NeoFS: Practical introduction

NEO Saint Petersburg Competence Center
Alexey Vanin, Stanislav Bogatyrev

Neo Frontier Launchpad
June 2021

nspcc.ru

Neo St. Petersburg Competence Center

R&D company based in St. Ptersburg, Russia

- Started July 27, 2018
- Deep expertize in data storage and distributed systems
- Close collaboration with universities
- Team of experiences researchers and engineers
- Maintenance of core Neo Blockchain infrastructure
- Neo Core development
- NeoFS, NeoGo and smaller ecosystem projects

NeoFS

Overall view

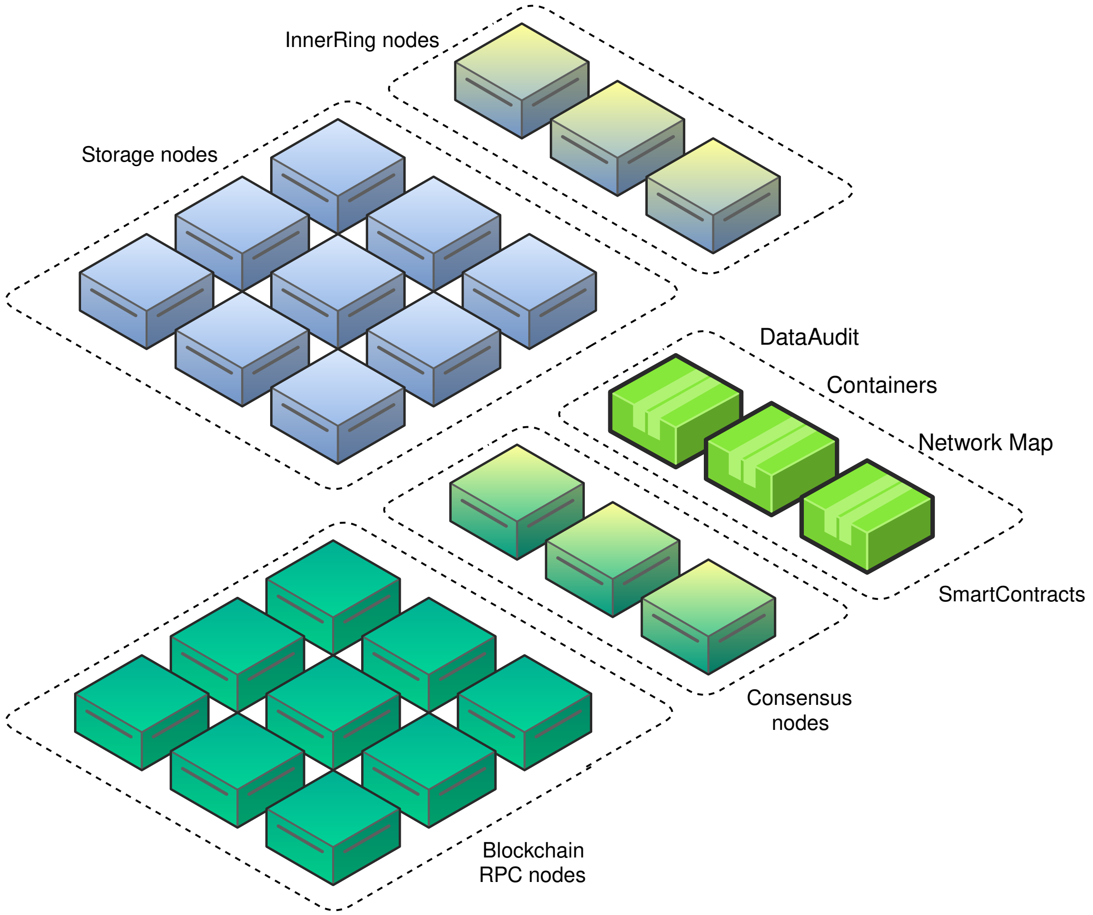
Decentralized object storage

InnerRing and Storage node types

Actions synced via Blockchain

gRPC API, support for S3 и HTTP

Storage Policy and ACL support



How does NeoFS store data?

Containers, Objects and their attributes

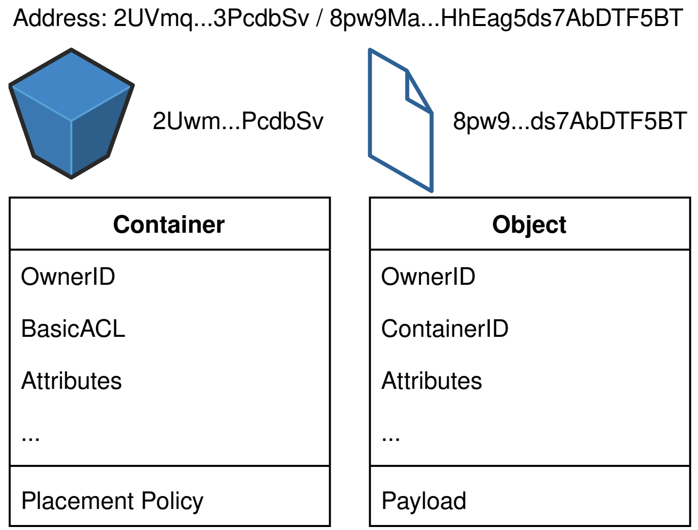
Address: ContainerID/ObjectID

ContainerID: Hash(Container)
Container: Storage Policy + Attributes

ObjectID: Hash(Header+Payload)
Object: Payload + Attributes

Container – like directory or bucket
Stores onchain (SideChain)

Object – like file
Stored offchain on NeoFS Storage Nodes



Handling large objects

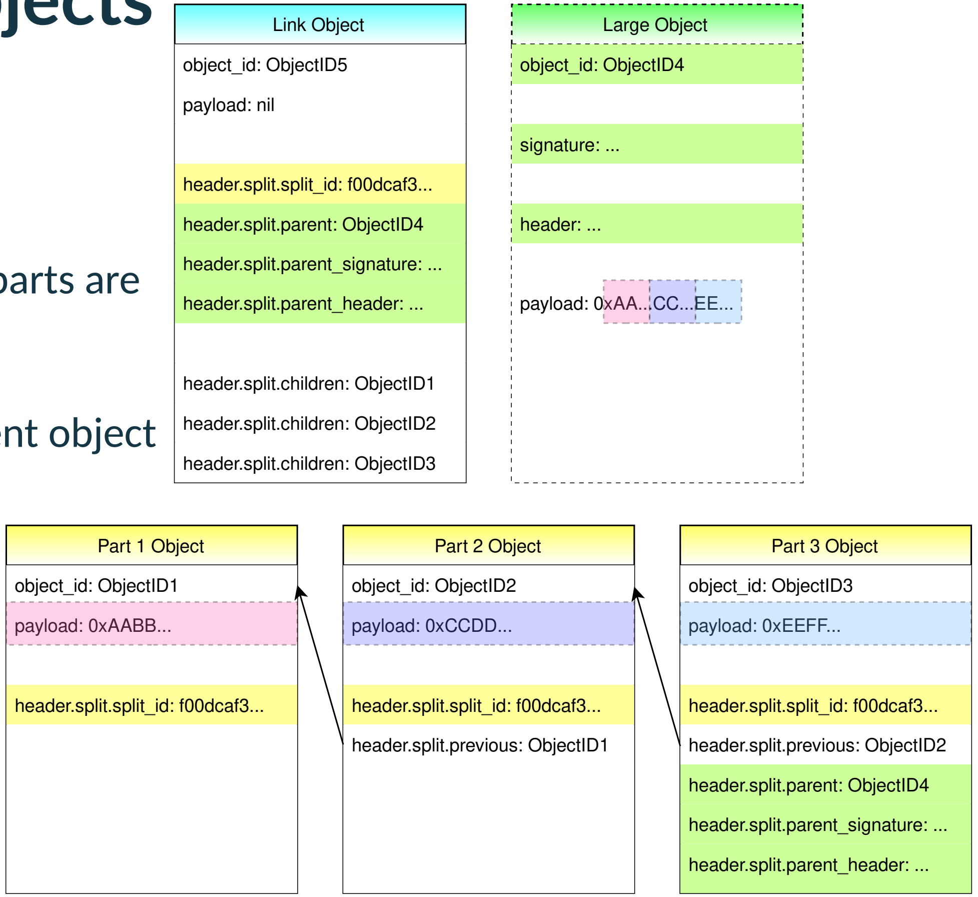
Transparent stream upload

Large objects are split automatically

The whole large object and each of it's parts are available

Steam upload/download with transparent object reconstruction

Compound objects are just objects =)


Link Object
object_id: ObjectID5
payload: nil
header.split.split_id: f00dcaf3...
header.split.parent: ObjectID4
header.split.parent_signature: ...
header.split.parent_header: ...
header.split.children: ObjectID1
header.split.children: ObjectID2
header.split.children: ObjectID3
Large Object
object_id: ObjectID4
signature: ...
header: ...
payload: 0xAA...CC...EE...
Part 1 Object
object_id: ObjectID1
payload: 0xAABB...
header.split.split_id: f00dcaf3...
Part 2 Object
object_id: ObjectID2
payload: 0xCCDD...
header.split.split_id: f00dcaf3...
header.split.previous: ObjectID1
Part 3 Object
object_id: ObjectID3
payload: 0xEEFF...
header.split.split_id: f00dcaf3...
header.split.previous: ObjectID2
header.split.parent: ObjectID4
header.split.parent_signature: ...
header.split.parent_header: ...

Network Map

Nodes and where to find them

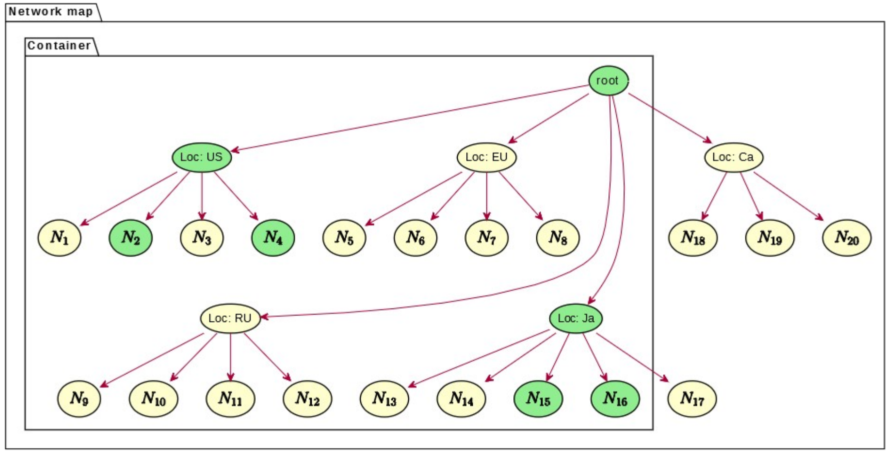
Network map contains active storage nodes

Storage node has a set of Attributes

Storage Policy works with attributes grouped into graph subtrees

HRW selection down to the depth set

New version of netmap is issued by Inner Ring each Epoch and stored in SideChain



```
REP 2 IN OURNODES
CBF 1
SELECT 4 FROM NSPCC AS OURNODES
FILTER "Deployed" EQ "NSPCC" AS NSPCC
```

Storage Policy

Simple rules

Storage Policy works with node attributes from Network Map

Flexible policy language with translation to SQL-like, JSON, Blockly, and more

Replicas distributed between node groups

Selectors define a node group from the network map

```
REP 1 in CORP
REP 1 in EXT
REP 1 in EMP

CBF 4

SELECT 4 Node IN CorpSRV as CORP
SELECT 2 Node IN GoodAliens as EXT
SELECT 6 Node IN EmpWKS as EMP

FILTER CorpProperty EQ ACMECorp AS IsACME
FILTER @IsACME AND NodeType EQ “SRV”  AS CorpSRV
FILTER @IsACME AND NodeType EQ “WKS”  AS EmpWKS
FILTER Rating GE 5 AS GoodAliens
```

Data Audit

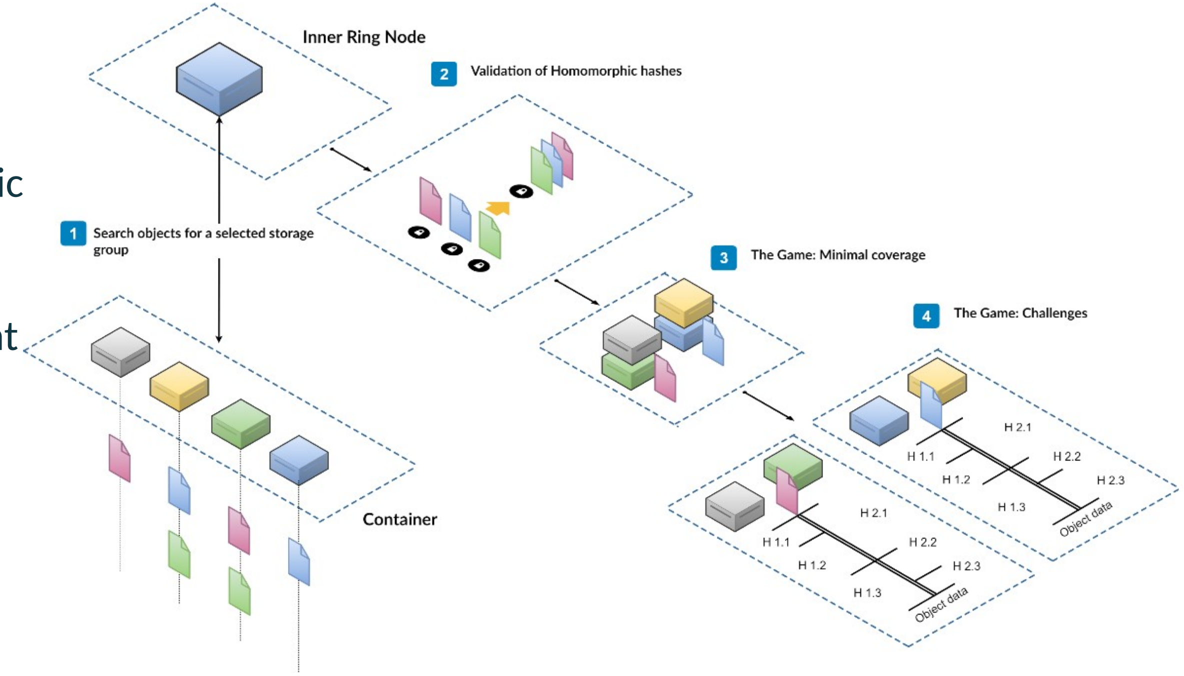
Multi stage game

Audit without data disclosure

ZK Proof based on homomorphic hashing

Works in untrusted environment

Traceable audit results onchain



Access Control List

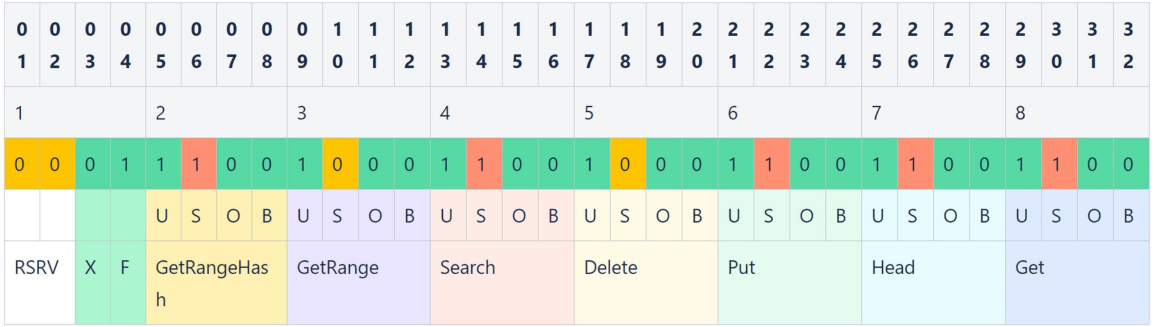
BasicACL

Basic ACL for each container

Immutable

May be extended

Looks like POSIX file Permissions =)



01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
1				2				3				4				5				6				7				8			
0	0	0	1	1	1	0	0	1	0	0	0	1	1	0	0	1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0
				U	S	O	B	U	S	O	B	U	S	O	B	U	S	O	B	U	S	O	B	U	S	O	B	U	S	O	B
RSRV		X	F	GetRangeHash				GetRange				Search				Delete				Put				Head				Get			

Access Control List

ExtendedACL

May be changed, but can’t conflict with Basic ACL

Stored onchain in SideChain

Filters applied to both object and request fields and attributes

```
{
 "records": [
  {
      "operation": "GET",
      "action": "DENY",
      "filters": [
        {
          "headerType": "OBJECT",
          "matchType": "STRING_NOT_EQUAL",
          "key": "Classification",
          "value": "Public"
        }
       ],
      "targets": [
       {
         "role": "OTHERS"
       }
    ]
  }
 ]
}
```

Access Control List

BearerToken

May replace eACL for particular request

Just like JWT in modern Web

May limit the requesters

May be used via HTTP and other protocol gateways

Useful for complex authz cases

```
{
 "body": {
  "eaclTable": {
   "version": {...},
     "containerID": {
      "value":"Ab1X...UeZTz8ZAVtzH5NWfKbC"
   },
   "records": [...]
  },
  "ownerID": null,
  "lifetime": {
   "exp": "100500",
   "nbf": "2",
   "iat": "1"
  }
 },
 "signature": {
  "key": "A/ZjE3Ys1tvs6vVP7wPTxwqHYFIgoDhRnoeNBUwTq9Tq",
  "signature": "BCDU1mdfKmo9xVv/8H8XOmiSQYN...uvtGNXTuDCWgHuSHZUcwqHtvXjsltBbl4gA0="
 }
}
```

Practical demonstration

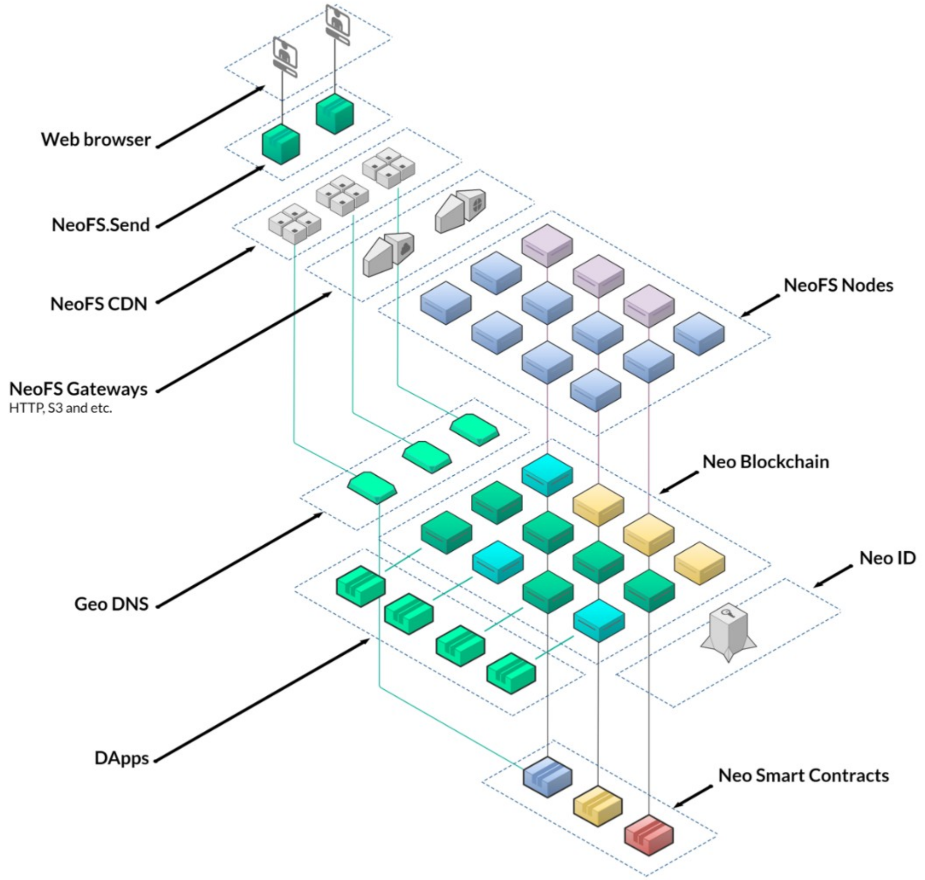
Get in touch with NeoFS

DEMO

NeoFS.Send

Web Application example

- Fully functional modern web app
- Serverless mode for real DApps
- Simple CDN example



NeoFS.CDN

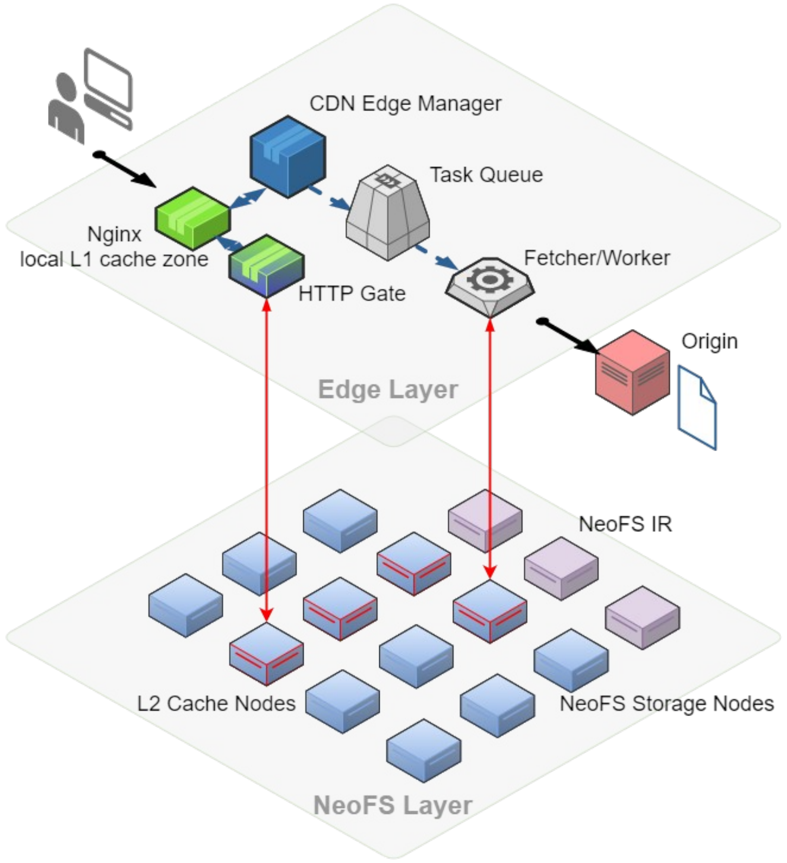
Multi-level caching

GeoDNS

SmartContracts in control

Optimal data placement with NeoFS Storage Policy

Local regulations support



Thank You!

Q&A

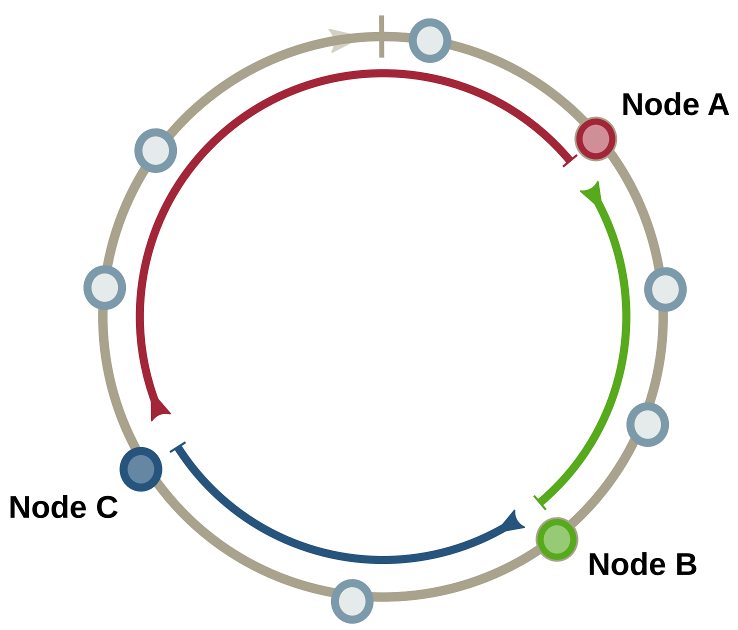
E-mail:

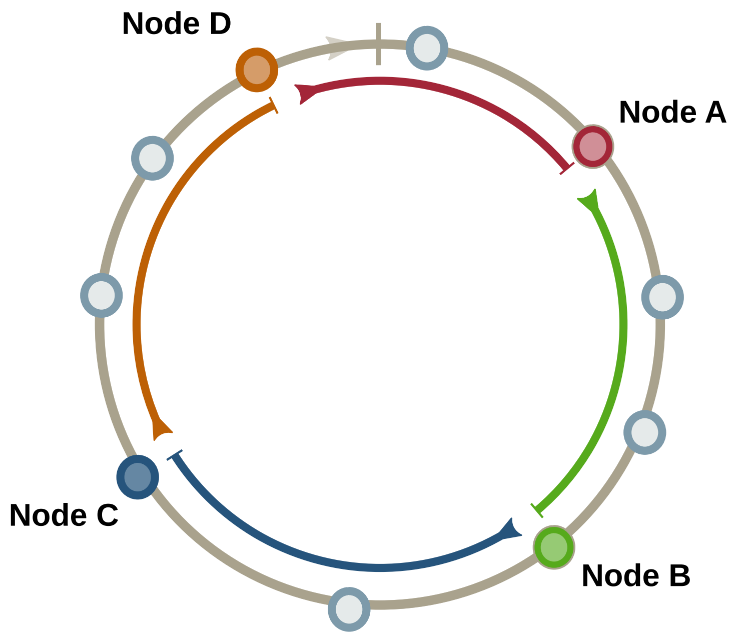
- *stanislav@nspcc.ru*
- *alexey@nspcc.ru*

Neo SPCC:	https://nspcc.ru
NeoFS:	https://fs.neo.org
	https://github.com/nspcc-dev/neofs-node
NeoGo:	https://github.com/nspcc-dev/neo-go
GitHub:	https://github.com/nspcc-dev/
Medium:	https://medium.com/@neospcc/

DHT and HashRing

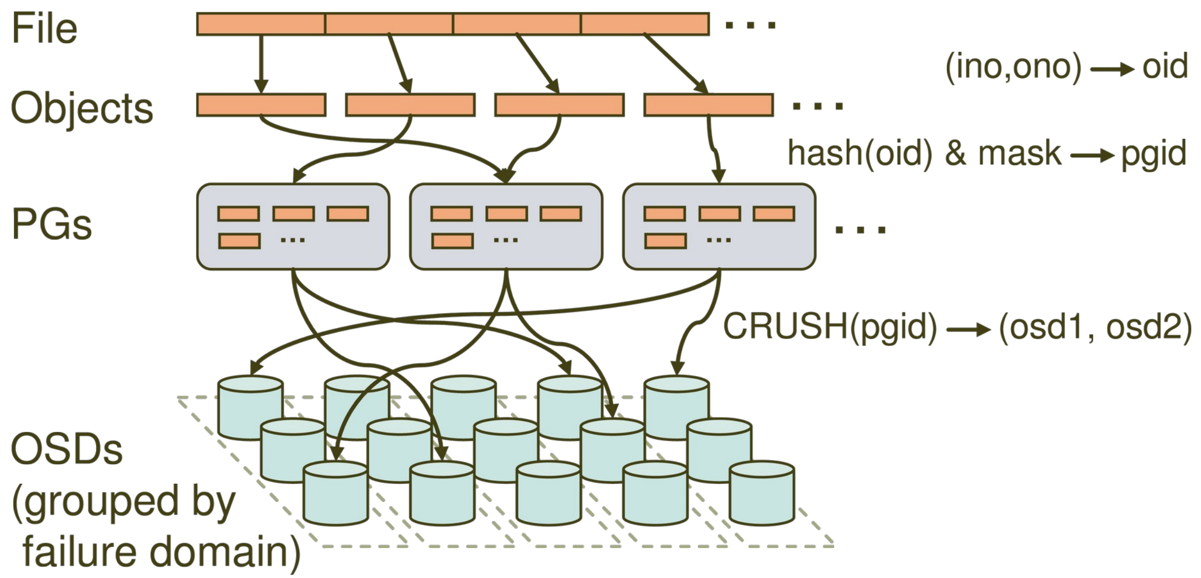
Balance and data migration





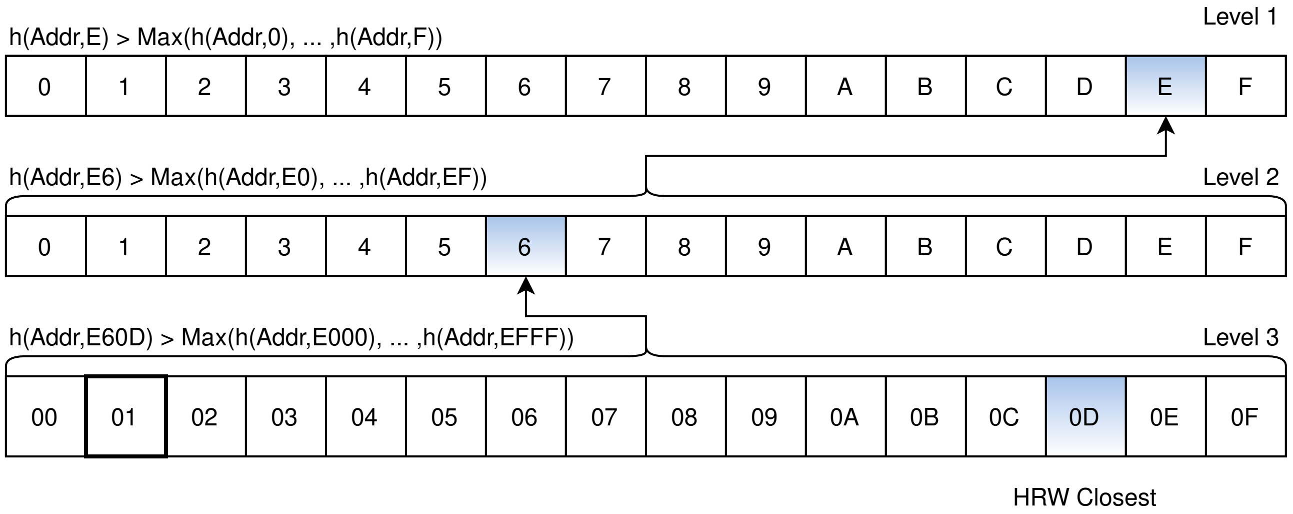
CRUSH in Ceph

Policy for the whole cluster



Rendezvous hashing

Calculating hash distance


Level 1
h(Addr,E) > Max(h(Addr,0), ... ,h(Addr,F))
0
1
2
3
4
5
6
7
8
9
A
B
C
D
E
F
h(Addr,E6) > Max(h(Addr,E0), ... ,h(Addr,EF))
Level 2
0
1
2
3
4
5
6
7
8
9
A
B
C
D
E
F
h(Addr,E60D) > Max(h(Addr,E000), ... ,h(Addr,EFFF))
Level 3
00
01
02
03
04
05
06
07
08
09
0A
0B
0C
0D
0E
0F
HRW Closest

Storage Engine

Rendezvous with Blobovnicza

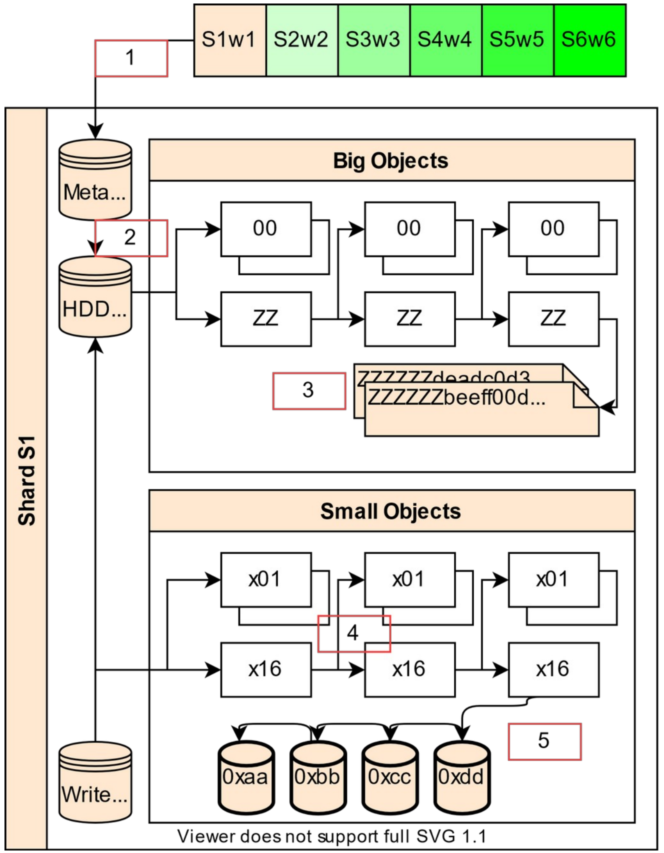
1. Shards with HRW data placement

2. Local placement cache

3. Shallow dir structure placement

4. Hash-based placement in Blobovnicza

5. HRW-based placement in Blobovnicza



S1w1
S2w2
S3w3
S4w4
S5w5
S6w6
1
Shard S1
Meta...
2
HDD...
Big Objects
00
00
00
ZZ
ZZ
ZZ
3
ZZZZZZdeadc0d3...
ZZZZZZbeeff00d...
Small Objects
x01
x01
x01
4
x16
x16
x16
5
0xaa
0xbb
0xcc
0xdd
Write...
Viewer does not support full SVG 1.1