Distributed Hosting

A Blockchain Based Distributed Hosting Engine

Alex Oberhauser, Stavros Champilomatis and Maria Efthimiadou

Copyright © 2015 Alex Oberhauser, Stavros Champilomatis and Maria Efthimiadou

PUBLISHED BY PUBLISHER

TODO

Licensed under the Creative Commons Attribution-NonCommercial 3.0 Unported License (the "License"). You may not use this file except in compliance with the License. You may obtain a copy of the License at `http://creativecommons.org/licenses/by-nc/3.0`. Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

First printing, April 2015

Contents

I Overview

1 Introduction 7
1.1 Hosting Paradigm Shift 7
1.2 Use Cases 8

2 Ecosystem 9
2.1 Hosting Provider 9
2.2 Complementary Services 9

II Specifications

3 Blockchain Specification 13
3.1 Transaction Definition 13
3.1.1 UPDATE Transaction Definition 13
3.1.2 CLONE Transaction Definition 14
3.2 Block Definition 14
3.3 Mining Process 14

4 Protocol Specification 15
4.1 Blockchain Overlay Network 15

4.2 Data Overlay Network 15
4.3 Hosting Algorithm 15

III Appendix

Bibliography 19

Index 21

I Overview

1 Introduction 7
1.1 Hosting Paradigm Shift
1.2 Use Cases

2 Ecosystem 9
2.1 Hosting Provider
2.2 Complementary Services

1. Introduction

This part gives a motivation and a general overview about the *Distributed Hosting Engine*. It introduces some concepts that will be explained in more detail in the next part. If you are looking for a detailed specification you can directly go to Part II.

1.1 Hosting Paradigm Shift

This whitepaper specifies a new protocol and blockchain that has the potential to change the current hosting paradigm. Instead of retrieving pages from a specific location, they are hosted directly on end-user devices. The distributed hosting algorithm and protocol, specified here, takes care of the optimal location for each page. By taking into account different metrics, like response times, availability and/or relevance, the perfect place for each page will be found after some time. This not only increases the performance for single page accesses, but reduced also network load and hence increases the overall network health.

Additional to the increased performance, the underlying blockchain assures the correctness and validity of each page, also, or especially, if hosted by a random node in the network. This validation mechanism is then used to create a reputation system, allowing blocking of malicious nodes that have a reputation score that is lower than a threshold value.

This mechanism, together with the page distribution algorithm, makes the network self-managing, self-healing and robust against changes.

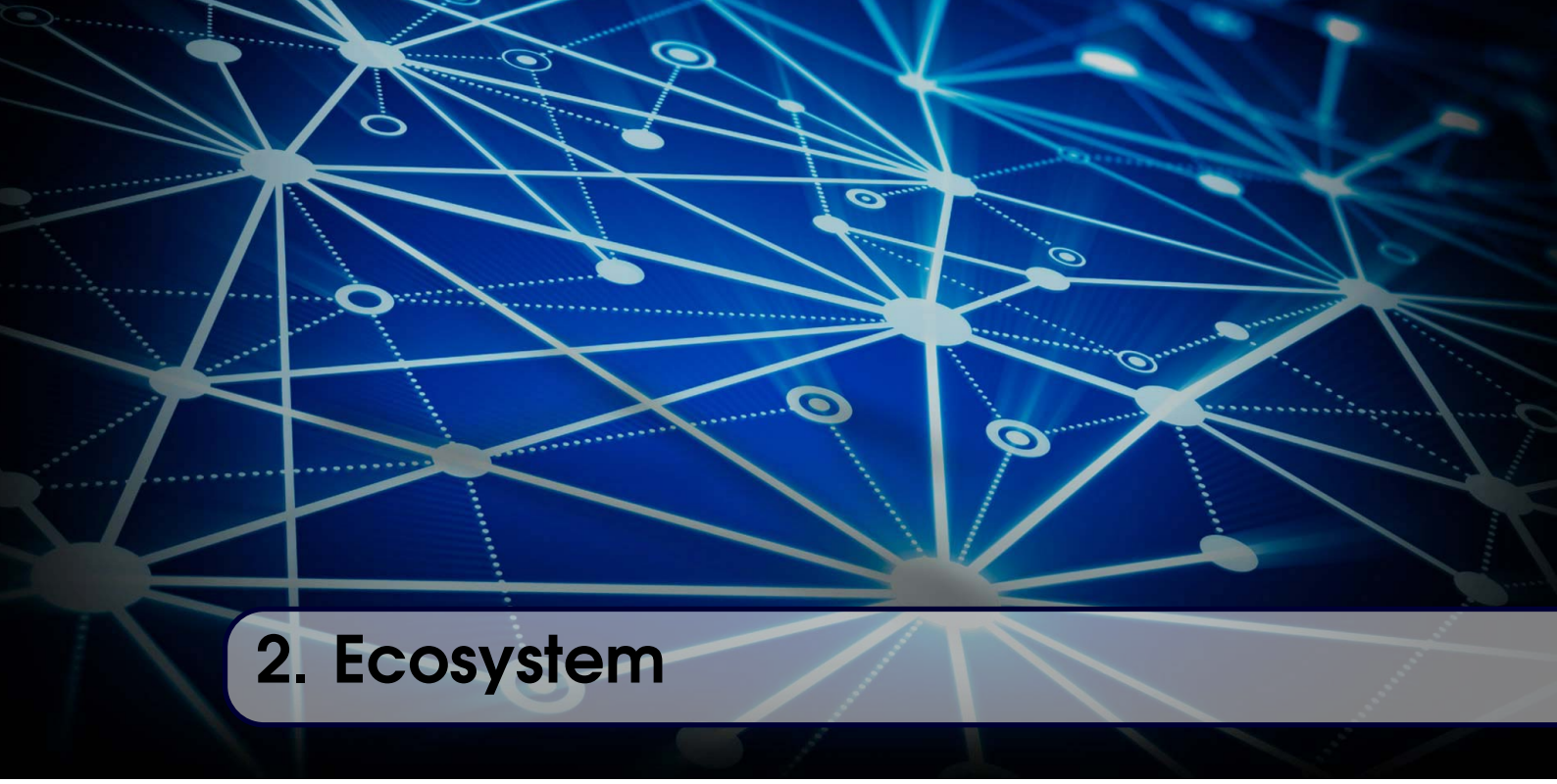
By design the *Distributed Hosting Engine* is censorship resistance, but only under the assumption that there are enough nodes that are willing to host the page. This moves responsibility what could be hosted from a central authority to the collective.

From a commercial point of view the here proposed approach allows the creation of an interoperable hosting ecosystem. Each hosting provider acts as clone and has no direct access to the page, but can support the user with additional (paid or free) services and

hosting space. By design a switch from one hosting provider to another one does not affect the hosting of the page, nor the page itself. Technical there is not even a migration happening, only a switch from one page creation tool to another one.

1.2 Use Cases

TODO: Explains here different use cases based on user stories of Jane and John Doe.



2. Ecosystem

2.1 Hosting Provider

TODO: Explain here the role change of hosting providers. For example: A hosting provider does provide service on top of the blockchain, like page generators and frontends for page publications and updates. If a hosting provider decides to host a page, it is only one clone out of many without special role.

2.2 Complementary Services

TODO: Figure out what alternatives services could be provided by the ecosystem. This includes also new business ideas for new types of startups.

II Specifications

3 Blockchain Specification 13

3.1 Transaction Definition
3.2 Block Definition
3.3 Mining Process

4 Protocol Specification 15

4.1 Blockchain Overlay Network
4.2 Data Overlay Network
4.3 Hosting Algorithm

3. Blockchain Specification

3.1 Transaction Definition

This section defines transactions and their relationship to each other and to external data. It also defines a (de-)serialization format to and from the binary format that is transferred over the wire. In Figure 3.1 a chain of transactions, with relevant fields, is shown. This example includes three hosts that have cloned the page and in total three updates.

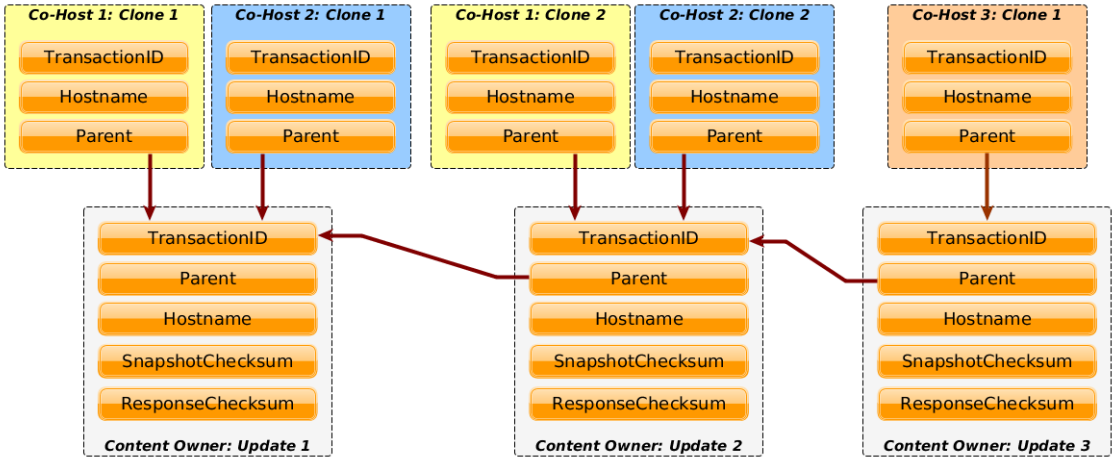


Figure 3.1: Transaction chain example for one page.

3.1.1 UPDATE Transaction Definition

Update transactions can only be created by the page creator. Additional to the proof of ownership they also include a reference to the new content, with related checksums.

TRANSACTION HEADER	
TransactionID	Double SHA256 hash of this transaction.
Parent	The previous update *TransactionID*
ScriptSig	**TODO Future Work:** Proof here ownernship of the hostname and all requirements that come with this update.
Hostname	The tor hidden service onion hostname of this page. Tor hidden services are needed to be able to host page behind firewalls and NATs. Additional they protect the identity of the content owner and more importantly of clone hosts.
CONTENT SECTION	
Snapshot	The hash value of the content. This hash value is used to find all peers in the Distributed Hash Table (DHT) that share currently this snapshot. The DHT is bootstrapped by using all active hostname in the blockchain. For increased compatibility the MAGNET URI scheme is used.
SnapshotChecksum	Checksum of the Snapshot content, used to verify if the downloaded data.
ResponseChecksum	Checksums for all endpoints of this page. Makes each page verifiable and hence could be served by a random node. If the returned content does not match the here checksum the response is invalid.
Flags	**TODO Future Work:** Could be used for differentiation between incremental and full updates, but also indicating that this page is not hosted anymore and could be deleted.

3.1.2 CLONE Transaction Definition

Clone transactions are created and propagated by all hosts that start hosting a page or if they receive a new update transaction. This transactions always link back to exactly one update transaction.

TransactionID	Double SHA256 hash of this transaction.
Parent	The previous update *TransactionID*
ScriptSig	**TODO Future Work:** Proof here ownernship of the hostname and all requirements that come with this update.
Hostname	The tor hidden service onion hostname of this page. Tor hidden services are needed to be able to host page behind firewalls and NATs. Additional they protect the identity of the content owner and more importantly of clone hosts.
Flags	Possible values are: *CLONE* when new update was received or *PURGE* if the page is not hosted anymore.

3.2 Block Definition

3.3 Mining Process

4. Protocol Specification

4.1 Blockchain Overlay Network

4.2 Data Overlay Network

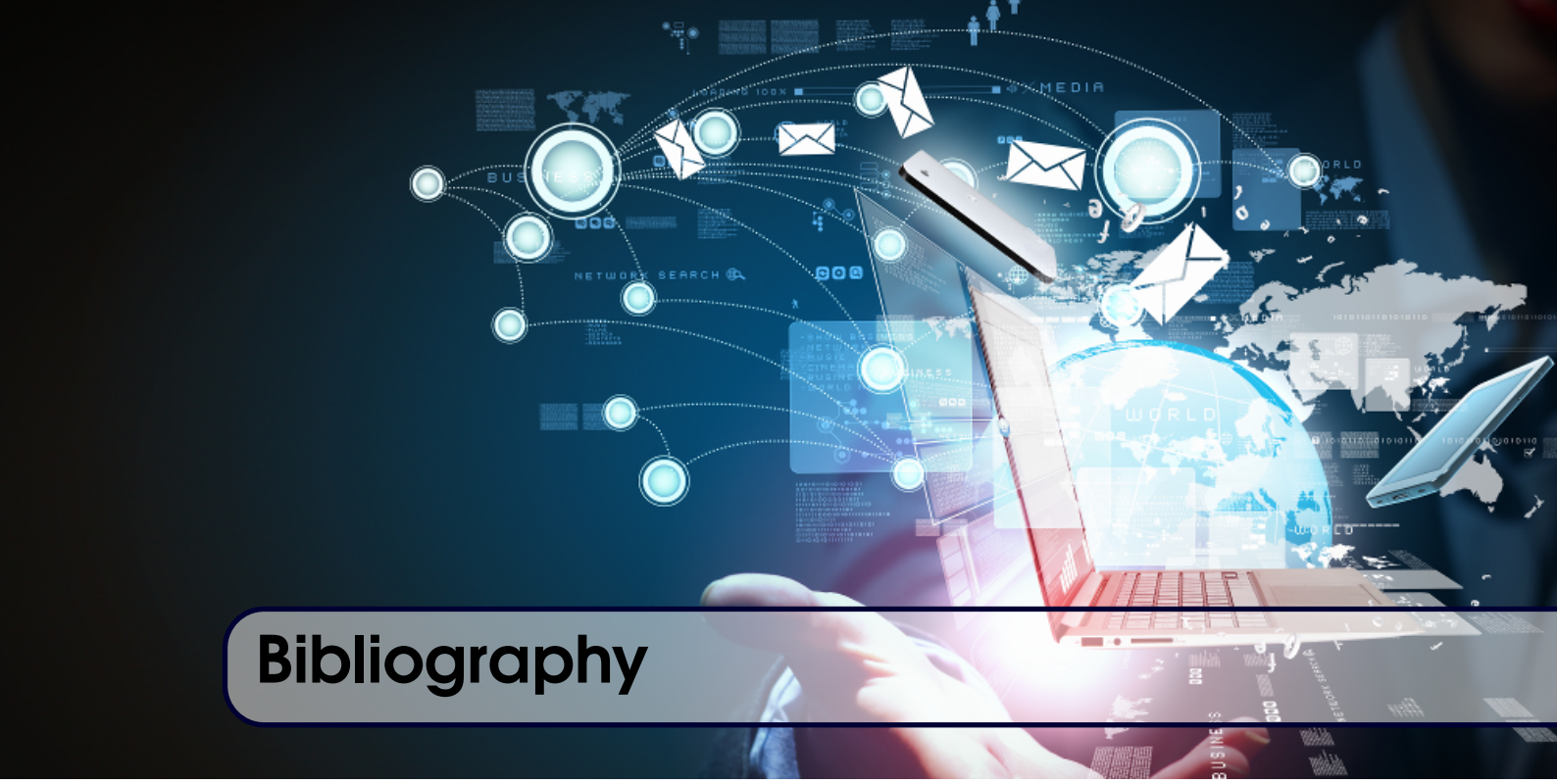
4.3 Hosting Algorithm

III

Appendix

Bibliography 19

Index 21



Bibliography

[08] "Bitcoin: A peer-to-peer electronic cash system". In: *Consulted* 1 (2008), page 2012.

Index

B

Block Definition 14
Blockchain Overlay Network 15

C

Complementary Services 9

D

Data Overlay Network 15

H

Hosting Algorithm 15
Hosting Paradigm Shift 7
Hosting Provider 9

M

Mining Process 14

T

Transaction Definition 13

U

Use Cases 8