Lecture 17

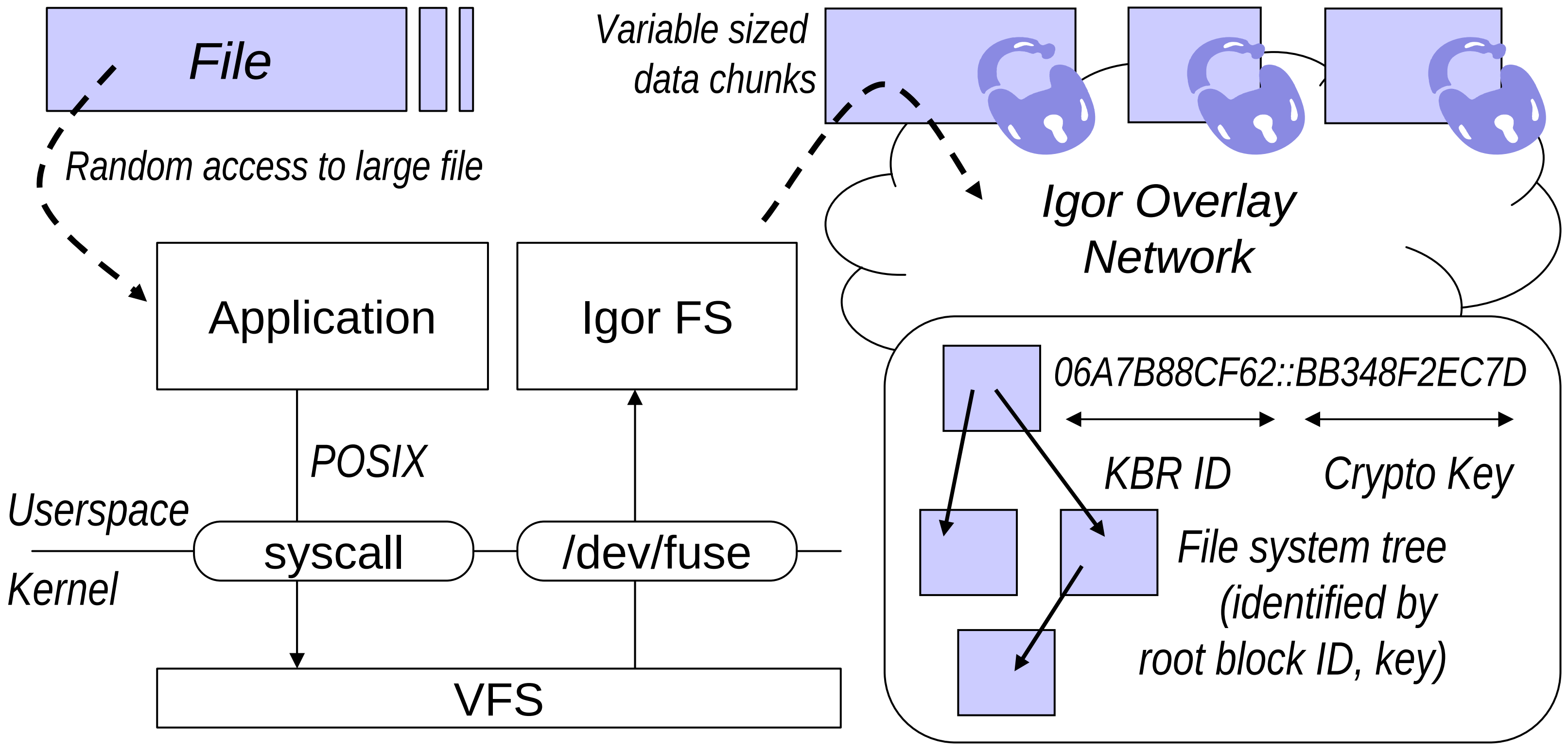
Internet Protokolle II

Practical P2P Systems

Thomas Fuhrmann

Network Architectures
Computer Science Department
Technical University Munich

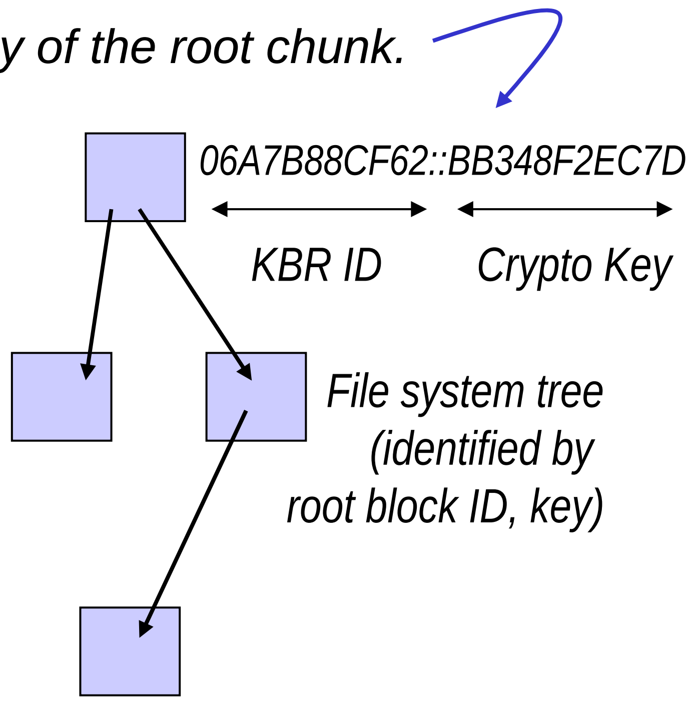
IGOR-FS is a File System



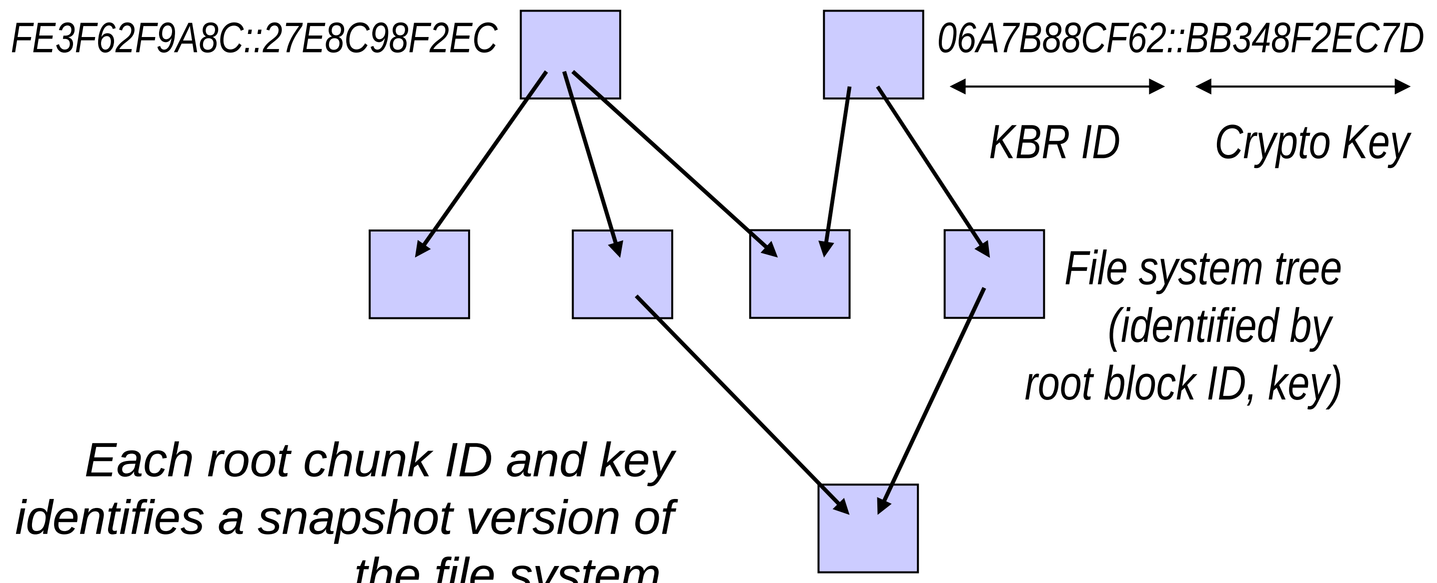
Backup & Versioning (1)

1. Obtain ID and key of the root chunk.

2. Root chunk then contains IDs and keys of the files (and folders) of the file system's root level, and so on …



Backup & Versioning (2)



Each root chunk ID and key identifies a snapshot version of the file system.

New chunks only reflect modifications in content and structure.

IGOR Plugins

Plugin Interface

`cMessagePlugin`
`Register new message types`

`cPolicyPlugin`
`Routing table policies, e.g.`
- `ConnectionOpened()`
- `ConnectionClosed()`
- `EvalForRouting()`
- `EvalForConnectionDrop()`
- `EvalPotentialConnection()`

`cPeekPlugin`
`Look at incoming and outgoing messages`

`cTaskPlugin`
`Do something at a regular basis`

Plugins are used for …

Limitation Policy:
Limit the number of Connections
Fix Fingers Task:
Obey the constraints of the underlying routing geometry
Dump Cache:
Write active connections to a file and bootstrap from it
Service Task:
Provide not only a specific service, provide *n* services
NAT:
Try to handle NAT
SSL:
Encrypt / Compress Connections
Proximity:
Optimize routing geometry

IGOR’s Library Interface – Overview

```
libigor:

    igor_socket()
    igor_bind()
    igor_unbind()
    igor_close()
    igor_sendto()
    igor_recvfromto()
    igor_register_callback_neighborset_change()
    igor_get_nodeid()

libdht:

    libdht_put() bzw. libdhq_push()
    libdht_get()
    libdht_erase()
```

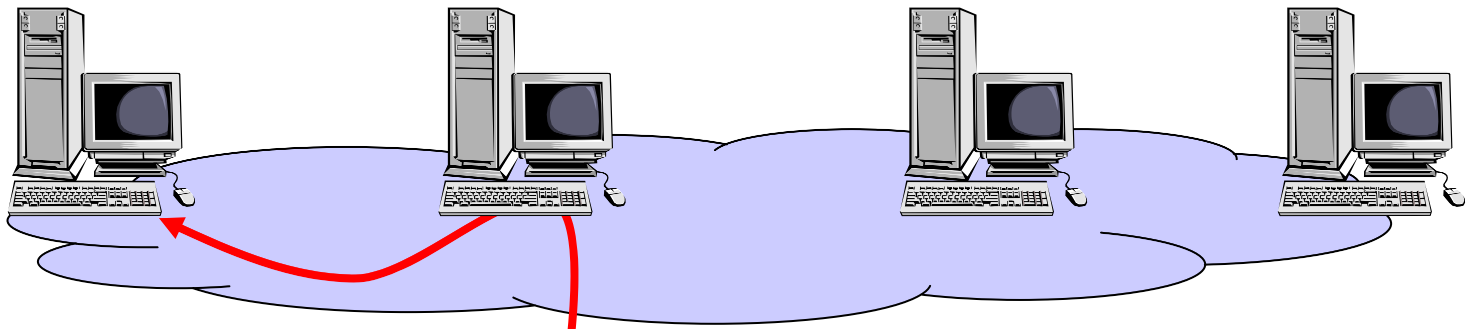
IGOR’s Library Interface

- `igor_socket()` creates an IGOR socket, i.e. the internal data structures needed in the library.
- `igor_bind()` associate an IGOR socket with an application type. The socket will only receive messages with this application type. Moreover, messages sent via this socket carry the same application type.
- `igor_unbind()` unbinds the IGOR socket so that it will no longer receive messages for this application type. Nevertheless, it will be able to forward messages to support a graceful exit of an application.
- `igor_close()` closes the IGOR socket. A

Publish/Subscribe with IGOR-FS

- IGOR-FS original design goals:
  - Sites publish content in release cycles
  - Subscribers obtain access key out of band, e.g. from web site or with e-commerce system.
- Meanwhile, IGOR-FS has become more flexible
  - Sites publish content continuously
  - Customers subscribe on per-time basis
- Thus …
  - Publishers push IDs and keys to the subscribers
  - Publishers can add and remove subscribers
- In other words, we want: …
  - encrypt content only once
  - all authorized clients can decrypt
  - all unauthorized clients cannot decrypt
  - small message overhead
  - low computational overhead
  - high security
  - archival of all old root-blocks
- What did we get:
  - stateless receivers
  - ½ $\log^2 N$ keys per user
  - maximum message size is 2r
  - full security, even if all revoked users cooperate
  - algorithm only used to exchange current key
  - actual encryption with AES
  - similar to multicast key exchange

IGOR-FS Feature Summary



IGOR-FS works automatically and fully transparently as a distributed file system – as if you had magically already downloaded everything you need.

Distributed storage

Strong cryptography

Backup & versioning

Questions?

Thomas Fuhrmann

Department of Informatics
Self-Organizing Systems Group
c/o I8 Network Architectures and Services
Technical University Munich, Germany

fuhrmann@net.in.tum.de