO: A Platform for Cybersecurity Investments and Cyber Insurance Decisions, 2020.

[34] Brian Jefferson: The 15 Most Common Types of Cyber Attacks, Data Security & Compliance Blog, last updated on 29.06.2021.

[35] Muriel Franco, Noah Berni, Eder John Scheid, Bruno Rodrigues, Christian Killer, Burkhard Stiller: SaCI: a Blockchain-based Cyber Insurance Approach for the Deployment and Management of a Contract Coverage; 18th International Conference on the Economics of Grids, Clouds, Systems and Services (GECON 2021), Virtually, September 2021, pp. 1-14.

[36] Cipher: 10 Cybersecurity Metrics You Should Be Monitoring, `https://cipher.com/blog/10-cybersecurity-metrics-you-should-be-monitoring/`, last accessed August 2021.

[37] Brian Jefferson: The 15 Most Common Types of Cyber Attacks, Lepide, Data Security & Compliance Blog, June 08, 2021.

[38] Muriel Franco, Bruno Rodrigues, Burkhard Stiller: MENTOR: The Design and Evaluation of a Protection Services Recommender System; 15th International Conference on Network and Service Management (CNSM 2019), Halifax, Canada, October 2019, pp 1-7.

[39] Bruno Rodrigues, Muriel Franco, Geetha Parangi, Burkhard Stiller: SEConomy: A Framework for the Economic Assessment of Cybersecurity; 16th Conference on the Economics of Grids, Clouds, Systems, and Services (GECON 2019), Leeds, UK, September 2019, pp 1-13.

[40] Eder John Scheid, Bruno Rodrigues, Christian Killer, Muriel Franco, Sina Rafati Niya, Burkhard Stiller: Blockchains and Distributed Ledgers Uncovered: Clarifications, Achievements, and Open Issues; in: Michael Goedicke, Erich Neuhold, Kai Rannenberg (Edt.), "Advancing Research in Information and Communication Technology", Springer, Cham, Switzerland, No. 1, August 2021, ISBN 978-3-030-81701-5, pp 1-29.

Abbreviations

SC	Smart Contract
DDoS	Distributed Denial-of-Service
PoW	Proof-of-Work
EVM	Ethereum Virtual Machine
API	Application Programming Interface
CSIM	Cyber Security Investment Module
UI	User Interface
DOM	Document Object Model

List of Figures

Appendix A

Installation Guidelines

This chapter provides the necessary information to install and run the prototype of the solution. The following instructions have been executed in a Windows environment, but other operation systems should work similar.

Preconditions:

1. Install Python version >= 3
2. Install Ganache
3. Install the Node Package Manager (npm)
4. Download the source code from: `https://github.com/fimami/SC4CyberInsurance-WebInterface`
5. Install dependencies with `py -m pip install -r requirements.txt`
6. Make sure you have solidity compiler version 0.7.1 installed:
   `https://docs.soliditylang.org/en/v0.7.1/installing-solidity.html`
   OR edit the 2nd line of `SmartContractCode.py` according to the solidity compiler version you are using.

Starting the Server:

1. First, start Ganache and quickstart an ethereum local node.
2. Open a Command-Line-Interface (CLI) and navigate into the server folder of the source code. Execute following command: `py APIInsurer.py`
3. Open another CLI and navigate into the server folder of the source code. Execute following command: `py APICustomer.py`

Starting the Interface:

1. Make sure that the server is running. Open a new CLI and navigate into the client-customer folder of the project. To install all necessary packages, execute following command: `npm install`
2. Start the customer interface with following command: `npm start`
3. The customer interface will run on *localhost:3001*
4. Open a new CLI and navigate into the client-insurer folder of the project. Again install all necessary packages with `npm install`
5. Start the insurer interface with `npm start`
6. The insurer interface will run on *localhost:3002*
7. To start using the tool, copy paste a private key from one of the ganache accounts into the log in screen of the customer. Repeat the process for the insurer. (Do not use the same account for both)

Using the Exchange Rate Updater:

1. Make sure the server and clients are running. The insurer needs to be logged in.
2. Set up an Ethereum bridge by calling following command:
   `ethereum-bridge -H localhost:7545 -a 2`
   The last number defines the index of Ganache used as bridge account. Make sure to not use the same account for another user.
3. Call the command: `py ExchangeRateUpdater.py`
   in the prototype folder of the project.

Appendix B

Contents of the CD

- Presentation Slides
- Source code
- Thesis as PDF