

Jollar

Una criptomoneda en Jolie

- Jollar – Laboratorio Sistemi Operativi A.A. 2017-2018
- Gruppo13
- matteo.tancredi@studio.unibo.it – giandomenico.derrico@studio.unibo.it – nicola.cresci@studio.unibo.it
- Componenti:
 - Tancredi, Matteo, Matr. 0000731746
 - D’Errico, Giandomenico, Matr. 0000766245
 - Cresci, Nicola, Matr. 0000768026

Indice

0. *Introduzione*
1. *Composizione progetto*
 - 1.1 *Interfaccia*
 - 1.1.1 *OneWay*
 - 1.1.2 *RequestResponse*
 - 1.2 *ServerTimeStamp*
 - 1.3 *Network Visualizer*
 - 1.4 *Nodi*
 - 1.4.1 *Nodo2, Nodo3, Nodo4*
2. *Implementazione*
3. *Proof-of-Work*
4. *Demo*

0. Introduzione

Il progetto che abbiamo svolto, consiste nella creazione di un sistema Peer-to-peer, atto allo scambio di criptovalute. La criptovaluta in questione si chiama “**Jollar**”. Il codice è interamente scritto in **Jolie**.

1. Composizione Progetto

I file che compongono questo progetto, sono **7**:

- **Interfaccia.ol**
- **ServerTimeStamp.ol**
- **NetworkVisualizer.ol**
- **Nodo1.ol**
- **Nodo2.ol**
- **Nodo3.ol**
- **Nodo4.ol**

Ogni file ha la sua funzione specifica, e i nodi, benchè simili, sono diversi tra loro.

1.1 Interfaccia

All'interno di questo file, sono definiti tutti tipi principali del sistema Jollar.

- **Nodo** → possiede un campo **id**, di tipo **int**, un campo **public key** ed uno **private key**, entrambi di tipo **string**, e un campo **jollar** di tipo **int**.
- **Transaction** → possiede un campo **hashTransazione** di tipo **string**, un campo **node seller** di tipo **int**, un campo **node buyer** di tipo **string** ed un campo **jollar** di tipo **int**.
- **Block** → possiede un campo **id blocco** di tipo **int**, un campo **previous block hash** di tipo **string**, un campo **block hash** di tipo **string**, un campo **difficutly** di tipo **double**, un campo **lunghezza proof of work** di tipo **int**, un campo **transaction** di tipo **Transaction**, un campo **timestamp** di tipo **undefined**, ed un campo **firma** di tipo **string**.
- **Blockchain** → contiene un campo **block*** di tipo **Block**

Dopo la definizione di questi tipi, è definita l'interfaccia, chiamata **InterfacciaJollar**, ed all'interno di essa sono definite le **operation** utili al progetto.

1.1.1. OneWay

- **collegamentoNodo1(Nodo)** → Invia i dati del nodo1 al NetworkVisualizer.
- **collegamentoNodo2(Nodo)** → Invia i dati del nodo2 al NetworkVisualizer.
- **collegamentoNodo3(Nodo)** → Invia i dati del nodo3 al NetworkVisualizer.
- **collegamentoNodo4(Nodo)** → Invia i dati del nodo4 al NetworkVisualizer.
- **nodiCollegati(undefined)** → Questa OneWay viene utilizzata dal nodo4, comunica ai primi tre nodi che si è collegato, e di conseguenza tutti insieme possono scaricare i dati della blockchain contenente il blocco genesis.
- **invioHashGenesis(string)** → Serve ad inviare a tutti i nodi l'Hash del blocco Genesis creato dal nodo1. Il nodo1 invia l'hash ai nodi restanti in contemporanea.

- **invioBlockchain(Blockchain)** → Il primo nodo che convalida la transazione, invia la blockchain al ServerTimeStamp.
- **blockchainScaricata(string)** → Ogni nodo invia un “ok” al NetworkVisualizer dopo che ha scaricato la blockchain
- **proofOfWork(string)** → Utilizzata dal nodo1 per dare il via agli altri nodi, cosicchè possano iniziare insieme la proof of work.
- **invioBlockchainAggiornata(Blockchain)** → invia a tutti i nodi la blockchain aggiornata.
- **netConnesso(string)** → Utilizzata dal NetworkVisualizer per comunicare al ServerTimeStamp che si è connesso.
- **jollarTotali(undefined)** → Utilizzato per contare il numero totale di Jollar in circolazione nel sistema.

1.1.2. RequestResponse

nuovaTransazione(Transaction)(undefined) → invia al nodo interessato i dati della transazione

creazioneBloccoPrimo(Transaction)(undefined) → crea il primo blocco della blockchain

creazioneBloccoSecondo(Transaction)(undefined) → crea il secondo blocco della blockchain

creazioneBloccoTerzo(Transaction)(undefined) → crea il terzo blocco della blockchain

proofOfWorkTerminata(int)(string) → Serve ai nodi per comunicare al Network Visualizer, la posizione con cui ha terminato la proof of work

1.2. ServerTimeStamp

Un server timestamp offre un servizio che permette di conoscere l’ora esatta all’interno della rete. Un nodo che intende effettuare le operazioni di scrittura o di validazione di un blocco deve richiedere il timestamp al server. Esso garantisce l’esistenza dei dati al momento della richiesta.

1.3. NetworkVisualizer

All’interno del Network Visualizer vengono stampate tutte le informazioni relative allo stato dei nodi e dei blocchi e alla fine dell’esecuzione dell’applicativo, viene anche stampata la blockchain più lunga.

1.4. Nodo1

E’ il nodo principale. Sceglie se effettuare le transazioni, a chi inviare i Jollar e la quantità.

1.4.1. Nodo2, Nodo3, Nodo4

Sono simili tra loro, attendono uno per volta le transazioni.

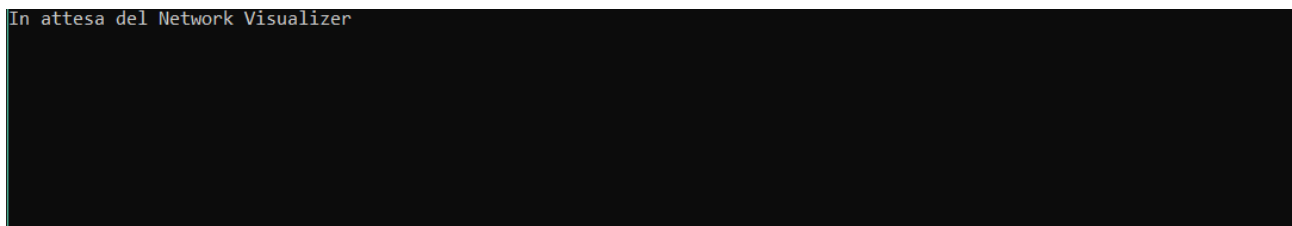
2. Implementazione

L'intero progetto, si basa sulla demo. Viene avviato il ServerTimeStamp, che resta in attesa del NetworkVisualizer, il quale dopo l'avvio attende la connessione dei quattro nodi. Al collegamento, il nodo1 crea il blocco genesi e guadagna 6 Jollar. Quando il quarto ed ultimo nodo si collega, lo fa sapere anche agli altri nodi. Secondo, terzo e quarto nodo restano in attesa, mentre il nodo uno può scegliere se effettuare o meno una transazione. Se sceglie "S", dovrà scegliere il nodo e la quantità di Jollar da inviargli. Viene dunque effettuata la transazione, la quale viene convalidata dal primo nodo che riesce a terminare la Proof-of-Work. La Proof-of-Work all'interno di questo progetto, viene effettuata mediante 4 metodi. Il primo, proofOfWork, sceglie un numero compreso tra 1 e 3. In base al numero che viene generato in maniera casuale, viene chiamato un altro metodo. Gli altri tre metodi sono: cunninghamPrimo, cunninghamSecondo, bitwin. Ognuno di questi metodi, sceglie un numero casuale compreso tra 1 e 5. In base al numero generato, viene scelta una catena. Il primo a validare la transazione, firma il blocco. La prima transazione avviene dal nodo1 al nodo2, la seconda transazione dal nodo1 al nodo3, la terza transazione dal nodo1 al nodo4. Quando tutte le transazioni vengono effettuate, nel Network Visualizer viene stampata l'intera blockchain.

4. Demo


Per avviare una demo del progetto bisogna eseguire i file in maniera sequenziale:

1. ServerTimeStamp.ol (attende la connessione del NetworkVisualizer)



```
In attesa del Network Visualizer
```

2. NetworkVisualizer.ol (attende la connessione dei nodi, in maniera ordinata)



```
J O L L A R J$  
-----  
BENVENUTO, SONO IN ATTESA DEL  
COLLEGAMENTO DEL PRIMO NODO  
-----  
.....Attendo la connessione.....  
-----
```

3. Nodo1.ol
4. Nodo2.ol
5. Nodo3.ol
6. Nodo4.ol

Quando tutti i nodi si saranno collegati, avremo queste schermate all'interno dei terminali

```

Hai accettato la connessione.
Sei il terzo nodo ad essersi collegato
TUTTI I NODI SI SONO COLLEGATI
-----|
| INFORMAZIONI NODO 3 |
|-----|
ID: 3
Chiave pubblica: 3dd5e08afe643ae88d8d86431a6be5e
Dollar Posseduti: 0
Ho scaricato la BLOCKCHAIN
In attesa di operazioni.....

Hai accettato la connessione.
Sei il quarto ed ultimo nodo ad essersi collegato
TUTTI I NODI SI SONO COLLEGATI
-----|
| INFORMAZIONI NODO 4 |
|-----|
ID: 4
Chiave pubblica: bca6e50d2df6bf59049d475b886cf569
Dollar Posseduti: 0
Ho scaricato la BLOCKCHAIN
In attesa di operazioni.....

```

Dopo quando tutte e tre le transazioni vengono effettuate, avremo questa schermata all'interno del Network Visualizer.

```

BLOCKCHAIN
-----
BLOCCO GENESI
-----
ID: 0
HASH: 43536a64710669190239ec2a2e8b08f6
DIFFICULTY: 2
-----
BLOCCO UNO
-----
ID: 1
HASH: d7608d380fd53704ec1c7ed727da6485
HASH PRECEDENTE: 43536a64710669190239ec2a2e8b08f6
LUNGHEZZA POW: 4
DIFFICULTY: 5
FIRMA: c188elfeead9490dd2cf6a59ce44bb03
ID NODO SELLER: 1
NODO BUYER: 2
JOLLAR INVIATI: 1
-----
BLOCCO DUE
-----
ID: 2
HASH: 0604fd456493ceaae0e2e545863b924
HASH PRECEDENTE: d7608d380fd53704ec1c7ed727da6485
LUNGHEZZA POW: 4
DIFFICULTY: 383
FIRMA: 4f7d3f6371c425d0959456bec42378c
ID NODO SELLER: 1
NODO BUYER: 3
JOLLAR INVIATI: 2
-----
BLOCCO TRE
-----
ID: 3
HASH: 28a37bd2f8bc189a93733fa3ab1488ae
HASH PRECEDENTE: 0604fd456493ceaae0e2e545863b924
LUNGHEZZA POW: 4
DIFFICULTY: 5
FIRMA: c188elfeead9490dd2cf6a59ce44bb03
ID NODO SELLER: 1
NODO BUYER: 4
JOLLAR INVIATI: 2

```

