Scuola di Scienze Matematiche, Fisiche e Naturali
Corso di Laurea Magistrale in Informatica

Tesi di Laurea Magistrale

PROGETTAZIONE DI UNO SMART CONTRACT A SUPPORTO DEL PROTOCOLLO DI FAIR EXCHANGE DI VERIOSS, UNA PIATTAFORMA BUG BOUNTY BASATA SULLA BLOCKCHAIN

DESIGN OF A SMART CONTRACT TO SUPPORT THE FAIR EXCHANGE PROTOCOL OF VERIOSS, A BLOCKCHAIN-BASED BUG-BOUNTY PLATFORM

FRANCESCO MUCCI

Relatore: Prof. *Rosario Pugliese*
Correlatori: Prof. *Gabriele Costa*, Dott. *Letterio Galletta*

Anno Accademico 2022-2023

A <Nome>,
<frase di dedica>.

"Le vent se lève!... il faut tenter de vivre!"
— Paul Valéry, Le Cimetière marin, 1920 [1].

"The best theory is inspired by practice
and the best practice is inspired by theory."
— Donald E. Knuth, Theory and practice, 1991 [2].

PREFAZIONE

Durante l'anno accademico 2019-2020 ho collaborato con l'unità di ricerca SySMA della Scuola IMT Alti Studi Lucca in qualità di beneficiario della borsa di ricerca **VeriOSS smart contract development** (finanziata con i fondi del progetto PAI 2018 "*VeriOSS: a security-by-smart contract verification framework for Open Source Software*" - P0137). L'obiettivo della borsa era quello di progettare e sviluppare smart contract Solidity a supporto del protocollo di fair exchange di VeriOSS, una piattaforma per la bug bounty basata sulla blockchain. Il lavoro di tesi svolto prosegue e conclude quanto iniziato durante la suddetta collaborazione di ricerca.

Tutto il materiale prodotto per questo lavoro di tesi è accessibile attraverso diverse repository pubbliche su GitHub; in particolare:

- i file LaTeX associati a questo documento di tesi si trovano in `github.com/FrancescoMucci/VeriOSS-thesis`;
- il codice implementato è disponibile in `github.com/FrancescoMucci/VeriOSS-challenge-reward`;
- infine, i diagrammi di sequenza, di stato e di classe sono raccolti in `github.com/FrancescoMucci/VeriOSS-diagrams`.

Per individuare e correggere involontarie somiglianze o citazioni non adeguate, è stato utilizzato *Turnitin*, il software antiplagio messo a disposizione dall'Università degli Studi di Firenze.

INDICE

ACRONIMI

ELENCO DELLE FIGURE

ELENCO DELLE TABELLE

ELENCO DEI CODICI

1

INTRODUZIONE

Lo scopo principale di questo capitolo è presentare il problema di ricerca trattato, illustrare gli obiettivi della tesi e delineare una panoramica del suo contenuto [3].

L'introduzione, insieme alle conclusioni, rappresenta uno dei capitoli più importanti dell'intera tesi: molti lettori si concentreranno principalmente su questi due e daranno un'occhiata rapida alle figure e alle tabelle presenti nel resto del lavoro [4].

1.1 CONTESTO

Per prima cosa, come indicato sia da Pfandzelter et al. [3] che da Zobel [5], presenteremo l'area di ricerca che fa da sfondo al nostro lavoro di tesi. In questa sezione forniremo dunque una breve panoramica del particolare campo di studio, evidenziandone la rilevanza e le motivazioni per cui merita attenzione.

1.2 PROBLEMA AFFRONTATO

In secondo luogo, introdurremo lo specifico problema di ricerca che si vuole affrontare e, anche in questo caso, argomenteremo al fine di mettere in evidenza la sua importanza [5].

1.3 STATO DELL'ARTE

Successivamente, come suggerito da Zobel [5], riassumeremo in modo sintetico le soluzioni standard al problema affrontato, enfatizzando le limitazioni di queste soluzioni per far comprendere meglio le motivazioni dietro al nostro lavoro di ricerca.

1.4 DOMANDE DI RICERCA

In questa sezione elencheremo esplicitamente le domande di ricerca che guideranno il nostro studio, cioè le domande che identificano le lacune nelle conoscenze esistenti che cercheremo di colmare [3]. Ad ogni domanda identificata sarà associato un obiettivo che la nostra tesi si prefigge di raggiungere; tali obiettivi indicano cosa abbiamo intenzione di dimostrare, sviluppare, testare o esplorare.

Sarà appropriato introdurre le domande di ricerca nell'introduzione solo se risultano chiare anche senza aver prima esaminato i lavori precedenti; qualora ciò non fosse possibile, potrebbe essere opportuno posticipare la loro presentazione [6].

1.5 APPROCCIO USATO

A questo punto, specificheremo l'approccio seguito per risolvere il problema affrontato, fornendo una breve descrizione della soluzione proposta. Zobel [5] ci ricorda di menzionare eventuali articoli usati come base di partenza per il lavoro svolto e di evidenziare le motivazioni per cui la soluzione fornita può essere ritenuta efficace, trattando in modo succinto le tecniche usate per valutarla.

È in questa sezione che specifichiamo come abbiamo risposto alle domande di ricerca illustrando brevemente le tecniche, i processi e i metodi utilizzati per raggiungere i nostri obiettivi [3].

1.6 CONTRIBUTI ORIGINALI

In questa penultima sezione, metteremo in evidenza i contributi originali dati dal nostro lavoro di tesi alla specifica area di ricerca [3]: andremo quindi a illustrare sinteticamente i nuovi metodi, le teorie, i modelli o le implementazioni software introdotte dal nostro lavoro.

Se il lavoro presentato è parte di un progetto più grande, è importante specificare chiaramente quale sia stato il nostro particolare contributo a tale progetto [4].

1.7 STRUTTURA DELLA TESI

Per concludere, delineeremo la struttura della tesi: per ogni capitolo forniremo una breve descrizione del suo contenuto e del suo contributo al lavoro complessivo [4].

Seguendo l'approccio proposto da Pfandzelter et al. [3], il restante lavoro di tesi è strutturato nei seguenti capitoli:

- Capitolo 2 - PRELIMINARI: introduce le nozioni fondamentali per la comprensione del lavoro svolto e inquadra la tesi collegandola a eventuali lavori precedenti.
- Capitolo 3 - APPROCCIO: illustra l'idea risolutiva proposta per il problema affrontato.
- Capitolo 4 - VALUTAZIONE: dimostra, attraverso un processo di valutazione, l'efficacia dell'approccio risolutivo adottato.
- Capitolo 5 - DISCUSSIONE: conduce un'analisi critica e oggettiva dell'approccio e del metodo di valutazione utilizzati.
- Capitolo 6 - LAVORI CORRELATI: esamina le ricerche correlate mettendole a confronto con il lavoro realizzato.
- Capitolo 7 - CONCLUSIONI: riassume in modo conciso quanto svolto (concentrandosi principalmente sui risultati ottenuti) e suggerisce possibili sviluppi futuri.

Oltre ai capitoli principali, potremmo decidere di includere nella nostra tesi delle appendici; queste dovrebbero contenere materiale che, pur essendo necessario per comprendere il lavoro svolto, non rappresenta una parte centrale della tesi o non può essere inserito nel testo principale a causa delle sue dimensioni eccessive o del formato particolare [4].

Ad esempio, potremmo inserire come appendice: codici sorgente sviluppati; dimostrazioni matematiche articolate; tabelle e figure di grandi dimensioni; diagrammi complessi; descrizioni dettagliate di processi di misura, degli esperimenti condotti e dei risultati sperimentali ottenuti; manuali d'uso e di manutenzione redatti; sondaggi condotti con relativi risultati [4].

La tesi include, oltre ai capitoli, anche le seguenti appendici:

- Appendice A - CODICI SORGENTE ADDIZIONALI:
- Appendice B - DIMOSTRAZIONI ADDIZIONALI:

2

PRELIMINARI

Gli obiettivi di questo capitolo sono i seguenti:

- dimostrare che lo studente possiede una comprensione completa dell'area di ricerca in cui opera [3];
- fornire al lettore le nozioni avanzate[1] essenziali per la comprensione della tesi [5];
- inquadrare la tesi nel suo contesto, collegandola ai lavori precedenti, pubblicati o inediti, che costituiscono la base per il progetto svolto [7].

Alla luce di quanto detto, sarà necessario:

- introdurre concetti, definizioni e terminologia che verranno utilizzati nel resto della tesi [3];
- presentare i lavori che costituiscono il fondamento del nostro progetto [7];
- descrivere i metodi e le tecniche che formano la base del nostro lavoro [8];
- illustrare l'hardware e il software impiegato [8].

1 In questo contesto, consideriamo come conoscenza comune tutto ciò che è stato trattato nei corsi obbligatori dello specifico corso di laurea [3].

2.1 INTRODUZIONE AL CAPITOLO

All'inizio di ogni capitolo includeremo una breve introduzione che fornisce contesto al capitolo stesso; questa introduzione faciliterà la transizione logica da un capitolo all'altro mostrando come quello corrente si colleghi ai precedenti [5].

Per essere più precisi, in ognuna di queste introduzioni forniremo [7]:

- gli obiettivi generali del capitolo, specificando cosa si intende affrontare e quale aspetto del nostro lavoro verrà esplorato;
- una spiegazione di come il capitolo corrente si inserisca nel contesto più ampio della tesi, collegandolo ai temi generali e agli obiettivi più ampi del nostro lavoro;
- un breve riassunto di come si è concluso il capitolo precedente e come quello corrente si costruisce sulle fondamenta del primo (ciò va fatto solo se pertinente);
- un sommario del contenuto del capitolo, fornendo, ad esempio, una concisa panoramica delle sezioni e sottosezioni presenti.

2.2 NOZIONI PRELIMINARI

Iniziamo con l'introduzione di concetti, definizioni e teorie fondamentali che costituiscono le basi del nostro lavoro. Questa sezione potrebbe includere, ad esempio, modelli matematici e teoremi essenziali per la comprensione della tesi.

È importante tenere a mente che non stiamo scrivendo un libro di testo: ogni volta che introduciamo un termine o un concetto sarà sufficiente fornirne una breve spiegazione e includere un riferimento bibliografico; in tal modo il lettore potrà approfondire autonomamente l'argomento [4].

2.3 LAVORI PRECEDENTI

Successivamente, se la tesi è una diretta continuazione di articoli o progetti di ricerca preesistenti, esporremo questi lavori evidenziando come la nostra ricerca si sviluppi a partire dalle loro fondamenta.

2.4 METODI E TECNICHE UTILIZZATE

In questa sezione descriveremo i metodi e le tecniche adottate nel corso della tesi, quali le analisi statistiche o i metodi di sviluppo software utilizzati per portare avanti il nostro progetto.

Nel caso di un progetto implementativo, descriveremo la metodologia di sviluppo impiegata (es. Agile, Waterfall, TDD, BDD) e come questa abbia influenzato il processo di implementazione.

2.5 TECNOLOGIE UTILIZZATE

Concludiamo il capitolo presentando le tecnologie, inclusi gli strumenti software e hardware, impiegati nella nostra ricerca, spiegando il loro ruolo e come hanno contribuito al raggiungimento degli obiettivi della tesi.

Nel caso di un progetto implementativo, in questa sezione elencheremo i linguaggi di programmazione, i framework, i database e altri strumenti utilizzati per lo sviluppo e il test del software. Ogni scelta andrà motivata illustrando quali sono i vantaggi per il nostro progetto.

2.6 RIASSUNTO DEL CAPITOLO E CONCLUSIONI

Alla fine di ogni capitolo includeremo un breve riassunto del suo contenuto, una riflessione su come quanto trattato contribuisca agli obiettivi generali della tesi e, per concludere, un'anticipazione di come i capitoli successivi faranno uso di quanto introdotto in quello corrente (in tal modo metteremo in evidenza come questi sono tra loro collegati) [5].

3

APPROCCIO

Questo è il primo dei due capitoli centrali in cui descriviamo il lavoro progettuale svolto. Il suo obiettivo principale è rispondere alla domanda: *"Come abbiamo risolto il problema di ricerca affrontato?"* [3].

Sarà qui che presenteremo, tramite una descrizione ad alto livello, la nostra idea risolutiva. A seconda della natura del problema affrontato, potremmo dunque trovarci a descrivere la progettazione di uno studio comparativo, l'architettura di un nuovo sistema, un nuovo algoritmo per la soluzione di un problema irrosolto oppure un algoritmo noto, ma che risolve un problema nuovo [3].

Di norma, se la nostra domanda di ricerca è del tipo *"Come posso risolvere questo problema?"*, la soluzione proposta dovrà essere presentata senza illustrare il processo iterativo che ci ha portato alla sua formulazione [3]. La situazione cambia se la domanda di ricerca è *"Quale tra i vari approcci noti è il migliore?"*: in questo caso, descriveremo come intendiamo eseguire il nostro studio comparativo, inserendo una descrizione dettagliata dei singoli approcci nel capitolo sulle nozioni preliminari [3].

La struttura di questo capitolo sarà fortemente influenzata dal tipo di progetto intrapreso e generalmente coprirà le prime fasi del suo sviluppo [7]: per un progetto implementativo, includeremo la specifica dei requisiti e una descrizione ad alto livello del design del software, facendo uso di strumenti come pseudocodice o diagrammi di flusso per facilitarne la comprensione.

3.1 INTRODUZIONE AL CAPITOLO

All'inizio di ogni capitolo includeremo una breve introduzione che fornisce contesto al capitolo stesso; questa introduzione faciliterà la transizione logica da un capitolo all'altro mostrando come quello corrente si colleghi ai precedenti [5].

Per essere più precisi, in ognuna di queste introduzioni forniremo [7]:

- gli obiettivi generali del capitolo, specificando cosa si intende affrontare e quale aspetto del nostro lavoro verrà esplorato;
- una spiegazione di come il capitolo corrente si inserisca nel contesto più ampio della tesi, collegandolo ai temi generali e agli obiettivi più ampi del nostro lavoro;
- un breve riassunto di come si è concluso il capitolo precedente e come quello corrente si costruisce sulle fondamenta del primo (ciò va fatto solo se pertinente);
- un sommario del contenuto del capitolo, fornendo, ad esempio, una concisa panoramica delle sezioni e sottosezioni presenti.

3.2 SPECIFICA DEI REQUISITI

In questa sezione verranno elencati i requisiti funzionali e non funzionali del nostro progetto implementativo. Specificheremo, in sostanza, cosa il nostro sistema è tenuto a fare, ma non come lo andrà a fare [9].

3.2.1 *Requisiti funzionali*

I requisiti funzionali specificano le funzionalità che il sistema dovrà essere in grado di eseguire. Questi requisiti delineano gli input e gli output, le funzioni eseguite dal sistema e i dati che esso deve gestire. Sono inclusi anche i dettagli sulle interfacce utente e le interazioni tra il sistema e altri sistemi [10].

3.2.2 *Requisiti non funzionali*

I requisiti non funzionali descrivono le caratteristiche qualitative generali del sistema software. Questi comprendono aspetti come le prestazioni, l'usabilità, la sicurezza, l'affidabilità, la disponibilità, la manutenibilità e la portabilità. Includono anche i vincoli entro cui ci si aspetta che il sistema finale operi, quali il sistema operativo, la velocità di elaborazione, la larghezza di banda della rete, la capacità di memoria e il linguaggio di programmazione utilizzato [10].

3.3 ARCHITETTURA DEL SISTEMA

In questa sezione descriviamo l'architettura del sistema software che intendiamo implementare.

3.3.1 *Design architetturale del sistema*

Illustreremo, anzitutto, la struttura del nostro sistema, evidenziando le relazioni tra i vari sottosistemi che lo compongono. Ci concentreremo su una visione ad alto livello, evitando di addentrarci eccessivamente nei dettagli specifici; per far ciò, forniremo una descrizione generale di come le responsabilità siano state suddivise e assegnate alle diverse componenti e di come queste interagiscono tra loro per realizzare le funzionalità desiderate. Al fine di rendere la trattazione più chiara, includeremo una rappresentazione visiva dell'architettura del sistema, utilizzando, ad esempio, un diagramma UML [11].

3.3.2 *Componente 1 del sistema*

Per ogni componente del sistema, forniremo un'analisi dettagliata delle sue responsabilità e delle sue interfacce di input e di output. Presenteremo, ove necessario, una descrizione dei suoi aspetti algoritmici e, inoltre, analizzeremo il modo in cui essa interagisce con le altre componenti, utilizzando ad esempio dei sequence diagram per illustrare ciascun caso d'uso [11].

Responsabilità

Interfacce

Dettagli algoritmici

Comportamento dinamico

3.3.3 *Componente n del sistema*

Responsabilità

Interfacce

Dettagli algoritmici

Comportamento dinamico

3.3.4 *Considerazioni sulle scelte architetturali*

In questa sezione spieghiamo le ragioni che hanno guidato la decomposizione del sistema nelle sue componenti [11]. Evidenzieremo, inoltre, eventuali misure di sicurezza integrate nell'architettura e strategie impiegate per assicurare prestazioni efficienti.

3.4 RIASSUNTO DEL CAPITOLO E CONCLUSIONI

Alla fine di ogni capitolo includeremo un breve riassunto del suo contenuto, una riflessione su come quanto trattato contribuisca agli obiettivi generali della tesi e, per concludere, un'anticipazione di come i capitoli successivi faranno uso di quanto introdotto in quello corrente (in tal modo metteremo in evidenza come questi sono tra loro collegati) [5].

4

VALUTAZIONE

Questo è il secondo dei due capitoli centrali in cui descriviamo il lavoro progettuale svolto. Il suo obiettivo principale è dimostrare, attraverso un processo di valutazione, l'efficacia dell'approccio risolutivo presentato nel capitolo precedente [3].

Il metodo di valutazione scelto dipenderà dalla specifica domanda di ricerca; alcuni metodi che potremmo adottare sono i seguenti [3]:

- **Implementazione**: nel caso in cui abbiamo progettato l'architettura di un nuovo sistema, possiamo dimostrarne la validità fornendo una sua implementazione; in tal caso, è essenziale che questa sia accompagnata da dei test che ne verifichino il comportamento o da dei benchmark che ne attestino il miglioramento rispetto a soluzioni esistenti.

- **Verifica formale**: forniamo una dimostrazione formale della correttezza del nostro approccio.

- **Simulazione**: testare la soluzione proposta in un ambiente controllato può essere un metodo efficace per valutarla; per esempio, potremmo creare un ambiente simulato per eseguire il nostro algoritmo o sistema; se scegliamo questa via, sarà cruciale definire accuratamente cosa misurare durante la simulazione al fine di ottenere dei risultati significativi.

In sostanza, a seconda del metodo di valutazione scelto, dovremo presentare: i risultati sperimentali; i teoremi e le relative dimostrazioni; l'analisi dei dati e le scoperte fatte [5].

La struttura di questo capitolo, esattamente come quella del precedente, dipenderà dal tipo di progetto intrapreso e, in genere, tratterà le fasi conclusive del suo sviluppo [7]: in un progetto implementativo,

presenteremo una discussione dettagliata dell'implementazione e delle sue criticità, includendo anche una descrizione dei test eseguiti.

4.1 INTRODUZIONE AL CAPITOLO

All'inizio di ogni capitolo includeremo una breve introduzione che fornisce contesto al capitolo stesso; questa introduzione faciliterà la transizione logica da un capitolo all'altro mostrando come quello corrente si colleghi ai precedenti [5].

Per essere più precisi, in ognuna di queste introduzioni forniremo [7]:

- gli obiettivi generali del capitolo, specificando cosa si intende affrontare e quale aspetto del nostro lavoro verrà esplorato;
- una spiegazione di come il capitolo corrente si inserisca nel contesto più ampio della tesi, collegandolo ai temi generali e agli obiettivi più ampi del nostro lavoro;
- un breve riassunto di come si è concluso il capitolo precedente e come quello corrente si costruisce sulle fondamenta del primo (ciò va fatto solo se pertinente);
- un sommario del contenuto del capitolo, fornendo, ad esempio, una concisa panoramica delle sezioni e sottosezioni presenti.

4.2 IMPLEMENTAZIONE

In questa sezione presenteremo un'analisi dettagliata dell'implementazione del sistema illustrato nel capitolo precedente.

4.2.1 *Implementazione componente 1*

Per ogni componente implementata, forniremo, anzitutto, una descrizione accurata delle sue sotto-componenti e delle risorse che usa o gestisce; dopodiché, spiegheremo come essa esegua i compiti necessari per adempiere alle proprie responsabilità.

4.2.2 *Implementazione componente n*

4.2.3 *Sfide implementative e soluzioni*

Illustreremo eventuali sfide incontrate durante il processo di implementazione e come queste siano state superate.

4.3 TEST

In questa sezione descriviamo i test effettuati per verificare che il sistema si comporti nel modo atteso.

4.3.1 *Test d'unità*

L'obiettivo dei test d'unità è verificare che le singole componenti funzionino correttamente quando isolate dal resto del sistema [12].

Test componente 1

Test componente n

4.3.2 *Test d'integrazione*

L'obiettivo dei test d'integrazione è verificare che due o più componenti del sistema, già testate individualmente in isolamento, continuino a funzionare correttamente una volta messe insieme e fatte interagire [12].

Test integrazione componenti 1 e 2

Test integrazione componenti n-1 e n

4.3.3 *Test end-to-end*

L'obiettivo dei test end-to-end è verificare che il nostro sistema operi correttamente durante l'interazione con un suo client, sia esso umano o un'altra applicazione, attraverso le interfacce fornite [12].

4.4 QUALITÀ DEI TEST

Descriviamo i metodi e le tecniche impiegati per valutare la qualità dei nostri test.

4.4.1 *Test coverage*

Illustreremo la metodologia adottata per monitorare il test coverage, ovvero la percentuale di linee di codice di ogni singola componente eseguite durante i test. Più alto sarà il test coverage e più bassa sarà la probabilità che la componente contenga dei bug non rilevati dai test [12].

È buona norma puntare a raggiungere un test coverage del 100%, utilizzando unicamente i test d'unità per la specifica componente; tuttavia, un coverage completo non garantisce di per sé la correttezza dei test (è condizione necessaria, ma non sufficiente): ci dice solamente che tutte le linee di codice sono state eseguite, ma non se le asserzioni dei test sono esaustive e appropriate [12].

4.4.2 *Mutation testing*

Per aumentare la fiducia nella qualità dei nostri test, possiamo utilizzare un framework di mutation testing per introdurre mutazioni nelle componenti testate e verificare che i nostri test d'unità rilevino queste mutazioni (fallendo) [12].

Dato che il mutation testing è un processo che richiede tempo, di solito viene utilizzato per valutare unicamente i test d'unità delle componenti che contengono la logica del sistema [12].

4.5 RISULTATI

In questa sezione illustriamo in modo chiaro e conciso quali sono i risultati ottenuti dalla nostra ricerca, evitando, almeno per il momento, qualsiasi tipologia di analisi critica [6].

Nel contesto di un progetto implementativo, presenteremo, anzitutto, una sintesi dei risultati ottenuti dai test. Dopodiché, illustreremo come l'implementazione realizzata soddisfi i requisiti specificati nel capitolo precedente e evidenzieremo i risultati chiave relativi alle prestazioni, alla scalabilità e alla manutenibilità del sistema.

4.6 RIASSUNTO DEL CAPITOLO E CONCLUSIONI

Alla fine di ogni capitolo includeremo un breve riassunto del suo contenuto, una riflessione su come quanto trattato contribuisca agli obiettivi generali della tesi e, per concludere, un'anticipazione di come i capitoli successivi faranno uso di quanto introdotto in quello corrente (in tal modo metteremo in evidenza come questi sono tra loro collegati) [5].

5

DISCUSSIONE

Questo capitolo si focalizza sull'analisi critica e obiettiva del lavoro svolto, esaminando sia l'approccio risolutivo al problema affrontato sia il metodo di valutazione impiegato per dimostrarne l'efficacia [3].

Per condurre questa analisi possiamo lasciarci guidare dalle successive domande [5]:

- In cosa il lavoro ha avuto successo?
- E in cosa invece ha fallito?
- Quali problemi non sono stati risolti?
- Quali nuove domande sono emerse?
- Quali approcci alternativi avremmo potuto considerare?
- Quali sono le implicazioni dei risultati ottenuti?

Altre domande che possono venirci in aiuto sono le seguenti [4]:

- I risultati ottenuti corrispondono agli obiettivi iniziali?
- Siamo riusciti a rispondere alle nostre domande di ricerca?
- Qual'è l'importanza scientifica e pratica dei risultati ottenuti?

Inizieremo, quindi, l'analisi valutando se i risultati ottenuti corrispondono agli obiettivi prefissati della nostra ricerca e se abbiamo efficacemente risposto alle domande di ricerca poste. Successivamente, valuteremo le debolezze e le limitazioni del nostro lavoro, esplorando anche le questioni ancora aperte, le nuove domande emerse e gli eventuali approcci alternativi che avremmo potuto esplorare. Concluderemo discutendo

l'importanza scientifica e pratica dei nostri risultati, evidenziando come questi possano essere applicati nella pratica e contribuire alle teorie esistenti.

Se ritenuto necessario, questo capitolo può essere convertito in una sezione del capitolo conclusivo; tale sezione andrebbe posizionata subito dopo il riassunto della tesi [5].

5.1 INTRODUZIONE AL CAPITOLO

All'inizio di ogni capitolo includeremo una breve introduzione che fornisce contesto al capitolo stesso; questa introduzione faciliterà la transizione logica da un capitolo all'altro mostrando come quello corrente si colleghi ai precedenti [5].

Per essere più precisi, in ognuna di queste introduzioni forniremo [7]:

- gli obiettivi generali del capitolo, specificando cosa si intende affrontare e quale aspetto del nostro lavoro verrà esplorato;
- una spiegazione di come il capitolo corrente si inserisca nel contesto più ampio della tesi, collegandolo ai temi generali e agli obiettivi più ampi del nostro lavoro;
- un breve riassunto di come si è concluso il capitolo precedente e come quello corrente si costruisce sulle fondamenta del primo (ciò va fatto solo se pertinente);
- un sommario del contenuto del capitolo, fornendo, ad esempio, una concisa panoramica delle sezioni e sottosezioni presenti.

5.2 OBIETTIVI RAGGIUNTI

In questa sezione valuteremo se i risultati ottenuti corrispondono agli obiettivi prefissati e se siamo riusciti a rispondere alle nostre domande di ricerca.

Non sarà necessario illustrare nuovamente i risultati in quanto questi dovrebbero già essere stati trattati nel capitolo *Valutazione* [3].

5.3 DEBOLEZZE E LIMITAZIONI

Valutiamo le debolezze e le limitazioni del nostro lavoro.

5.4 QUESTIONI IRRISOLTE

Illustriamo eventuali questioni irrisolte.

5.5 NUOVE DOMANDE EMERSE

Evidenziamo le nuove domande che sono emerse dalla ricerca svolta.

5.6 APPROCCI ALTERNATIVI

Trattiamo approcci alternativi che avremmo potuto prendere in considerazione.

5.7 IMPATTO SCIENTIFICO E PRATICO DEI RISULTATI

Discutiamo infine dell'importanza scientifica e pratica dei risultati ottenuti, riflettendo sulle loro potenziali implicazioni e ripercussioni. Considereremo, quindi, come questi risultati possano essere applicati nella pratica o contribuire a teorie esistenti.

Per scrivere questa sezione, possiamo provare a rispondere alla seguente domanda: "Quali lezioni o scoperte fatte durante il nostro lavoro di ricerca possono essere applicate in altri contesti?" [6].

5.8 RIASSUNTO DEL CAPITOLO E CONCLUSIONI

Alla fine di ogni capitolo includeremo un breve riassunto del suo contenuto, una riflessione su come quanto trattato contribuisca agli obiettivi generali della tesi e, per concludere, un'anticipazione di come i capitoli successivi faranno uso di quanto introdotto in quello corrente (in tal modo metteremo in evidenza come questi sono tra loro collegati) [5].

6

LAVORI CORRELATI

L'obiettivo di questo capitolo è di sottolineare l'originalità e la rilevanza del nostro lavoro di tesi rispondendo alle seguenti domande chiave [13]:

- Quali sono le origini delle nostre idee?
- Sono state pubblicate o proposte idee simili precedentemente?
- Quali sono gli aspetti originali del nostro lavoro?

In sostanza, il capitolo serve a mostrare che non ci siamo limitati a reinventare la ruota [4]. Per far ciò, esamineremo altri studi presenti in letteratura che cercano di risolvere lo stesso problema di ricerca o uno correlato [3], mettendo in evidenza le similitudini e le differenze rispetto al nostro approccio [4].

Ogni lavoro correlato verrà affrontato separatamente seguendo questa metodologia: [3]:

1. riassumiamo l'idea principale dello studio;
2. analizziamo i punti di forza, le limitazioni e i difetti (anche in confronto con il nostro approccio);
3. evidenziamo se e come il nostro studio sia stato influenzato dal lavoro esaminato.

Tireremo le fila dell'analisi evidenziando se emergono delle tendenze e identificando sia l'approccio più diffuso sia quello raccomandato per affrontare il problema di ricerca in questione. Per concludere, sottolineeremo le nuove conoscenze prodotte dal nostro lavoro di ricerca [4].

Sebbene questo capitolo possa essere posizionato sia dopo i preliminari teorici e tecnici sia prima delle conclusioni, seguendo il consiglio

di Pfandzelter et al. [3], la seconda opzione è preferibile per evitare di presentare i lavori correlati prima che il lettore abbia acquisito una piena comprensione dell'approccio utilizzato nel nostro lavoro di tesi. Alternativamente, se l'analisi dei lavori correlati è sufficientemente breve, può essere direttamente integrata nel capitolo sui preliminari o in quello contenente la discussione [6].

6.1 INTRODUZIONE AL CAPITOLO

All'inizio di ogni capitolo includeremo una breve introduzione che fornisce contesto al capitolo stesso; questa introduzione faciliterà la transizione logica da un capitolo all'altro mostrando come quello corrente si colleghi ai precedenti [5].

Per essere più precisi, in ognuna di queste introduzioni forniremo [7]:

- gli obiettivi generali del capitolo, specificando cosa si intende affrontare e quale aspetto del nostro lavoro verrà esplorato;
- una spiegazione di come il capitolo corrente si inserisca nel contesto più ampio della tesi, collegandolo ai temi generali e agli obiettivi più ampi del nostro lavoro;
- un breve riassunto di come si è concluso il capitolo precedente e come quello corrente si costruisce sulle fondamenta del primo (ciò va fatto solo se pertinente);
- un sommario del contenuto del capitolo, fornendo, ad esempio, una concisa panoramica delle sezioni e sottosezioni presenti.

6.2 PANORAMICA SULLO STATO DELL'ARTE

In questa sezione descriviamo i progressi e le scoperte principali compiute fino ad oggi nel campo di ricerca pertinente al nostro lavoro, stabilendo così il contesto per l'analisi dei lavori correlati.

6.3 LAVORI DEBOLMENTE CORRELATI

Procediamo con un'analisi dei lavori che hanno un legame indiretto o generale con il problema di ricerca che stiamo affrontando.

6.3.1 *Lavoro debolmente correlato 1*

Idea principale

Punti di forza

Limitazioni e difetti

Influenza sul nostro lavoro

6.3.2 *Lavoro debolmente correlato 2*

Idea principale

Punti di forza

Limitazioni e difetti

Influenza sul nostro lavoro

6.4 LAVORI STRETTAMENTE CORRELATI

Esaminiamo poi in dettaglio i lavori che si propongono di risolvere il problema di ricerca da noi affrontato e che adottano un approccio simile al nostro.

6.4.1 *Lavoro strettamente correlato 1*

Idea principale

Punti di forza

Limitazioni e difetti

Influenza sul nostro lavoro

6.4.2 *Lavoro strettamente correlato 2*

Idea principale

Punti di forza

Limitazioni e difetti

Influenza sul nostro lavoro

6.5 TENDENZE IDENTIFICATE

Evidenziamo le eventuali tendenze emerse a seguito dell'analisi dei lavori correlati e identifichiamo sia l'approccio più diffuso sia quello raccomandato per affrontare il particolare problema di ricerca.

6.6 LACUNE NELLA LETTERATURA E NOSTRO CONTRIBUTO

Identifichiamo le lacune nella letteratura esistente e mostriamo come il nostro lavoro miri a colmarle, mettendo in tal modo in evidenza il contributo originale apportato dalla nostra ricerca.

6.7 RIASSUNTO DEL CAPITOLO E CONCLUSIONI

Alla fine di ogni capitolo includeremo un breve riassunto del suo contenuto, una riflessione su come quanto trattato contribuisca agli obiettivi generali della tesi e, per concludere, un'anticipazione di come i capitoli successivi faranno uso di quanto introdotto in quello corrente (in tal modo metteremo in evidenza come questi sono tra loro collegati) [5].

7

CONCLUSIONI

Il capitolo conclusivo, insieme all'introduzione, rappresenta uno dei capitoli più importanti dell'intera tesi: molti lettori si concentreranno principalmente su questi due e daranno un'occhiata rapida alle figure e alle tabelle presenti nel resto del lavoro [4].

7.1 RIASSUNTO DELLA TESI

Il capitolo conclusivo della tesi si apre comunemente con un riassunto sintetico del lavoro svolto che si sofferma soprattutto sui risultati chiave raggiunti [8]. Questa sezione dovrà quindi contenere:

1. un riepilogo del problema indagato [3];
2. una descrizione essenziale del metodo adottato per risolverlo [3];
3. una sintesi dei risultati conseguiti [5] e del contributo fornito dal nostro lavoro di tesi [13].

7.2 SVILUPPI FUTURI

È essenziale dedicare una sezione alla prospettiva di ulteriori ricerche che possano estendere o approfondire lo studio presente. Questa parte dovrebbe concentrarsi principalmente sugli aspetti ancora da esplorare, piuttosto che sulle metodologie da adottare [8]. Potrebbe includere, per esempio, proposte non implementate in questa tesi o strategie per superare le limitazioni e le debolezze individuate [3].

A

CODICI SORGENTE ADDIZIONALI

Le appendici dovrebbero includere materiale che, pur essendo necessario per comprendere il lavoro svolto, non rappresenta una parte centrale della tesi o non può essere inserito nel testo principale a causa delle sue dimensioni eccessive o del formato particolare [4].

A.1 INTRODUZIONE ALL'APPENDICE

Poiché è possibile che il lettore consulti le appendici senza aver letto integralmente la tesi, è consigliabile includere in ognuna di queste una breve introduzione che ne descriva il contenuto e che la collochi nel contesto più ampio del lavoro svolto [4].

A.2 CODICE ADDIZIONALE 1

A.3 CODICE ADDIZIONALE 2

A.4 CODICE ADDIZIONALE 3

B

DIMOSTRAZIONI ADDIZIONALI

Le appendici dovrebbero includere materiale che, pur essendo necessario per comprendere il lavoro svolto, non rappresenta una parte centrale della tesi o non può essere inserito nel testo principale a causa delle sue dimensioni eccessive o del formato particolare [4].

B.1 INTRODUZIONE ALL'APPENDICE

Poiché è possibile che il lettore consulti le appendici senza aver letto integralmente la tesi, è consigliabile includere in ognuna di queste una breve introduzione che ne descriva il contenuto e che la collochi nel contesto più ampio del lavoro svolto [4].

B.2 DIMOSTRAZIONE ADDIZIONALE 1

B.3 DIMOSTRAZIONE ADDIZIONALE 2

B.4 DIMOSTRAZIONE ADDIZIONALE 3

TEST DELLA BIBLIOGRAFIA

INTRODUZIONE ALL'APPENDICE

In questa appendice, riservata unicamente alla bozza della tesi, vengono presentati i riferimenti bibliografici consultati, organizzati in base all'argomento e alla tipologia di documento.

VERIOSS

Articoli scientifici di Costa et al.

- VeriOSS: Using the Blockchain to Foster Bug Bounty Programs [14];
- Verifying a Blockchain-Based Remote Debugging Protocol for Bug Bounty [15].

PIATTAFORME BUG BOUNTY

Tesi di dottorato di Walshe e articoli scientifici di Walshe et al.

- Supporting data-driven software development life-cycles with bug bounty programmes [16];
- Current State of Bug Bounty Programmes and Platforms [17];
- An Empirical Study of Bug Bounty Programs [18].

Articoli scientifici di Akgul et al.

- Bug hunters' perspectives on the challenges and benefits of the bug bounty ecosystem [19];
- The Hackers' Viewpoint: Exploring Challenges and Benefits of Bug-Bounty Programs [20].

Altri articoli scientifici

- Bug Bounty Programs for Cybersecurity: Practices, Issues, and Recommendations [21];
- Web Science Challenges in Researching Bug Bounties [22].

PIATTAFORME BUG BOUNTY BASATE SULLE BLOCKCHAIN

Articoli scientifici di Hoffman et al. su Bountychain

- Decentralized Security Bounty Management on Blockchain and IPFS [23];
- Bountychain: Toward Decentralizing a Bug Bounty Program with Blockchain and IPFS [24].

Articoli scientifici di Badash et al. su BBBB Framework

- Blockchain-Based Bug Bounty Framework [25].

Articoli scientific di Lisi et al. su ARD

- Automated Responsible Disclosure of Security Vulnerabilities [26].

PROTOCOLLI DI FAIR EXCHANGE

Articoli scientifici seminali

- Optimistic Protocols for Multi-Party Fair Exchange [27];
- Fair exchange with a semi-trusted third party [28];
- Optimistic fair exchange of digital signatures [29];
- Secure group barter: Multi-party fair exchange with semi-trusted neutral parties [30].

Articoli scientifici panoramici

- A review of fair exchange protocols [31];
- A survey on optimistic fair exchange protocol and its variants [32];
- Fair Exchange Protocol in Electronic Transactions Revisited [33].

PROTOCOLLI DI FAIR EXCHANGE BASATI SULLA BLOCKCHAIN

Articoli scientifici su FairSwap

- FairSwap: How To Fairly Exchange Digital Goods [34];
- Privacy-preserving FairSwap: Fairness and privacy interplay [35].

Articoli scientifici su OptiSwap

- OptiSwap: Fast Optimistic Fair Exchange [36];
- Privacy-enhanced OptiSwap [37].

Articoli scientifici su cost fairness

- Cost Fairness for Blockchain-Based Two-Party Exchange Protocols [38];
- Formalizing Cost Fairness for Two-Party Exchange Protocols using Game Theory and Applications to Blockchain [39];
- Formalizing Cost Fairness for Two-Party Exchange Protocols using Game Theory and Applications to Blockchain (Extended Version) [40].

Articoli scientifici su protocolli che usano zero-knowledge proof

- FileBounty: Fair Data Exchange [41];
- Contingent payments from two-party signing and verification for abelian groups [42].

Altri articoli scientifici

- FairTrade: Efficient Atomic Exchange-based Fair Exchange Protocol for Digital Data Trading [43].

PROOF OF KNOWLEDGE

Monografie

- Proofs, Arguments, and Zero-Knowledge [44].

Capitoli di libri

- Sigma Protocols and Efficient Zero-Knowledge [45];
- Identification and signatures from Sigma protocols [46];
- Proving properties in zero-knowledge [47];
- A Survey on Zero-Knowledge Proofs [48].

Articoli scientifici seminali

- The Knowledge Complexity of Interactive Proof-Systems [49].

Articoli scientifici

- Do You Need a Zero Knowledge Proof? [50];
- A survey on zero knowledge range proofs and applications [51].

PROOF OF KNOWLEDGE PER LA BLOCKCHAIN

Articoli scientifici panoramici

- Overview of Zero-Knowledge Proof and Its Applications in Blockchain [52];
- Non-Interactive Zero-Knowledge for Blockchain: A Survey [53];
- A Survey on Zero-Knowledge Proof in Blockchain [54].

FONDAMENTI DI BLOCKCHAIN, ETHEREUM E SOLIDITY

Libri generici sulla blockchain

- Handbook on Blockchain [55];
- Blockchain Essentials - Core Concepts and Implementations [56].

Libri specifici per Ethereum e sviluppo di smart contracts Solidity

- Mastering Ethereum: Building Smart Contracts and DApps [57];
- Ethereum Smart Contract Development in Solidity [58];
- Blockchain and Ethereum Smart Contract Solution Development - Dapp Programming with Solidity [59];
- Solidity Programming Essentials: A guide to building smart contracts and tokens using the widely used Solidity language [60].

Documentazione di Ethereum e Solidity

- Ethereum Development Documentation [61];
- Solidity Documentation - Release 0.8.18 [62].

White e yellow paper

- Bitcoin: A peer-to-peer electronic cash system [63];
- Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform [64];
- Ethereum: A secure decentralised generalised transaction ledger [65].

Lavori seminali

- Pricing via Processing or Combatting Junk Mail [66];
- Smart Contracts [67];
- Formalizing and securing relationships on public networks [68];
- b-money [69];
- Karma: A secure economic framework for peer-to-peer resource sharing [70];
- RPOW - Reusable Proofs of Work [71];
- Bit Gold [72].

Libro e articoli scientifici di Xu et al.

- Architecture for Blockchain Applications [73];
- A Pattern Collection for Blockchain-based Applications [74];
- Applying Design Patterns in Smart Contracts [75];
- A Taxonomy of Blockchain-Based Systems for Architecture Design [76].

Tesi di dottorato di Wöhrer e articoli scientifici di Wöhrer et al.

- Engineering Blockchain-Based Applications in the Context of the Ethereum Ecosystem [77];
- Design Patterns for Smart Contracts in the Ethereum Ecosystem [78];
- Smart contracts: security patterns in the ethereum ecosystem and solidity[79];
- Architectural Design Decisions for Blockchain-Based Applications [80];
- Architecture Design of Blockchain-Based Applications [81].

Articoli scientifici di Marchesi et al.

- Design Patterns for Gas Optimization in Ethereum [82];
- ABCDE–agile block chain DApp engineering [83];
- An Agile Software Engineering Method to Design Blockchain Applications [84].

Altri articoli scientifici - architettura

- Do you Need a Blockchain? [85].

Altri articoli scientifici - revisioni sistematiche

- A systematic literature review of blockchain and smart contract development: Techniques, tools, and open challenges [86];
- A comprehensive survey on smart contract construction and execution: paradigms, tools, and systems [87];
- Ethereum Smart Contract Analysis Tools: A Systematic Review [88].

Altri articoli scientifici - design pattern

- Challenges and Common Solutions in Smart Contract Development [89];
- Some Blockchain Design Patterns for Overcoming Immutability, Chain-Boundedness, and Gas Fees [90];
- Towards saving money in using smart contracts [91].

Altri articoli scientifici - gas cost

- Computing Exact Worst-Case Gas Consumption for Smart Contracts [92];
- Profiling gas consumption in solidity smart contracts [93];
- Reduction in Gas Cost for Blockchain Enabled Smart Contract [94].

ORACOLI BLOCKCHAIN

Introduzione agli oracoli blockchain

- A Study of Blockchain Oracles [95];

Design pattern per oracoli blockchain

- Blockchain Patterns [96];
- Foundational Oracle Patterns: Connecting Blockchain to the Off-Chain World [97];
- Off-chain Data Fetching Architecture for Ethereum Smart Contract [98].

Confronto tra oracoli blockchain

- Trustworthy Blockchain Oracles: Review, Comparison, and Open Research Challenges [99];
- From trust to truth: Advancements in mitigating the Blockchain Oracle problem [100];
- Connect API with Blockchain: A Survey on Blockchain Oracle Implementation [101].

Provable (Oraclize)

- Provable Documentation [102].

Chainlink

- Chainlink Docs [103];
- Chainlink 2.0: Next Steps in the Evolution of Decentralized Oracle Networks [104];
- Chainlink Off-chain Reporting Protocol [105].

OFF-CHAIN DATA STORAGES

Confronto tra on-chain e off-chain data storages

- An Overview of Blockchain Scalability for Storage [106];
- Performance Comparison of On-Chain and Off-Chain Data Storage Model Using Blockchain Technology [107].

Confronto tra diverse soluzioni per off-chain data storage

- Cost and Performance Analysis on Decentralized File Systems for Blockchain-Based Applications: State-of-the-Art Report [108];
- Blockchain-Based Distributed File System Security and Privacy: A Systematic Mapping Study [109].

Documentazione e articoli scientifici ufficiali di IPFS

- IPFS Documentation [110];
- IPFS - Content Addressed, Versioned, P2P File System [111];
- Design and evaluation of IPFS: a storage layer for the decentralized web [112].

Altri articoli scientifici su IPFS

- Toward Decentralized Cloud Storage With IPFS: Opportunities, Challenges, and Future Considerations [113];
- IPFS: An Off-Chain Storage Solution for Blockchain [114].

FONDAMENTI DI VERIFICA FORMALE

Libri di testo

- Handbook of Model Checking [115];
- Handbook of Satisfiability [116];
- Logic: Reference Book for Computer Scientists [117].

Nozioni di base

- Software Verification [118];
- Predicate Abstraction for Program Verification [119];
- Control flow analysis [120];
- Propositional SAT Solving [121];
- Sentential Logic (SL) [122];
- On Sentences Which are True of Direct Unions of Algebras [123].

Model checking

- Model checking [124];
- 2^5 Years of Model Checking [125].

Satisfiability Modulo Theories (SMT)

- Satisfiability Modulo Theories [126];
- Satisfiability Modulo Theories [127];
- A Survey of Satisfiability Modulo Theory [128];
- A Tutorial on Satisfiability Modulo Theories [129].

Bounded Model Checking (BMC)

- SAT-Based Model Checking [130];
- Bounded Model Checking [131].

Lavori seminali su BMC

- Bounded model checking using satisfiability solving [132];
- SMT-Based Bounded Model Checking for Embedded ANSI-C Software [133].

Verifica di programmi e clausole di Horn

- Program Verification with Constrained Horn Clauses [134];
- Horn Clause Solvers for Program Verification [135];
- Analysis and Transformation of Constrained Horn Clauses for Program Verification [136];

Lavori seminali su Horn SAT

- Linear-time algorithms for testing the satisfiability of propositional horn formulae [137];
- Algorithms for testing the satisfiability of propositional formulae [138].

VERIFICA FORMALE DI SMART CONTRACT

Revisioni sistematiche

- Formal Verification of Smart Contracts [139];
- A Survey of Smart Contract Formal Specification and Verification [140];
- Formal Methods for the Verification of Smart Contracts: A Review [141];
- Formally Verifying a Real World Smart Contract [142].

Documentazione e articoli scientifici su SMTChecker di Solidity

- Solidity Documentation - SMTChecker and Formal Verification [143]
- A Solicitous Approach to Smart Contract Verification [144];
- Accurate Smart Contract Verification Through Direct Modelling [145];
- SMT-Based Verification of Solidity Smart Contracts [146];
- SolCMC: Solidity Compiler's Model Checker [147].

DEBUGGING

Revisioni sistematiche

- Debugging: a review of the literature from an educational perspective [148];
- A Systematic Review on Program Debugging Techniques [149].

Remote debugging

- Mercury: Properties and Design of a Remote Debugging Solution using Reflection [150];
- Remote Debugging for Containerized Applications in Edge Computing Environments [151].

Reverse debugging

- A review of reverse debugging [152];
- Implementation of Live Reverse Debugging in LLDB [153].

WEAKEST PRECONDITION CALCULUS

Articoli scientifici seminali

- Guarded commands, nondeterminacy and formal derivation of programs [154].

Libri di testo

- A Discipline of Programming [155];
- The Science of Programming [156];
- Predicate Calculus and Program Semantics [157].

Altri articoli scientifici

- The Weakest Precondition Calculus: Recursion and Duality [158].

SYMBOLIC EXECUTION

Revisioni sistematiche

- A Survey of Symbolic Execution Techniques [159];
- Advances in Symbolic Execution [160];
- Symbolic Execution and Recent Applications to Worst-Case Execution, Load Testing, and Security Analysis [161].

Revisioni di tools

- Benchmarking the Capability of Symbolic Execution Tools with Logic Bombs [162];
- Concolic Execution on Small-Size Binaries: Challenges and Empirical Study [163];
- Systematic comparison of symbolic execution systems: intermediate representation and its generation [164].

Altri articoli scientifici

- Symbolic execution formally explained [165].

SYMBOLIC EXECUTION CON ANGR

Documentazione

- angr: The angr Project [166].

Articoli scientifici

- SOK: (State of) The Art of War: Offensive Techniques in Binary Analysis [167];

- Driller: Augmenting Fuzzing Through Selective Symbolic Execution [168];
- Firmalice - Automatic Detection of Authentication Bypass Vulnerabilities in Binary Firmware [169].

Altri articoli scientifici

- Teaching with angr: A Symbolic Execution Curriculum and CTF [170];
- Tutorial: An Overview of Malware Detection and Evasion Techniques [171].

BACKWARD SYMBOLIC EXECUTION

Backward symbolic execution via weakest precondition calculus

- Snugglebug: a powerful approach to weakest preconditions [172];
- Handling Heap Data Structures in Backward Symbolic Execution [173];
- Higher-order demand-driven symbolic evaluation [174];
- Backward Symbolic Execution with Loop Folding [175];
- Generation of the weakest preconditions of programs with dynamic memory in symbolic execution [176].

BIBLIOGRAFIA

[1] Paul Valéry: *Il cimitero marino*. Interlinea edizioni, 2016, ISBN 9788868570880. Pubblicato per la prima volta nel 1920 con il titolo *Le Cimetière marin*.

[2] Donald E. Knuth: *Theory and Practice*. Theoretical Computer Science (Elsevier), vol. 90 (no. 1): pp. 1–15, novembre 1991. https://doi.org/10.1016/0304-3975(91)90295-D.

[3] Tobias Pfandzelter, Martin Grambow, Trever Schirmer e David Bermbach: *Writing a Computer Science Thesis*. Pubblicato dal gruppo di ricerca *Mobile Cloud Computing* della *TU Berlin* sulla pagina web del gruppo, dicembre 2022. https://github.com/pfandzelter/thesis-tips.

[4] *Guide to Writing a Thesis in Technical Fields*. Publicato dalla *Tampere University* sulla pagina web dedicata alla tesi per i corsi magistrali in ambito tecnologico, gennaio 2019. https://content-webapi.tuni.fi/proxy/public/2019-10/tau_thesis_guide_for_technical_fields_2019_version-3-1.pdf.

[5] Justin Zobel: *Writing for Computer Science*. Springer Publishing Company, terza edizione, 2015, ISBN 1447166388. https://doi.org/10.1007/978-1-4471-6639-9.

[6] Tomi Männistö, Juha Tiihonen e Fabian Fagerholm: *Scientific Writing - Guide of the Empirical Software Engineering Research Group of the University of Helsinki*. Publicato dall'*Empirical Software Engineering Research Group* della *Università di Helsinki* sulla pagina web del gruppo, novembre 2022. https://cs.helsinki.fi/group/ese/ScientificWritingGuide.pdf.

[7] *Master in Computer Science - Guidelines for the Thesis*. Publicato dalla *Libera Università di Bolzano* sulla pagina web dedicata alla tesi per i corsi magistrali affini all'informatica, 2022. https://guide.unibz.it/assets/graduation/Computer-Science/Master/Guidelines-Thesis-Master-2022.pdf.

[8] *Writing Your Thesis*. Pagina web dedicata alle linee guida per la tesi presente sul sito del laboratorio *Computer Science 7 (Computer Networks and Communication Systems)* della *Friedrich-Alexander Universität*. https://cs7.tf.fau.eu/teaching/student-theses/writing-your-thesis/, consultata in data 16 novembre 2023.

[9] William D. Shoaff: *How to Write a Master's Thesis in Computer Science*. Pubblicato sulla pagina web *Guides to Research* del Prof. Shoaff, ospitata sul sito del Dipartimento di Informatica del *Florida Institute of Technology*, agosto 2001. https://cs.fit.edu/~wds/guides/howto/howto.html.

[10] Narayanan Subramanian: *Requirements Specification and Analysis*. Nel testo di Benjamin W. Wah (curatore): *Wiley Encyclopedia of Computer Science and Engineering, 5 Volume Set*, capitolo R. Wiley, febbraio 2009.

[11] *Software Design Report*. Pubblicato sul sito del Dipartimento di Ingegneria Informatica della *Harran University*. https://web.harran.edu.tr/assets/uploads/other/files/bilgisayar/files/Software_Design_Document_Template.pdf, consultata in data 17 novembre 2023.

[12] Lorenzo Bettini: *Test-Driven Development, Build Automation, Continuous Integration with Java, Eclipse and friends*. Leanpub, febbraio 2021. https://leanpub.com/tdd-buildautomation-ci.

[13] Luca Aceto: *How to Write a Paper*. Slides pubblicate sulla pagina web dedicata ai consigli su come fare ricerca dell'*Icelandic Center of Excellence in Theoretical Computer Science*. http://icetcs.ru.is/luca/howto-lectures/howtowrite-gssi.pdf, consultata in data 17 novembre 2023.

[14] Andrea Canidio, Gabriele Costa e Letterio Galletta: *VeriOSS: Using the Blockchain to Foster Bug Bounty Programs*. Nel *2nd International Conference on Blockchain Economics, Security and Protocols (Tokenomics 2020)*, volume 82 della serie *Open Access Series in Informatics (OASIcs)*, pagine 6:1–14. Schloss Dagstuhl, Leibniz-Zentrum für Informatik, Germania, febbraio 2021. https://doi.org/10.4230/OASIcs.Tokenomics.2020.6.

[15] Pierpaolo Degano, Letterio Galletta e Selene Gerali: *Verifying a Blockchain-Based Remote Debugging Protocol for Bug Bounty*. Nel

Protocols, Strands, and Logic, volume 13066 della serie *Lecture Notes in Computer Science*, pagine 124–138. Springer, novembre 2021. https://doi.org/10.1007/978-3-030-91631-2_7.

[16] Thomas J. Walshe: *Supporting Data-driven Software Development Lifecycles with Bug Bounty Programmes*. Tesi di dottorato, University of Oxford, Wolfson College, Inghilterra, giugno 2023. https://ora.ox.ac.uk/objects/uuid:4a828bbb-8ff4-4cac-9e09-5699b30c6d52.

[17] Thomas J. Walshe: *Current State of Bug Bounty Programmes and Platforms*. Nel *Supporting data-driven software development life-cycles with bug bounty programmes*, tesi di dottorato, cap. 3, pagine 62–99. University of Oxford, Wolfson College, Inghilterra, giugno 2023. https://ora.ox.ac.uk/objects/uuid:4a828bbb-8ff4-4cac-9e09-5699b30c6d52.

[18] Thomas J. Walshe e Andrew Simpson: *An Empirical Study of Bug Bounty Programs*. Nel *2020 IEEE 2nd International Workshop on Intelligent Bug Fixing (IBF)*, pagine 35–44. Institute of Electrical and Electronics Engineers (IEEE), febbraio 2020. https://doi.org/10.1109/IBF50092.2020.9034828.

[19] Omer Akgul, Taha Eghtesad, Amit Elazari, Omprakash Gnawali, Jens Grossklags, Michelle L. Mazurek, Daniel Votipka e Aron Laszka: *Bug Hunters' Perspectives on the Challenges and Benefits of the Bug Bounty Ecosystem*. Nel *Proceedings of the 32nd USENIX Conference on Security Symposium (SEC '23)*, pagine 2275–2291. USENIX Association, agosto 2023. https://usenix.org/conference/usenixsecurity23/presentation/akgul.

[20] Omer Akgul, Taha Eghtesad, Amit Elazari, Omprakash Gnawali, Jens Grossklags, Daniel Votipka e Aron Laszka: *The Hackers' Viewpoint: Exploring Challenges and Benefits of Bug-Bounty Programs*. Nel *Proceedings of the 6th Workshop on Security Information Workers (WSIW '20)*. Leibniz University Hannover, Germania, novembre 2020. https://wsiw2020.sec.uni-hannover.de/downloads/WSIW2020-The%20Hackers%20Viewpoint.pdf.

[21] Suresh S. Malladi e Hemang C. Subramanian: *Bug Bounty Programs for Cybersecurity: Practices, Issues, and Recommendations*. IEEE Software (Institute of Electrical and Electronics Engineers), vol. 37 (no. 1): pp. 31–39, gennaio-febbario 2020. https://doi.org/10.1109/MS.2018.2880508.

[22] Huw Fryer e Elena Simperl: *Web Science Challenges in Researching Bug Bounties*. Nel *Proceedings of the 2017 ACM on Web Science Conference (WebSci '17)*, pagina 273–277. Association for Computing Machinery (ACM), giugno 2017. https://doi.org/10.1145/3091478.3091517.

[23] Alex Hoffman, Eric Becerril-Blas, Kevin Moreno e Yoohwan Kim: *Decentralized Security Bounty Management on Blockchain and IPFS*. Nel *2020 10th Annual Computing and Communication Workshop and Conference (CCWC)*, pagine 241–247. Institute of Electrical and Electronics Engineers (IEEE), gennaio 2020. https://doi.org/10.1109/CCWC47524.2020.9031109.

[24] Alex Hoffman, Phillipe Austria, Chol Hyun Park e Yoohwan Kim: *Bountychain: Toward Decentralizing a Bug Bounty Program with Blockchain and IPFS*. International Journal of Networked and Distributed Computing (Atlantis Press), vol. 9: pp. 86–93, luglio 2021. https://doi.org/10.2991/ijndc.k.210527.001.

[25] Lital Badash, Nachiket Tapas, Asaf Nadler, Francesco Longo e Asaf Shabtai: *Blockchain-Based Bug Bounty Framework*. Nel *Proceedings of the 36th Annual ACM Symposium on Applied Computing (SAC '21)*, pagine 239–248. Association for Computing Machinery (ACM), marzo 2021. https://doi.org/10.1145/3412841.3441906.

[26] Andrea Lisi, Prateeti Mukherjee, Laura De Santis, Lei Wu, Dmitrij Lagutin e Yki Kortesniemi: *Automated Responsible Disclosure of Security Vulnerabilities*. IEEE Access (Institute of Electrical and Electronics Engineers), vol. 10: pp. 10472–10489, settembre 2022. https://doi.org/10.1109/ACCESS.2021.3126401.

[27] N. Asokan, Matthias Schunter e Michael Waidner: *Optimistic Protocols for Multi-Party Fair Exchange*. Report di ricerca, IBM Research Division, novembre 1996. https://schunter.org/bibliography/AsSW2_96FairMPX.IBMrep.pdf.

[28] Matthew K. Franklin e Michael K. Reiter: *Fair Exchange with a Semi-Trusted Third Party*. Nel *Proceedings of the 4th ACM Conference on Computer and Communications Security (CCS '97)*, pagine 1–5. Association for Computing Machinery (ACM), aprile 1997. https://doi.org/10.1145/266420.266424.

[29] N. Asokan, Victor Shoup e Michael Waidner: *Optimistic Fair Exchange of Digital Signatures*. Nel *Advances in Cryptology (EUROCRYPT'98)*, volume 1403 della serie *Lecture Notes in Computer Science*, pagine 591–606. Springer, maggio 1998. https://doi.org/10.1007/BFb0054156.

[30] Matt Franklin e Gene Tsudik: *Secure Group Barter: Multi-party Fair Exchange with Semi-trusted Neutral Parties*. Nel *Financial Cryptography (FC '98)*, volume 1465 della serie *Lecture Notes in Computer Science*, pagine 90–102. Springer, maggio 1998. https://doi.org/10.1007/BFb0055475.

[31] Abdullah AlOtaibi e Hamza Aldabbas: *A Review of Fair Exchange Protocols*. International Journal of Computer Networks & Communications (AIRCC), vol. 4 (no. 4): pp. 20:1–13, luglio 2012. https://doi.org/10.5121/ijcnc.2012.4420.

[32] Jia Ch'ng Loh, Swee Huay Heng e Syh Yuan Tan: *Fair Exchange Protocol in Electronic Transactions Revisited*. Nel *2017 5th International Conference on Information and Communication Technology (ICoIC7)*, pagine 21:1–6. Institute of Electrical and Electronics Engineers (IEEE), maggio 2017. https://doi.org/10.1109/ICoICT.2017.8074660.

[33] Surakarn Duangphasuk, Pruegsa Duangphasuk e Chalee Thammarat: *Fair Exchange Protocol in Electronic Transactions Revisited*. Nel *2020 17th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON)*, pagine 331–334. Institute of Electrical and Electronics Engineers (IEEE), giugno 2020. https://doi.org/10.1109/ECTI-CON49241.2020.9158264.

[34] Stefan Dziembowski, Lisa Eckey e Sebastian Faust: *FairSwap: How To Fairly Exchange Digital Goods*. Nel *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security (CCS '18)*, pagine 967–984. Association for Computing Machinery (ACM), ottobre 2018. https://doi.org/10.1145/3243734.3243857.

[35] Sepideh Avizheh, Preston Haffey e Reihaneh Safavi-Naini: *Privacy-preserving FairSwap: Fairness and privacy interplay*. Proceedings on Privacy Enhancing Technologies (De Gruyter Open), vol. 2022 (no. 1): pp. 417–439, gennaio 2022. https://doi.org/10.2478/popets-2022-0021.

[36] Lisa Eckey, Sebastian Faust e Benjamin Schlosser: *OptiSwap: Fast Optimistic Fair Exchange*. Nel *Proceedings of the 15th ACM Asia Conference on Computer and Communications Security (ASIA CCS '20)*, pagine 543–557. Association for Computing Machinery (ACM), ottobre 2020. https://doi.org/10.1145/3320269.3384749.

[37] Sepideh Avizheh, Preston Haffey e Reihaneh Safavi-Naini: *Privacy-enhanced OptiSwap*. Nel *Proceedings of the 2021 on Cloud Computing Security Workshop (CCSW '21)*, pagine 39–57. Association for Computing Machinery (ACM), novembre 2021. https://doi.org/10.1145/3474123.3486756.

[38] Matthias Lohr, Benjamin Schlosser, Jan Jürjens e Steffen Staab: *Cost Fairness for Blockchain-Based Two-Party Exchange Protocols*. Nel *2020 IEEE International Conference on Blockchain (Blockchain)*, pagine 428–435. Institute of Electrical and Electronics Engineers (IEEE), novembre 2020. https://doi.org/10.1109/Blockchain50366.2020.00062.

[39] Matthias Lohr, Kenneth Skiba, Marco Konersmann, Jan Jürjens e Steffen Staab: *Formalizing Cost Fairness for Two-Party Exchange Protocols using Game Theory and Applications to Blockchain*. Nel *2022 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, pagine 25:1–5. Institute of Electrical and Electronics Engineers (IEEE), maggio 2022. https://doi.org/10.1109/ICBC54727.2022.9805522.

[40] Matthias Lohr, Kenneth Skiba, Marco Konersmann, Jan Jürjens e Steffen Staab: *Formalizing Cost Fairness for Two-Party Exchange Protocols using Game Theory and Applications to Blockchain (Extended Version)*. Computing Research Repository: Distributed, Parallel, and Cluster Computing (arXiv), marzo 2022. https://doi.org/10.48550/arXiv.2203.05925.

[41] Simon Janin, Kaihua Qin, Akaki Mamageishvili e Arthur Gervais: *FileBounty: Fair Data Exchange*. Nel *2020 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*, pagine 357–366. Institute of Electrical and Electronics Engineers (IEEE), settembre 2020. https://doi.org/10.1109/EuroSPW51379.2020.00056.

[42] Sergiu Bursuc e Sjouke Mauw: *Contingent payments from two-party signing and verification for abelian groups*. Nel *2022 IEEE 35th Computer*

Security Foundations Symposium (CSF), pagine 195–210. Institute of Electrical and Electronics Engineers (IEEE), agosto 2022. https://doi.org/10.1109/CSF54842.2022.9919654.

[43] Changhao Chenli, Wenyi Tang e Taeho Jung: *FairTrade: Efficient Atomic Exchange-based Fair Exchange Protocol for Digital Data Trading*. Nel *2021 IEEE International Conference on Blockchain (Blockchain)*, pagine 38–46. Institute of Electrical and Electronics Engineers (IEEE), dicembre 2021. https://doi.org/10.1109/Blockchain53845.2021.00017.

[44] Justin Thaler: *Proofs, Arguments, and Zero-Knowledge*. Foundations and Trends in Privacy and Security (Now), vol. 4 (no. 2-4): pp. 117–660, dicembre 2022. https://doi.org/10.1561/3300000030.

[45] Carmit Hazay e Yehuda Lindell: *Sigma Protocols and Efficient Zero-Knowledge*. Nel *Efficient Secure Two-Party Protocols: Techniques and Constructions*, capitolo 6, pagine 147–175. Springer, ottobre 2010. https://doi.org/10.1007/978-3-642-14303-8_6.

[46] Dan Boneh e Victor Shoup: *Identification and signatures from Sigma protocols*. Nel *A Graduate Course in Applied Cryptography*, capitolo 19, pagine 755–822. Pubblicato su https://toc.cryptobook.us/, versione 0.6, gennaio 2023. https://crypto.stanford.edu/~dabo/cryptobook/BonehShoup_0_6.pdf.

[47] Dan Boneh e Victor Shoup: *Proving properties in zero-knowledge*. Nel *A Graduate Course in Applied Cryptography*, capitolo 20, pagine 823–854. Pubblicato su https://toc.cryptobook.us/, versione 0.6, gennaio 2023. https://crypto.stanford.edu/~dabo/cryptobook/BonehShoup_0_6.pdf.

[48] Feng Li e Bruce McMillin: *A Survey on Zero-Knowledge Proofs*. Nel *Advances in Computers*, volume 94, capitolo 2, pagine 25–69. Elsevier, luglio 2014. https://doi.org/10.1016/B978-0-12-800161-5.00002-5.

[49] Shafi Goldwasser, Silvio Micali e Charles Rackoff: *The Knowledge Complexity of Interactive Proof-Systems*. Nel *Proceedings of the Seventeenth Annual ACM Symposium on Theory of Computing (STOC '85)*, pagine 291–304. Association for Computing Machinery (ACM), dicembre 1985. https://doi.org/10.1145/22145.22178.

[50] Jens Ernstberger, Stefanos Chaliasos, Liyi Zhou, Philipp Jovanovic e Arthur Gervais: *Do You Need a Zero Knowledge Proof?* Cryptology ePrint Archive, paper 2024/050, gennaio 2024. https://eprint.iacr.org/2024/050.

[51] Eduardo Morais, Tommy Koens, Cees van Wijk e Aleksei Koren: *A survey on zero knowledge range proofs and applications*. SN Applied Sciences (Springer), vol. 1 (no. 8): pp. 5:1–17, luglio 2019. https://doi.org/10.1007/s42452-019-0989-z.

[52] Yu Zhou, Zeming Wei, Shansi Ma e Hua Tang: *Overview of Zero-Knowledge Proof and Its Applications in Blockchain*. Nel *Blockchain Technology and Application - 5th CCF China Blockchain Conference*, volume 1736 della serie *Communications in Computer and Information Science*, pagine 60–82. Springer, dicembre 2022. https://doi.org/10.1007/978-981-19-8877-6_5.

[53] Juha Partala, Tri Hong Nguyen e Susanna Pirttikangas: *Non-Interactive Zero-Knowledge for Blockchain: A Survey*. IEEE Access (Institute of Electrical and Electronics Engineers), vol. 8: pp. 227945–227961, dicembre 2020. https://doi.org/10.1109/ACCESS.2020.3046025.

[54] Xiaoqiang Sun, F. Richard Yu, Peng Zhang, Zhiwei Sun, Weixin Xie e Xiang Peng: *A Survey on Zero-Knowledge Proof in Blockchain*. IEEE Network (Institute of Electrical and Electronics Engineers), vol. 35 (no. 4): pp. 198–205, agosto 2021. https://doi.org/10.1109/MNET.011.2000473.

[55] Duc A. Tran, My T. Thai e Bhaskar Krishnamachari: *Handbook on Blockchain*. Springer Optimization and Its Applications. Springer, prima edizione, novembre 2022, ISBN 9783031075353. https://doi.org/10.1007/978-3-031-07535-3.

[56] Ramchandra Sharad Mangrulkar e Pallavi Vijay Chavan: *Blockchain Essentials - Core Concepts and Implementations*. Apress, gennaio 2024, ISBN 9781484299753. https://doi.org/10.1007/978-1-4842-9975-3.

[57] Andreas M. Antonopoulos e Gavin Wood: *Mastering Ethereum: Building Smart Contracts and DApps*. O'Reilly, prima edizione, dicembre 2018, ISBN 9781491971949. https://github.com/ethereumbook/ethereumbook.

[58] Gavin Zheng, Longxiang Gao, Liqun Huang e Jian Guan: *Ethereum Smart Contract Development in Solidity*. Springer, prima edizione, agosto 2020, ISBN 9789811562181. https://doi.org/10.1007/978-981-15-6218-1.

[59] Weijia Zhang e Tej Anand: *Blockchain and Ethereum Smart Contract Solution Development - Dapp Programming with Solidity*. Apress, prima edizione, agosto 2022, ISBN 9781484281635. https://doi.org/10.1007/978-1-4842-8164-2.

[60] Ritesh Modi: *Solidity Programming Essentials: A guide to building smart contracts and tokens using the widely used Solidity language*. Packt, seconda edizione, giugno 2022, ISBN 9781803231181. https://packtpub.com/en-us/product/solidity-programming-essentials-9781803231181.

[61] Ethereum Foundation: *Ethereum Development Documentation*. Documentazione online. https://ethereum.org/developers/docs, consultato in data 15 gennaio 2024.

[62] The Solidity Authors: *Solidity Documentation - Release 0.8.18*, febbraio 2023. https://docs.soliditylang.org/_/downloads/en/v0.8.18/pdf/.

[63] Satoshi Nakamoto: *Bitcoin: A Peer-to-Peer Electronic Cash System*, 2008. https://bitcoin.org/bitcoin.pdf.

[64] Vitalik Buterin: *Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform*, dicembre 2014. https://ethereum.org/content/whitepaper/whitepaper-pdf/Ethereum_Whitepaper_-_Buterin_2014.pdf.

[65] Gavin Wood: *Ethereum: A Secure Decentralised Generalised Transaction Ledger (Paris Vesion 2f36cfo)*, gennaio 2024 (la prima versione è stata pubblicata nel 2014). https://ethereum.github.io/yellowpaper/paper.pdf, al link precedente è reperibile l'ultima versione disponibile.

[66] Cynthia Dwork e Moni Naor: *Pricing via Processing or Combatting Junk Mail*. Nel *Advances in Cryptology - CRYPTO '92 (12th Annual International Cryptology Conference)*, volume 740 della serie *Lecture Notes in Computer Science*, pagine 139–147. Springer, ottobre 1993. https://doi.org/10.1007/3-540-48071-4_10.

[67] Nick Szabo: *Smart Contracts*, 1994. https://nakamotoinstitute.org/smart-contracts/.

[68] Nick Szabo: *Formalizing and Securing Relationships on Public Networks*. First monday (First Monday Editorial Group), vol. 2 (no. 9), settembre 1997. https://doi.org/10.5210/fm.v2i9.548.

[69] Wei Dai: *b-money*, novembre 1997. https://nakamotoinstitute.org/b-money/.

[70] Vivek Vishnumurthy, Sangeeth Chandrakumar e Emin Gun Sirer: *KARMA : A Secure Economic Framework for Peer-to-Peer Resource Sharing*. Nel *Workshop on Economics of Peer-to-peer Systems*. University of California, Berkeley, School of Information, giugno 2003. https://groups.ischool.berkeley.edu/archive/p2pecon/papers/s5-vishnumurthy.pdf.

[71] Hal Finney: *RPOW - Reusable Proofs of Work*, agosto 2004. https://nakamotoinstitute.org/finney/rpow/index.html.

[72] Nick Szabo: *Bit Gold*, dicembre 2005. https://nakamotoinstitute.org/bit-gold/.

[73] Xiwei Xu, Ingo Weber e Mark Staples: *Architecture for Blockchain Applications*. Springer, prima edizione, marzo 2019, ISBN 9783030030353. https://doi.org/10.1007/978-3-030-03035-3.

[74] Xiwei Xu, Cesare Pautasso, Liming Zhu, Qinghua Lu e Ingo Weber: *A Pattern Collection for Blockchain-based Applications*. Nel *Proceedings of the 23rd European Conference on Pattern Languages of Programs (EuroPLoP '18)*, pagine 3:1–20. Association for Computing Machinery (ACM), luglio 2018. https://doi.org/10.1145/3282308.3282312.

[75] Yue Liu, Qinghua Lu, Xiwei Xu, Liming Zhu e Haonan Yao: *Applying Design Patterns in Smart Contracts*. Nel *Blockchain – ICBC 2018*, volume 10974 della serie *Lecture Notes in Computer Science*, pagine 92–106. Springer, giugno 2018. https://doi.org/10.1007/978-3-319-94478-4_7.

[76] Xiwei Xu, Ingo Weber, Mark Staples, Liming Zhu, Jan Bosch, Len Bass, Cesare Pautasso e Paul Rimba: *A Taxonomy of Blockchain-Based Systems for Architecture Design*. Nel *2017 IEEE International*

Conference on Software Architecture (ICSA), pagine 243–252. Institute of Electrical and Electronics Engineers (IEEE), aprile 2017. https://doi.org/10.1109/ICSA.2017.33.

[77] Maximilian Wöhrer: *Engineering Blockchain-Based Applications in the Context of the Ethereum Ecosystem*. Tesi di dottorato, Universität Wien, Faculty of Computer Science, Austria, settembre 2022. http://eprints.cs.univie.ac.at/7485/.

[78] Maximilian Wöhrer e Uwe Zdun: *Design Patterns for Smart Contracts in the Ethereum Ecosystem*. Nel *2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, pagine 1513–1520. Institute of Electrical and Electronics Engineers (IEEE), luglio 2018. https://doi.org/10.1109/Cybermatics_2018.2018.00255.

[79] Maximilian Wöhrer e Uwe Zdun: *Smart Contracts: Security Patterns in the Ethereum Ecosystem and Solidity*. Nel *2018 International Workshop on Blockchain Oriented Software Engineering (IWBOSE)*, pagine 2–8. Institute of Electrical and Electronics Engineers (IEEE), marzo 2018. https://doi.org/10.1109/IWBOSE.2018.8327565.

[80] Maximilian Wöhrer e Uwe Zdun: *Architectural Design Decisions for Blockchain-Based Applications*. Nel *2021 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, pagine 1–5. Institute of Electrical and Electronics Engineers (IEEE), maggio 2021. https://doi.org/10.1109/ICBC51069.2021.9461109.

[81] Maximilian Wöhrer, Uwe Zdun e Stefanie Rinderle-Ma: *Architecture Design of Blockchain-Based Applications*. Nel *2021 3rd Conference on Blockchain Research and Applications for Innovative Networks and Services (BRAINS)*, pagine 173–180. Institute of Electrical and Electronics Engineers (IEEE), settembre 2021. https://doi.org/10.1109/BRAINS52497.2021.9569813.

[82] Lodovica Marchesi, Michele Marchesi, Giuseppe Destefanis, Giulio Barabino e Danilo Tigano: *Design Patterns for Gas Optimization in Ethereum*. Nel *2020 IEEE International Workshop on Blockchain Oriented Software Engineering (IWBOSE)*, pagine 9–15. Institute of Electrical and Electronics Engineers (IEEE), febbraio 2020. https://doi.org/10.1109/IWBOSE50093.2020.9050163.

[83] Lodovica Marchesi, Michele Marchesi e Roberto Tonelli: *ABCDE–agile block chain DApp engineering*. Blockchain: Research and Applications (Elsevier), vol. 1 (no. 1): pp. 2:1–18, dicembre 2020. https://doi.org/10.1016/j.bcra.2020.100002.

[84] Michele Marchesi, Lodovica Marchesi e Roberto Tonelli: *An Agile Software Engineering Method to Design Blockchain Applications*. Nel *Proceedings of the 14th Central and Eastern European Software Engineering Conference Russia (CEE-SECR '18)*, pagine 3:1–8. Association for Computing Machinery (ACM), ottobre 2018. https://doi.org/10.1145/3290621.3290627.

[85] Karl Wüst e Arthur Gervais: *Do you Need a Blockchain?* Nel *2018 Crypto Valley Conference on Blockchain Technology (CVCBT)*, pagine 45–54. Institute of Electrical and Electronics Engineers (IEEE), giugno 2018. https://doi.org/10.1109/CVCBT.2018.00011.

[86] Anna Vacca, Andrea Di Sorbo, Corrado A. Visaggio e Gerardo Canfora: *A systematic literature review of blockchain and smart contract development: Techniques, tools, and open challenges*. Journal of Systems and Software (Elsevier), vol. 174 (articolo no. 110891): pp. 1–19, aprile 2021. https://doi.org/10.1016/j.jss.2020.110891.

[87] Bin Hu, Zongyang Zhang, Jianwei Liu, Yizhong Liu, Jiayuan Yin, Rongxing Lu e Xiaodong Lin: *A comprehensive survey on smart contract construction and execution: paradigms, tools, and systems*. Patterns (Cell Press), vol. 2 (no. 2): pp. 5:1–51, febbraio 2021. https://doi.org/10.1016/j.patter.2020.100179.

[88] Satpal Singh Kushwaha, Sandeep Joshi, Dilbag Singh, Manjit Kaur e Heung No Lee: *Ethereum Smart Contract Analysis Tools: A Systematic Review*. IEEE Access (Institute of Electrical and Electronics Engineers), vol. 10 (no.): pp. 57037–57062, aprile 2022. https://doi.org/10.1109/ACCESS.2022.3169902.

[89] Niclas Kannengießer, Sebastian Lins, Christian Sander, Klaus Winter, Hellmuth Frey e Ali Sunyaev: *Challenges and Common Solutions in Smart Contract Development*. IEEE Transactions on Software Engineering (Institute of Electrical and Electronics Engineers), vol. 48 (no. 11): pp. 4291–4318, novembre 2022. https://doi.org/10.1109/TSE.2021.3116808.

[90] Valerio Mandarino, Giuseppe Pappalardo e Emiliano Tramontana: *Some Blockchain Design Patterns for Overcoming Immutability, Chain-Boundedness, and Gas Fees*. Nel *2022 3rd Asia Conference on Computers and Communications (ACCC)*, pagine 65–71. Institute of Electrical and Electronics Engineers (IEEE), dicembre 2022. https://doi.org/10.1109/ACCC58361.2022.00018.

[91] Ting Chen, Zihao Li, Hao Zhou, Jiachi Chen, Xiapu Luo, Xiaoqi Li e Xiaosong Zhang: *Towards Saving Money in Using Smart Contracts*. Nel *Proceedings of the 40th International Conference on Software Engineering: New Ideas and Emerging Results (ICSE-NIER '18)*, pagine 81–84. Association for Computing Machinery (ACM), maggio 2018. https://doi.org/10.1145/3183399.3183420.

[92] Matteo Marescotti, Martin Blicha, Antti E. J. Hyvärinen, Sepideh Asadi e Natasha Sharygina: *Computing Exact Worst-Case Gas Consumption for Smart Contracts*. Nel *Leveraging Applications of Formal Methods, Verification and Validation: Industrial Practice (ISoLA 2018)*, volume 11247 della serie *Lecture Notes in Computer Science*, pagine 450–465. Springer, novembre 2018. https://doi.org/10.1007/978-3-030-03427-6_33.

[93] Andrea Di Sorbo, Sonia Laudanna, Anna Vacca, Corrado A. Visaggio e Gerardo Canfora: *Profiling gas consumption in solidity smart contracts*. Journal of Systems and Software (Elsevier), vol. 186 (articolo no. 111193): pp. 1–17, aprile 2022. https://doi.org/10.1016/j.jss.2021.111193.

[94] Nitima Masla, Vaibhav Vyas, Jyoti Gautam, Rabindra Nath Shaw e Ankush Ghosh: *Reduction in Gas Cost for Blockchain Enabled Smart Contract*. Nel *2021 IEEE 4th International Conference on Computing, Power and Communication Technologies (GUCON)*, pagine 951:1–6. Institute of Electrical and Electronics Engineers (IEEE), settembre 2021. https://doi.org/10.1109/GUCON50781.2021.9573701.

[95] Abdeljalil Beniiche: *A Study of Blockchain Oracles*. Computing Research Repository: Cryptography and Security (arXiv), luglio 2020. https://doi.org/10.48550/arXiv.2004.07140.

[96] Xiwei Xu, Ingo Weber e Mark Staples: *Blockchain Patterns*. Nel *Architecture for Blockchain Applications*, capitolo 7, pagine 113–148. Springer, prima edizione, marzo 2019. https://doi.org/10.1007/978-3-030-03035-3_7.

[97] Roman Mühlberger, Stefan Bachhofner, Eduardo Castelló Ferrer, Claudio Di Ciccio, Ingo Weber, Maximilian Wöhrer e Uwe Zdun: *Foundational Oracle Patterns: Connecting Blockchain to the Off-Chain World*. Nel *Business Process Management: Blockchain and Robotic Process Automation Forum (BPM '20)*, volume 393 della serie *Lecture Notes in Business Information Processing*, pagine 35–51. Springer, settembre 2020. https://doi.org/10.1007/978-3-030-58779-6_3.

[98] Xiaolong Liu, Riqing Chen, Yu Wen Chen e Shyan Ming Yuan: *Off-chain Data Fetching Architecture for Ethereum Smart Contract*. Nel *2018 International Conference on Cloud Computing, Big Data and Blockchain (ICCBB)*, pagine 15:1–4. Institute of Electrical and Electronics Engineers (IEEE), novembre 2018. https://doi.org/10.1109/ICCBB.2018.8756348.

[99] Hamda Al-Breiki, Muhammad Habib Ur Rehman, Khaled Salah e Davor Svetinovic: *Trustworthy Blockchain Oracles: Review, Comparison, and Open Research Challenges*. IEEE Access (Institute of Electrical and Electronics Engineers), vol. 8 : pp. 85675–85685, maggio 2020. https://doi.org/10.1109/ACCESS.2020.2992698.

[100] Ammar Hassan, Imran Makhdoom, Waseem Iqbal, Awais Ahmad e Asad Raza: *From trust to truth: Advancements in mitigating the Blockchain Oracle problem*. Journal of Network and Computer Applications (Elsevier), vol. 217 : pp. 14:1–17, agosto 2023. https://doi.org/10.1016/j.jnca.2023.103672.

[101] Amirmohammad Pasdar, Young Choon Lee e Zhongli Dong: *Connect API with Blockchain: A Survey on Blockchain Oracle Implementation*. ACM Computing Surveys (Association for Computing Machinery), vol. 55 (no. 10): pp. 208:1–39, febbraio 2023. https://doi.org/10.1145/3567582.

[102] Provable Things: *Provable Documentation*. Documentazione online. https://docs.provable.xyz, consultato in data 13 febbraio 2024.

[103] Chainlink Foundation: *Chainlink Docs*. Documentazione online. https://docs.chain.link, consultato in data 13 febbraio 2024.

[104] Lorenz Breidenbach, Christian Cachin, Benedict Chan, Alex Coventry, Steve Ellis, Ari Juels, Farinaz Koushanfar, Andrew Miller, Brendan Magauran, Daniel Moroz, Sergey Nazarov, Alexan-

dru Topliceanu, Florian Tramèr e Fan Zhang: *Chainlink 2.0: Next Steps in the Evolution of Decentralized Oracle Networks*, aprile 2021. https://research.chain.link/whitepaper-v2.pdf.

[105] Lorenz Breidenbach, Christian Cachin, Alex Coventry, Ari Juels e Andrew Miller: *Chainlink Off-chain Reporting Protocol*, febbraio 2021. https://research.chain.link/ocr.pdf.

[106] Fanshu Gong, Lanju Kong, Yuxuan Lu, Jin Qian e Xinping Min: *An Overview of Blockchain Scalability for Storage*. Nel *2023 26th International Conference on Computer Supported Cooperative Work in Design (CSCWD)*, pagine 516–521. Institute of Electrical and Electronics Engineers (IEEE), maggio 2023. https://doi.org/10.1109/CSCWD57460.2023.10152720.

[107] E. Sweetline Priya e R. Priya: *Performance Comparison of On-Chain and Off-Chain Data Storage Model Using Blockchain Technology*. Nel *Evolution in Computational Intelligence - Proceedings of the 11th International Conference on Frontiers of Intelligent Computing: Theory and Applications (FICTA 2023)*, volume 370 della serie *Smart Innovation, Systems and Technologies*, pagine 499–511. Springer, novembre 2023. https://doi.org/10.1007/978-981-99-6702-5_41.

[108] Aisyah Ismail, Mark Toohey, Young Choon Lee, Zhongli Dong e Albert Y. Zomaya: *Cost and Performance Analysis on Decentralized File Systems for Blockchain-Based Applications: State-of-the-Art Report*. Nel *2022 IEEE International Conference on Blockchain (Blockchain)*, pagine 230–237. Institute of Electrical and Electronics Engineers (IEEE), agosto 2022. https://doi.org/10.1109/Blockchain55522.2022.00039.

[109] Zulwaqar Zain Mohtar, Mohd Yazid Idris e Farhan Mohamed: *Blockchain-Based Distributed File System Security and Privacy: A Systematic Mapping Study*. Nel *2022 4th International Conference on Smart Sensors and Application (ICSSA)*, pagine 64–69. Institute of Electrical and Electronics Engineers (IEEE), luglio 2022. https://doi.org/10.1109/ICSSA54161.2022.9870967.

[110] Protocol Labs: *IPFS Documentation*. Documentazione online. https://docs.ipfs.tech/, consultato in data 9 febbrario 2024.

[111] Juan Benet: *IPFS - Content Addressed, Versioned, P2P File System*. Computing Research Repository: Networking and Internet Ar-

chitecture (arXiv), luglio 2014. https://doi.org/10.48550/arXiv.1407.3561.

[112] Dennis Trautwein, Aravindh Raman, Gareth Tyson, Ignacio Castro, Will Scott, Moritz Schubotz, Bela Gipp e Yiannis Psaras: *Design and Evaluation of IPFS: a Storage Layer for the Decentralized Web*. Nel *Proceedings of the ACM SIGCOMM 2022 Conference*, pagine 739–752. Association for Computing Machinery (ACM), agosto 2022. https://doi.org/10.1145/3544216.3544232.

[113] Trinh Viet Doan, Yiannis Psaras, Jörg Ott e Vaibhav Bajpai: *Toward Decentralized Cloud Storage With IPFS: Opportunities, Challenges, and Future Considerations*. IEEE Internet Computing (Institute of Electrical and Electronics Engineers), vol. 26 (no. 6): pp. 7–15, novembre 2022. https://doi.org/10.1109/MIC.2022.3209804.

[114] Manpreet Kaur, Shikha Gupta, Deepak Kumar, Maria Simona Raboaca, S. B. Goyal e Chaman Verma: *IPFS: An Off-Chain Storage Solution for Blockchain*. Nel *Proceedings of International Conference on Recent Innovations in Computing (ICRIC 2022)*, volume 1001 della serie *Lecture Notes in Electrical Engineering*, pagine 513–525. Springer, maggio 2023. https://doi.org/10.1007/978-981-19-9876-8_39.

[115] Edmund M. Clarke, Thomas A. Henzinger, Helmut Veith e Roderick Bloem: *Handbook of Model Checking*. Springer, prima edizione, maggio 2018, ISBN 9783319105741. https://doi.org/10.1007/978-3-319-10575-8.

[116] Armin Biere, Marijn Heule, Hans van Maaren e Toby Walsh: *Handbook of Satisfiability*, volume 336 della serie *Frontiers in Artificial Intelligence and Applications*. IOS Press, seconda edizione, aprile 2021, ISBN 9781643681603. https://doi.org/10.3233/FAIA336.

[117] Lech T. Polkowski: *Logic: Reference Book for Computer Scientists*, volume 245 della serie *Intelligent Systems Reference Library*. Springer, seconda edizione, ottobre 2023, ISBN 9783031420337. https://doi.org/10.1007/978-3-031-42034-4.

[118] Daniel Kroening: *Software Verification*. Nel *Handbook of Satisfiability*, volume 336 della serie *Frontiers in Artificial Intelligence and Applications*, capitolo 20, pagine 791–818. IOS Press, seconda edizione, aprile 2021. https://doi.org/10.3233/FAIA201004.

[119] Ranjit Jhala, Andreas Podelski e Andrey Rybalchenko: *Predicate Abstraction for Program Verification*. Nel *Handbook of Model Checking*, capitolo 15, pagine 447–491. Springer, prima edizione, maggio 2018. https://doi.org/10.1007/978-3-319-10575-8_15.

[120] Frances E. Allen: *Control Flow Analysis*. ACM SIGPLAN Notices (Association for Computing Machinery), vol. 5 (no. 7): pp. 1–19, luglio 1970. https://doi.org/10.1145/390013.808479.

[121] Joao Marques-Silva e Sharad Malik: *Propositional SAT Solving*. Nel *Handbook of Model Checking*, capitolo 9, pagine 247–275. Springer, prima edizione, aprile 2018. https://doi.org/10.1007/978-3-319-10575-8_9.

[122] Lech T. Polkowski: *Sentential Logic (SL)*. Nel *Logic: Reference Book for Computer Scientists*, volume 245 della serie *Intelligent Systems Reference Library*, capitolo 3, pagine 61–110. Springer, seconda edizione, ottobre 2023. https://doi.org/10.1007/978-3-031-42034-4_2.

[123] Alfred Horn: *On Sentences Which are True of Direct Unions of Algebras*. The Journal of Symbolic Logic (Cambridge University Press), vol. 16 (no. 1): pp. 14–21, marzo 1951. https://doi.org/10.2307/2268661.

[124] Edmund M. Clarke: *Model checking*. Nel *Foundations of Software Technology and Theoretical Computer Science (FSTTCS 1997)*, volume 1346 della serie *Lecture Notes in Computer Science*, pagine 54–56. Springer, ottobre 1997. https://doi.org/10.1007/BFb0058022.

[125] Edmund M. Clarke e Qinsi Wang: 2^5 *Years of Model Checking*. Nel *Perspectives of System Informatics*, volume 8974 della serie *Lecture Notes in Computer Science*, pagine 26–40. Springer, giugno 2014. https://doi.org/10.1007/978-3-662-46823-4_2.

[126] Clark W. Barrett, Roberto Sebastiani, Sanjit A. Seshia e Cesare Tinelli: *Satisfiability Modulo Theories*. Nel *Handbook of Satisfiability*, volume 336 della serie *Frontiers in Artificial Intelligence and Applications*, capitolo 33, pagine 1267–1329. IOS Press, seconda edizione, aprile 2021. https://doi.org/10.3233/FAIA201017.

[127] Clark Barrett e Cesare Tinelli: *Satisfiability Modulo Theories*. Nel *Handbook of Model Checking*, capitolo 11, pagine 305–343. Springer, prima edizione, aprile 2018. https://doi.org/10.1007/978-3-319-10575-8_11.

[128] David Monniaux: *A Survey of Satisfiability Modulo Theory*. Nel *Computer Algebra in Scientific Computing (CASC 2016)*, volume 9890 della serie *Lecture Notes in Computer Science*, pagine 401–425. Springer, settembre 2016. https://doi.org/10.1007/978-3-319-45641-6_26.

[129] Leonardo de Moura, Bruno Dutertre e Natarajan Shankar: *A Tutorial on Satisfiability Modulo Theories*. Nel *Computer Aided Verification (CAV 2007)*, volume 4590 della serie *Lecture Notes in Computer Science*, pagine 20–36. Springer, luglio 2007. https://doi.org/10.1007/978-3-540-73368-3_5.

[130] Armin Biere e Daniel Kröning: *SAT-Based Model Checking*. Nel *Handbook of Model Checking*, capitolo 10, pagine 277–303. Springer, prima edizione, aprile 2018. https://doi.org/10.1007/978-3-319-10575-8_10.

[131] Armin Biere: *Bounded Model Checking*. Nel *Handbook of Satisfiability*, volume 336 della serie *Frontiers in Artificial Intelligence and Applications*, capitolo 18, pagine 739–764. IOS Press, seconda edizione, aprile 2021. https://doi.org/10.3233/FAIA201002.

[132] Edmund Clarke, Armin Biere, Richard Raimi e Yunshan Zhu: *Bounded Model Checking Using Satisfiability Solving*. Formal methods in system design (Springer), vol. 19 (no. 1): pp. 7–34, luglio 2001. https://doi.org/10.1023/A:1011276507260.

[133] Lucas Cordeiro, Bernd Fischer e Joao Marques-Silva: *SMT-Based Bounded Model Checking for Embedded ANSI-C Software*. Nel *2009 IEEE/ACM International Conference on Automated Software Engineering (ASE)*, pagine 137–148. Institute of Electrical and Electronics Engineers (IEEE), novembre 2009. https://doi.org/10.1109/ASE.2009.63.

[134] Arie Gurfinkel: *Program Verification with Constrained Horn Clauses (Invited Paper)*. Nel *Computer Aided Verification (CAV 2022)*, volume 13371 della serie *Lecture Notes in Computer Science*, pagine 19–29. Springer, agosto 2022. https://doi.org/10.1007/978-3-031-13185-1_2.

[135] Nikolaj Bjørner, Arie Gurfinkel, Ken McMillan e Andrey Rybalchenko: *Horn Clause Solvers for Program Verification*. Nel *Fields*

of Logic and Computation II, volume 9300 della serie *Lecture Notes in Computer Science*, pagine 24–51. Springer, settembre 2015. https://doi.org/10.1007/978-3-319-23534-9_2.

[136] Emanuele De Angelis, Fabio Fioravanti, John P. Gallagher, Manuel V. Hermenegildo, Alberto Pettorossi e Maurizio Proietti: *Analysis and Transformation of Constrained Horn Clauses for Program Verification*. Theory and Practice of Logic Programming (Cambridge University Press), vol. 22 (no. 6): pp. 974–1042, novembre 2021. https://doi.org/10.1017/S1471068421000211.

[137] William F. Dowling e Jean H. Gallier: *Linear-time Algorithms for Testing the Satisfiability of Propositional Horn Formulae*. The Journal of Logic Programming (Elsevier), vol. 1 (no. 3): pp. 267–284, ottobre 1984. https://doi.org/10.1016/0743-1066(84)90014-1.

[138] Giorgio Gallo e Giampaolo Urbani: *Algorithms for Testing the Satisfiability of Propositional Formulae*. The Journal of Logic Programming (Elsevier), vol. 7 (no. 1): pp. 45–61, luglio 1989. https://doi.org/10.1016/0743-1066(89)90009-5.

[139] Ethereum Foundation: *Formal Verification of Smart Contracts*. Parte della *Ethereum Development Documentation*. https://ethereum.org/en/developers/docs/smart-contracts/formal-verification, consultato in data 15 gennaio 2024.

[140] Palina Tolmach, Yi Li, Shang Wei Lin, Yang Liu e Zengxiang Li: *A Survey of Smart Contract Formal Specification and Verification*. ACM Computing Surveys (Association for Computing Machinery), vol. 54 (no. 7): pp. 148:1–38, luglio 2021. https://doi.org/10.1145/3464421.

[141] Moez Krichen, Mariam Lahami e Qasem Abu Al–Haija: *Formal Methods for the Verification of Smart Contracts: A Review*. Nel *2022 15th International Conference on Security of Information and Networks (SIN)*, pagine 37–44. Institute of Electrical and Electronics Engineers (IEEE), novembre 2022. https://doi.org/10.1109/SIN56466.2022.9970534.

[142] Alexandre Mota, Fei Yang e Cristiano Teixeira: *Formally Verifying a Real World Smart Contract*. Computing Research Repository: Software Engineering (arXiv), luglio 2023. https://doi.org/10.48550/arXiv.2307.02325.

[143] The Solidity Authors: *SMTChecker and Formal Verification*. Nel *Solidity Documentation - Release 0.8.18*, sezione 3.30, pagine 282–296. febbraio 2023. https://docs.soliditylang.org/_/downloads/en/v0.8.18/pdf/.

[144] Rodrigo Otoni, Matteo Marescotti, Leonardo Alt, Patrick Eugster, Antti Hyvärinen e Natasha Sharygina: *A Solicitous Approach to Smart Contract Verification*. ACM Transactions on Privacy and Security (Association for Computing Machinery), vol. 26 (no. 2): pp. 15:1–28, marzo 2023. https://doi.org/10.1145/3564699.

[145] Matteo Marescotti, Rodrigo Otoni, Leonardo Alt, Patrick Eugster, Antti E. J. Hyvärinen e Natasha Sharygina: *Accurate Smart Contract Verification Through Direct Modelling*. Nel *Leveraging Applications of Formal Methods, Verification and Validation: Applications (ISoLA 2020)*, volume 12478 della serie *Lecture Notes in Computer Science*, pagine 178–194. Springer, ottobre 2020. https://doi.org/10.1007/978-3-030-61467-6_12.

[146] Leonardo Alt e Christian Reitwiessner: *SMT-Based Verification of Solidity Smart Contracts*. Nel *Leveraging Applications of Formal Methods, Verification and Validation: Industrial Practice (ISoLA 2018)*, volume 11247 della serie *Lecture Notes in Computer Science*, pagine 376–388. Springer, ottobre 2018. https://doi.org/10.1007/978-3-030-03427-6_28.

[147] Leonardo Alt, Martin Blicha, Antti E. J. Hyvärinen e Natasha Sharygina: *SolCMC: Solidity Compiler's Model Checker*. Nel *Computer Aided Verification (CAV 2022)*, volume 13371 della serie *Lecture Notes in Computer Science*, pagine 325–338. Springer, agosto 2022. https://doi.org/10.1007/978-3-031-13185-1_16.

[148] Renée McCauley, Sue Fitzgerald, Gary Lewandowski, Laurie Murphy, Beth Simon, Lynda Thomas e Carol Zander: *Debugging: a review of the literature from an educational perspective*. Computer Science Education (Routledge), vol. 18 (no. 2): pp. 67–92, giugno 2008. https://doi.org/10.1080/08993400802114581.

[149] Debolina Ghosh e Jagannath Singh: *A Systematic Review on Program Debugging Techniques*. Nel *Smart Computing Paradigms: New Progresses and Challenges - Proceedings of ICACNI 2018*, volume 767

della serie *Advances in Intelligent Systems and Computing*, pagine 193–199. Springer, dicembre 2019. https://doi.org/10.1007/978-981-13-9680-9_16.

[150] Nick Papoulias, Noury Bouraqadi, Luc Fabresse, Stéphane Ducasse e Marcus Denker: *Mercury: Properties and Design of a Remote Debugging Solution using Reflection*. Journal of Object Technology (AITO), vol. 14 (no. 2): pp. 1:1–36, maggio 2015. https://doi.org/10.5381/jot.2015.14.2.a1.

[151] Muhammet Oguz Ozcan, Fatih Odaci e Ismail Ari: *Remote Debugging for Containerized Applications in Edge Computing Environments*. Nel *2019 IEEE International Conference on Edge Computing (EDGE)*, pagine 30–32. Institute of Electrical and Electronics Engineers (IEEE), luglio 2019. https://doi.org/10.1109/EDGE.2019.00021.

[152] Jakob Engblom: *A review of reverse debugging*. Nel *Proceedings of the 2012 System, Software, SoC and Silicon Debug Conference*, pagine 5:1–6. Institute of Electrical and Electronics Engineers (IEEE), settembre 2012. https://ieeexplore.ieee.org/abstract/document/6338149.

[153] Anthony Savidis e Vangelis Tsiatsianas: *Implementation of Live Reverse Debugging in LLDB*. Computing Research Repository: Software Engineering (arXiv), agosto 2021. https://doi.org/10.48550/arXiv.2105.12819.

[154] Edsger W. Dijkstra: *Guarded Commands, Nondeterminacy and Formal Derivation of Programs*. Communications of the ACM (Association for Computing Machinery), vol. 18 (no. 8): pp. 453–457, agosto 1975. https://doi.org/10.1145/360933.360975.

[155] Edsger W. Dijkstra: *A Discipline of Programming*. Series in Automatic Computation. Prentice Hall, prima edizione, 1976, ISBN 9780132158718. https://worldcat.org/oclc/01958445.

[156] David Gries: *The Science of Programming*. Monographs in Computer Science. Springer, prima edizione, febbraio 1987, ISBN 9780387964805. https://doi.org/10.1007/978-1-4612-5983-1.

[157] Edsger W. Dijkstra e Carel S. Scholten: *Predicate Calculus and Program Semantics*. Monographs in Computer Science. Springer,

prima edizione, 1990, ISBN 9781461232285. https://doi.org/10.1007/978-1-4612-3228-5.

[158] Marcello M. Bonsangue e Joost N. Kok: *The Weakest Precondition Calculus: Recursion and Duality*. Formal Aspects of Computing (Springer), vol. 6 (no. 1): pp. 788–800, novembre 1994. https://doi.org/10.1007/BF01213603.

[159] Roberto Baldoni, Emilio Coppa, Daniele Cono D'elia, Camil Demetrescu e Irene Finocchi: *A Survey of Symbolic Execution Techniques*. ACM Computing Surveys (Association for Computing Machinery), vol. 51 (no. 3): pp. 50:1–39, maggio 2019. https://doi.org/10.1145/3182657.

[160] Guowei Yang, Antonio Filieri, Mateus Borges, Donato Clun e Junye Wen: *Advances in Symbolic Execution*. Volume 113 della serie *Advances in Computers*, capitolo 5, pagine 225–287. Elsevier, prima edizione, gennaio 2019. https://doi.org/10.1016/bs.adcom.2018.10.002.

[161] Corina S. Păsăreanu, Rody Kersten, Kasper Luckow e Quoc Sang Phan: *Symbolic Execution and Recent Applications to Worst-Case Execution, Load Testing, and Security Analysis*. Volume 113 della serie *Advances in Computers*, capitolo 6, pagine 289–314. Elsevier, prima edizione, gennaio 2019. https://doi.org/10.1016/bs.adcom.2018.10.004.

[162] Hui Xu, Zirui Zhao, Yangfan Zhou e Michael R. Lyu: *Benchmarking the Capability of Symbolic Execution Tools with Logic Bombs*. IEEE Transactions on Dependable and Secure Computing (Institute of Electrical and Electronics Engineers), vol. 17 (no. 6): pp. 1243–1256, novembre-dicembre 2020. https://doi.org/10.1109/TDSC.2018.2866469.

[163] Hui Xu, Yangfan Zhou, Yu Kang e Michael R. Lyu: *Concolic Execution on Small-Size Binaries: Challenges and Empirical Study*. Nel *2017 47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pagine 181–188. Institute of Electrical and Electronics Engineers (IEEE), giugno 2017. https://doi.org/10.1109/DSN.2017.11.

[164] Sebastian Poeplau e Aurélien Francillon: *Systematic Comparison of Symbolic Execution Systems: Intermediate Representation and its*

Generation. Nel *Proceedings of the 35th Annual Computer Security Applications Conference (ACSAC '19)*, pagine 163–176. Association for Computing Machinery (ACM), dicembre 2019. https://doi.org/10.1145/3359789.3359796.

[165] Frank S. de Boer e Marcello Bonsangue: *Symbolic execution formally explained*. Formal Aspects of Computing (Springer), vol. 33 (no. 4): pp. 617–636, agosto 2021. https://doi.org/10.1007/s00165-020-00527-y.

[166] The angr Project contributors: *angr: The angr Project (v9.2.90)*, febbraio 2024. https://docs.angr.io/_/downloads/en/v9.2.90/pdf/.

[167] Yan Shoshitaishvili, Ruoyu Wang, Christopher Salls, Nick Stephens, Mario Polino, Andrew Dutcher, John Grosen, Siji Feng, Christophe Hauser, Christopher Kruegel e Giovanni Vigna: *SOK: (State of) The Art of War: Offensive Techniques in Binary Analysis*. Nel *2016 IEEE Symposium on Security and Privacy (SP)*, pagine 138–157. Institute of Electrical and Electronics Engineers (IEEE), maggio 2016. https://doi.org/10.1109/SP.2016.17.

[168] Nick Stephens, Jessie Grosen, Christopher Salls, Audrey Dutcher, Ruoyu Wang, Jacopo Corbetta, Yan Shoshitaishvili, Christopher Kruegel e Giovanni Vigna: *Driller: Augmenting Fuzzing Through Selective Symbolic Execution*. Nel *Network and Distributed System Security (NDSS) Symposium 2016*. Internet Society (ISOC), febbraio 2016. https://doi.org/10.14722/NDSS.2016.23368.

[169] Yan Shoshitaishvili, Ruoyu Wang, Christophe Hauser, Christopher Kruegel e Giovanni Vigna: *Firmalice - Automatic Detection of Authentication Bypass Vulnerabilities in Binary Firmware*. Nel *Network and Distributed System Security (NDSS) Symposium 2015*. Internet Society (ISOC), febbraio 2015. https://doi.org/10.14722/NDSS.2015.23294.

[170] Jacob Springer e Wu chang Feng: *Teaching with angr: A Symbolic Execution Curriculum and CTF*. Nel *2018 USENIX Workshop on Advances in Security Education (ASE 18)*. USENIX Association, agosto 2018. https://usenix.org/conference/ase18/presentation/springer.

[171] Fabrizio Biondi, Thomas Given-Wilson, Axel Legay, Cassius Puodzius e Jean Quilbeuf: *Tutorial: An Overview of Malware Detection and Evasion Techniques*. Nel *Leveraging Applications of Formal Methods, Verification and Validation: Modeling (ISoLA 2018)*, volume 11244 della serie *Lecture Notes in Computer Science*, pagine 565–586. Springer, ottobre 2018. https://doi.org/10.1007/978-3-030-03418-4_34.

[172] Satish Chandra, Stephen J. Fink e Manu Sridharan: *Snugglebug: A Powerful Approach To Weakest Preconditions*. Nel *Proceedings of the 30th ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI '09)*, pagine 363–374. Association for Computing Machinery (ACM), giugno 2009. https://doi.org/10.1145/1542476.1542517.

[173] Robert Husák, Jan Kofroň e Filip Zavoral: *Handling Heap Data Structures in Backward Symbolic Execution*. Nel *Formal Methods - FM 2019 International Workshops*, volume 12233 della serie *Lecture Notes in Computer Science*, pagine 537–556. Springer, agosto 2020. https://doi.org/10.1007/978-3-030-54997-8_33.

[174] Zachary Palmer, Theodore Park, Scott Smith e Shiwei Weng: *Higher-Order Demand-Driven Symbolic Evaluation*. Proceedings of the ACM on Programming Languages (Association for Computing Machinery), vol. 4 (no. ICFP): pp. 102:1–28, agosto 2020. https://doi.org/10.1145/3408984.

[175] Marek Chalupa e Jan Strejček: *Backward Symbolic Execution with Loop Folding*. Nel *Static Analysis - 28th International Symposium (SAS 2021)*, volume 12913 della serie *Lecture Notes in Computer Science*, pagine 49–76. Springer, ottobre 2021. https://doi.org/10.1007/978-3-030-88806-0_3.

[176] Alexander V. Misonizhnik, Yury O. Kostyukov, Mikhail P. Kostitsyn, Dmitry A. Mordvinov e Dmitry V. Koznov: *Generation of the weakest preconditions of programs with dynamic memory in symbolic execution*. Scientific and Technical Journal of Information Technologies, Mechanics and Optics (ITMO Univeristy), vol. 22 (no. 5): pp. 982–991, settembre-ottobre 2022. https://doi.org/10.17586/2226-1494-2022-22-5-982-991, tradotto dal russo con DeepL.

RINGRAZIAMENTI

<Ringraziamento 1>.

<Ringraziamento 2>.

<Ringraziamento 3>.

<Ringraziamento 4>.

<Ringraziamento 5>.