Capitolo 2

Analisi statica

L'analisi statica è un processo di valutazione di programmi che rientra tra le tecniche di verifica del sotware. L'aggettivo *statica* identifica una serie di controlli che possono essere effettuati sul codice prima della sua esecuzione. In questo si differenzia dall'analisi dinamica, una tecnica complementare che comprende quei controlli che vengono invece effettuati a runtime.

L'analisi statica solitamente è il primo controllo che viene effettuato sul codice.

2.1 Analisi statica vs. analisi dinamica

Per analisi statica si intende l'analisi dei programmi dal punto di vista del codice che li compone, vale a dire senza doverli eseguire. L'analisi può essere fatta sia sul codice sorgente, che sul codice oggetto, ossia il prodotto della compilazione.

L'analisi statica viene condotta su tre dimensioni: esaminando la struttura del programma, costruendo un modello che rappresenti i possibili stati del codice e ragionando sul possibile comportamento in fase di esecuzione [4]. Rientrano in questa categoria la verifica formale dei programmi e le ottimizzazioni a tempo di compilazione. L'analisi statica viene spesso implementata da tool automatici, e garantisce proprietà di sound. Le principali critiche mosse nei confronti di questa tecnica derivano dal fatto che possa portare a dei *falsi positivi*, cioè situazioni in cui viene segnalata una vulnerabilità nel codice sebbene non sia stata violata alcuna regola.

Dall'altro lato l'analisi dinamica identifica quei controlli che possono essere effettuati sul programma soltanto durante la sua esecuzione, che sia su un processore reale o virtuale. Il software testing rientra in questa categoria.

Per condurre questo tipo di controllo è necessario fornire un input ben preciso e analizzare poi il comportamento del programma. Occorre inoltre stabilire a priori *che cosa* si vuole misurare.

Sebbene questa analisi sia più veloce rispetto alla prima, non garantisce la stessa *soundness*. Per essere rigorosa infatti l'analisi dinamica dovrebbe coprire ogni possibile configurazione del programma.

Le tecniche di analisi possono essere suddivise in due tipologie in base al loro obiettivo. La prima categoria comprende i tool di analisi volti a localizzare bug nel codice. La seconda identifica invece un gruppo di software con una forte base logica, che utilizzano tecniche matematiche per la verifica di specifiche prorietà del programma.

QUESTA SEZIONE CERCHEREI DI ESPANDERLA UN PO' DICENDO CHE ALCUNE TECNICHE SONO "MANUALI" E ALTRE "AUTOMATIZZABILI".

2.2 Tecniche di analisi statica in Informatica

Di seguito daremo una panoramica sulle principali tecniche [13] di analisi statica.

2.2.1 La compilazione

MEGLIO SE SPIEGHI CON UN PO' PIÙ DI DETTAGLIO CHE LA COMPILAZIONE VISTA COME TRADUZIONE RICHIEDE UN MINIMO DI ANALISI PER ESSERE ESEGUITA.

Questa tecnica identifica i controlli effettuati dal compilatore, che variano a seconda del linguaggio di programmazione del codice. Le anomalie che possono essere catturate da questo tipo di analisi possono variare, ma in generale comprendono l'incoerenza dei tipi, la mancata dichiarazione delle variabili e il codice non raggiungibile dal flusso di controllo.

2.2.2 Code reading

Come suggerisce il termine stesso si tratta della rilettura del codice da parte di una persona. Sebbene i bug identificabili possono variare in base a diversi fattori (es. numero di persone, conoscenza del codice, livello di esperienza) questa operazione può portare alla luce difetti che invece il compilatore non rileva. Commenti inconsistenti con il codice, nomi di variabili errati, loop infiniti, codice non strutturato, sono solo alcuni di questi. L'efficacia di questa tecnica è limitata se colui che legge il codice è la stessa persona ad averlo sviluppato.

2.2.3 Code reviews

Generalmente adottata in contesti aziendali. Identifica un controllo del codice fatto in gruppo, il quale viene costituito secondo requisiti specifici. È una riunione dove lo sviluppatore è chiamato a leggere il codice ad alta voce di fronte ad altri esperti, che possono commentare il programma con lo scopo di individuare gli errori; in questo modo possono essere rilevati dal 30 al 70% di quelli presenti nel programma.

2.2.4 Walktrough

Molto simile alla tecnica di precedente nelle modalità in cui viene effettuata, poiché prevede la riunione di un gruppo di persone. Differisce negli obiettivi: cerca di trovare dei difetti nel comportamento del programma, e per farlo simula l'esecuzione del codice a mano.

2.2.5 Control Flow Analysis

Prevede la rappresentazione del codice attraverso un grafo chiamato CFG (*Control Flow Graph*, dove ciascun nodo rappresenta un'istruzione o un predicato, mentre gli archi il passaggio del flusso di controllo. Successivamente il grafo viene analizzato, al fine

degli smart contract è il rischio di ottenere falsi positivi.

2.4 Tool per l'analisi

Durante questo lavoro è stato preso in considerazione un certo numero di software che implementano tecniche di analisi statica orientata alla verifica degli smart contract. Di seguito ne vedremo alcuni.

2.4.1 Verificare le proprietà di sicurezza

I seguenti software sono stati pensati per verificare la sicurezza dei programmi di Ethereum.

Il primo è uno strumento completo, per cui si può etichettare uno smart contract come *sicuro* o meno. Il secondo invece è in grado di indivuare dei comportamenti anomali dei programmi causati soltanto dall'esaurimento del gas. Dunque la sua verifica comprende una tipologia circoscritta di proprietà di sicurezza.

EtherTrust [8] questo framework offre la possibilità di analizzare i programmi al fine di verificarne le proprietà di sicurezza. Tali proprietaà, come ad es. la *single-entrancy*, per poter essere verificate devono prima essere modellate.
Per condurre la sua analisi EtherTrust produce una rappresentazione astratta del bytecode EVM nella forma di clausole di Horn. Successivamente questa rappresentazione viene data in input ad un SMT solver, il quale verifica che siano rispettate delle proprietà di sicurezza ben precise. EtherTrust è garantito essere *sound*.

MadMax [7] attraverso la combinazione di più tecniche di analisi statica (analisi Data Flow e Control Flow) questo software è in grado di verificare smart contract al fine di scoprire bug legati all'esaurimento del gas disponibile.
MadMax individua una serie di vulnerabilità *gas-focused* in modo da definire dei pattern da ricercare attraverso l'analisi dei programmi. Questa viene condotta a