The TPL Mission:

We Bring Customized Cloud Technology to Your Private Data Centers

Tim Waizenegger
University of Stuttgart,
Technology Partnership Lab,
Universitätsstr. 38,
70569 Stuttgart, Germany
tim.waizenegger@ipvs.uni-stuttgart.de
http://www.ipvs.uni-stuttgart.de

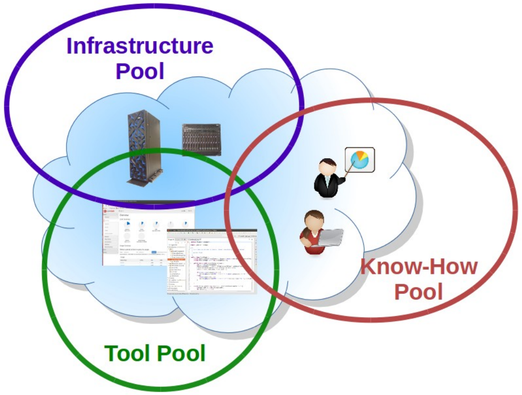
Agenda

- **Introduction – what is the TPL?**
- Our Projects: ECM on Cloud
- Our Projects: ECM DSL
- Our Projects: SDOS

What is the TPL?

- The **Technology Partnership Lab** is part of the University of Stuttgart Cooperative Research Campus
- It provides an umbrella organization for conducting research projects with industry partners
- For more information search for *„tpl uni stuttgart“*
- Find us on youtube, search for *„ibm uni stuttgart“*

Our Goals



- Provide infrastructure, software-tools and know-how in order to successfully conduct research projects
- Bring industry experience and current problem statements into the university curriculum
- Finalize and polish research findings and transfer them to industry products

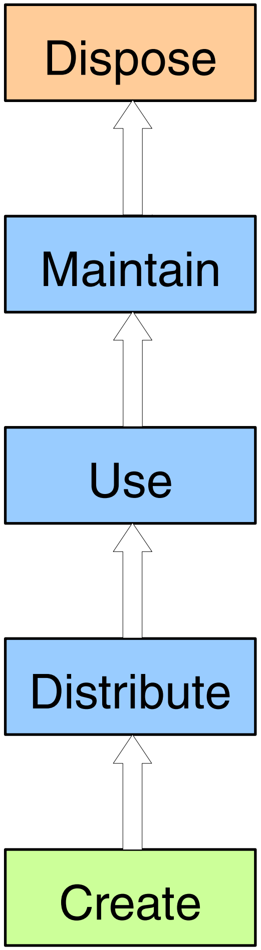
Agenda

- Introduction – what is the TPL?
- **Our Projects: ECM on Cloud**
- Our Projects: ECM DSL
- Our Projects: SDOS

Our Projects – ECM on Cloud

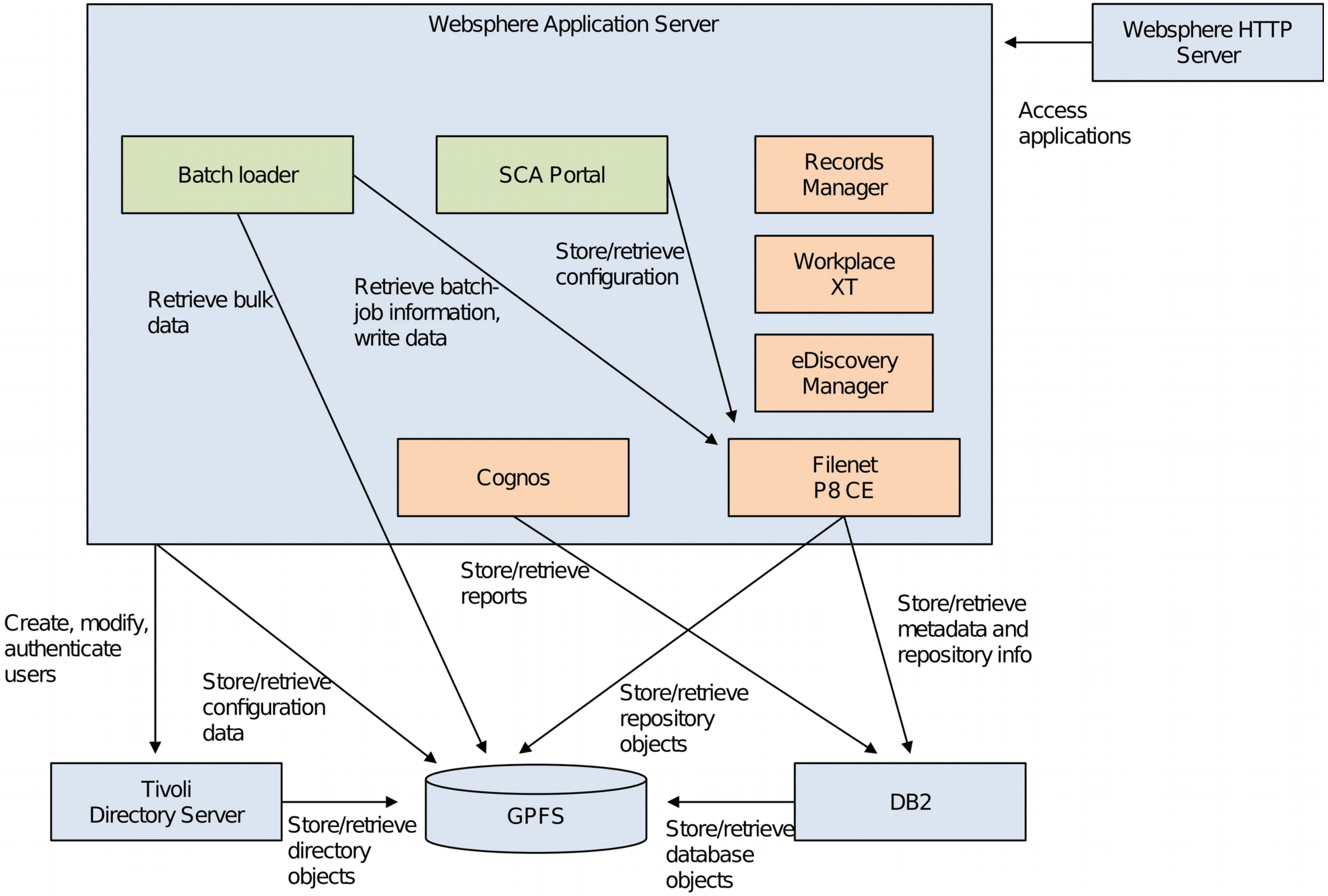
What is Enterprise Content Management?

- Manage all life-cycle stages of electronic content in an enterprise
- The functionality of these systems varies widely
- Many different software components comprise an ECM system
- The system topology and component configuration is customized
- The components are integrated with other IT-systems



Information life cycle

Our Projects – ECM on Cloud


Websphere Application Server
Websphere HTTP Server
Access applications
Batch loader
SCA Portal
Records Manager
Workplace XT
eDiscovery Manager
Cognos
Filenet P8 CE
Retrieve bulk data
Retrieve batch-job information, write data
Store/retrieve configuration
Store/retrieve reports
Create, modify, authenticate users
Store/retrieve configuration data
Store/retrieve repository objects
Store/retrieve metadata and repository info
Tivoli Directory Server
GPFS
DB2
Store/retrieve directory objects
Store/retrieve database objects

Our Projects – ECM on Cloud

What are the advantages of a cloud offering?

- Customers have lower upfront cost, lower entry hurdle for new customers
- Unified, homogeneous installations save overall operational and service cost

What are the challenges?

- We want to re-use existing ECM software, which is designed for single-tenant use
- Some components can be shared between customers, this needs to be evaluated
- Other components need to be instantiated for each new customer, this requires automation
- Operational procedures and best-practices for such an ECM infrastructure do not exist yet

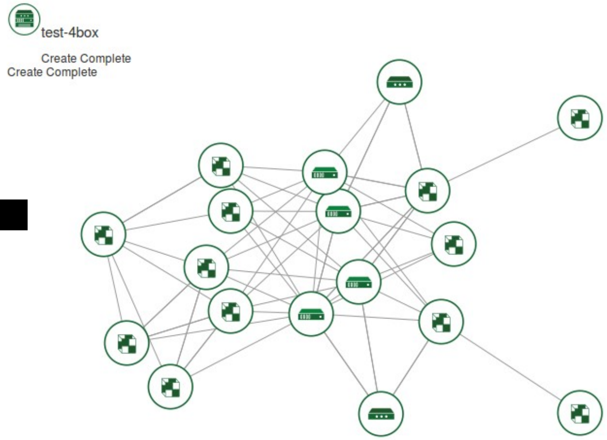
Our Projects – ECM on Cloud

+

openstack™

IBM Cloud Manager with OpenStack

+



IBM PureFlex

private cloud

IBM Cloud Manager & Openstack

Infrastructure management

HEAT

Deployment & management automation

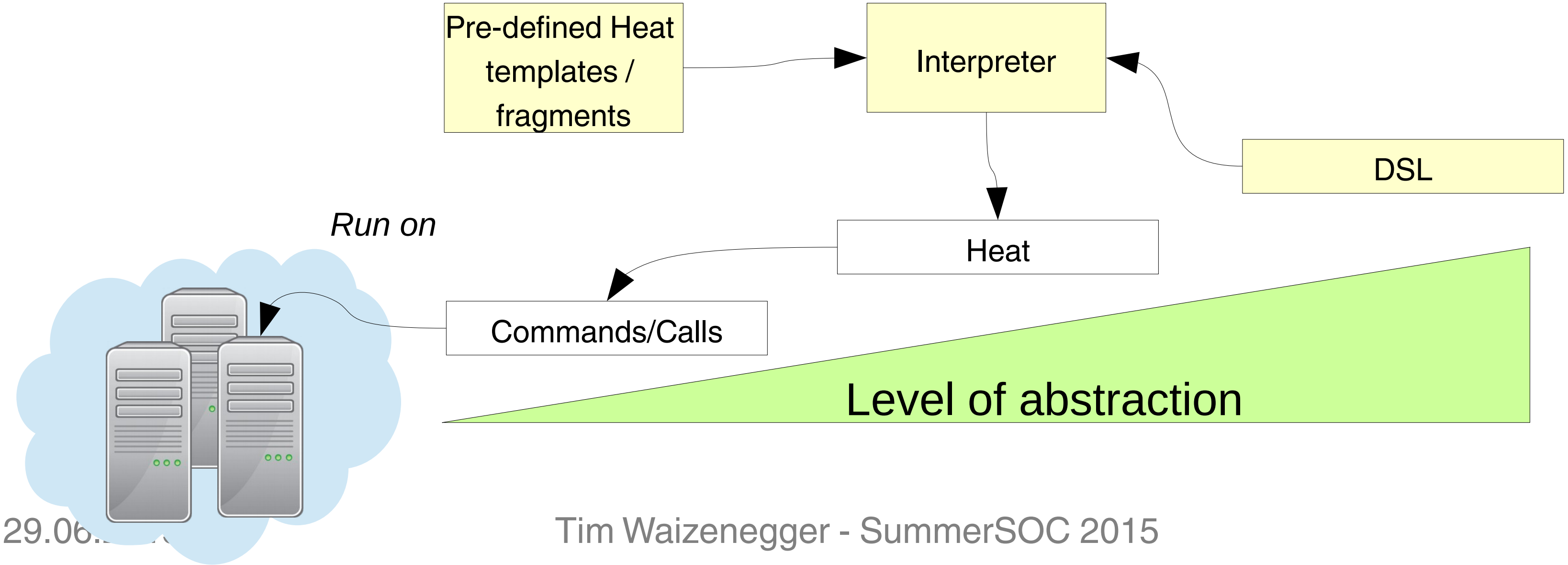
Agenda

- Introduction – what is the TPL?
- Our Projects: ECM on Cloud
- **Our Projects: ECM DSL**
- Our Projects: SDOS

Our Projects – ECM DSL

In this project, we develop a domain specific language (DSL) for describing ECM solutions

- To aid in communicating requirements with customers
- To document, in a formalized way, the specifications of the solution
- To drive our automation system



Pre-defined Heat templates / fragments

Interpreter

DSL

Heat

Run on

Commands/Calls

Level of abstraction

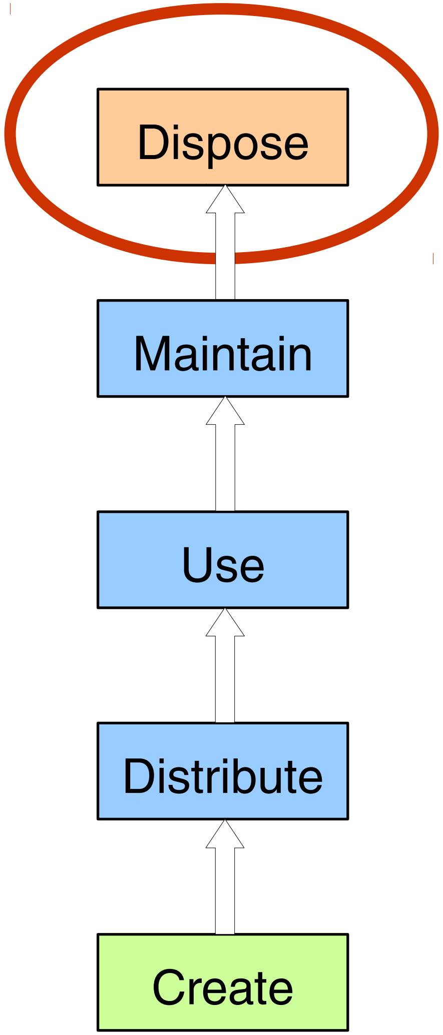
Agenda

- Introduction – what is the TPL?
- Our Projects: ECM on Cloud
- Our Projects: ECM DSL
- **Our Projects: SDOS**

Our Projects – SDOS *(see poster session)*

The Secure-Delete Object Store (SDOS)

- Information life-cycle management defines 5 phases
- After retention or intended use ends, information needs to be disposed of
- Sensitive or compromising information needs to be deleted irrevocably
  - Protecting intellectual property
  - Due to regulations
  - To avoid legal risk



Our Projects – SDOS *(see poster session)*

Garner TS-1 electromagnet

Current Solutions

- Physical destruction of storage media
  - Not possible in shared infrastructures
- Logical destruction of storage sectors
  - Requires low-level access, not possible in virtualized cloud storage scenarios

SEM Model 0101
Sledgehammer Hard Drive Crusher

PGP shredder

Our Projects – SDOS *(see poster session)*

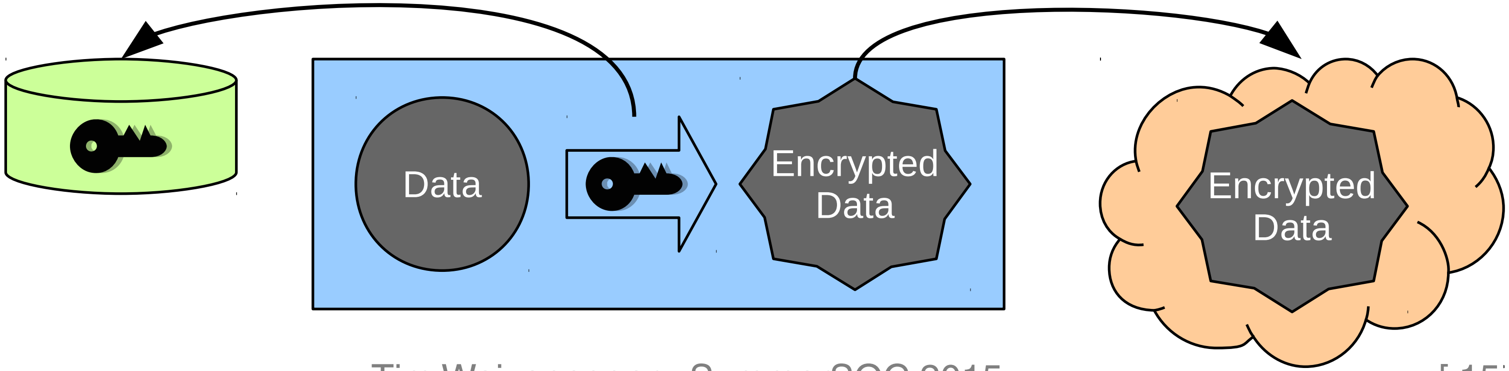
Cryptographic deletion

- The deletion of objects by storing them in encrypted form and securely deleting the encryption key

→ secure deletion only needs to be provided for keys, not large data objects

→ Data can be stored in cheap, untrusted storage systems

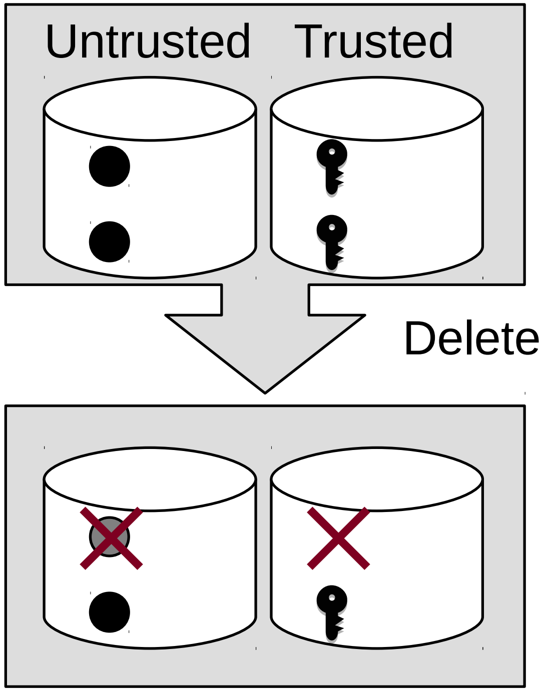
→ Only small encryption keys must be stored in a trusted location



Our Projects – SDOS *(see poster session)*

First approach: *individual per-object keys*

- Generate an individual, random key for each object on insertion (put operation)
- Store all keys in a separate, secure storage system that provides secure deletion
- **Delete operation**
  - Remove key from secure storage system
  - Remove object
- **Overhead**
  - Put: key generation, key storage, encryption
  - Get: key retrieval, decryption
  - Delete: key deletion
  - Secure storage system with a capacity of $n^* s_k$

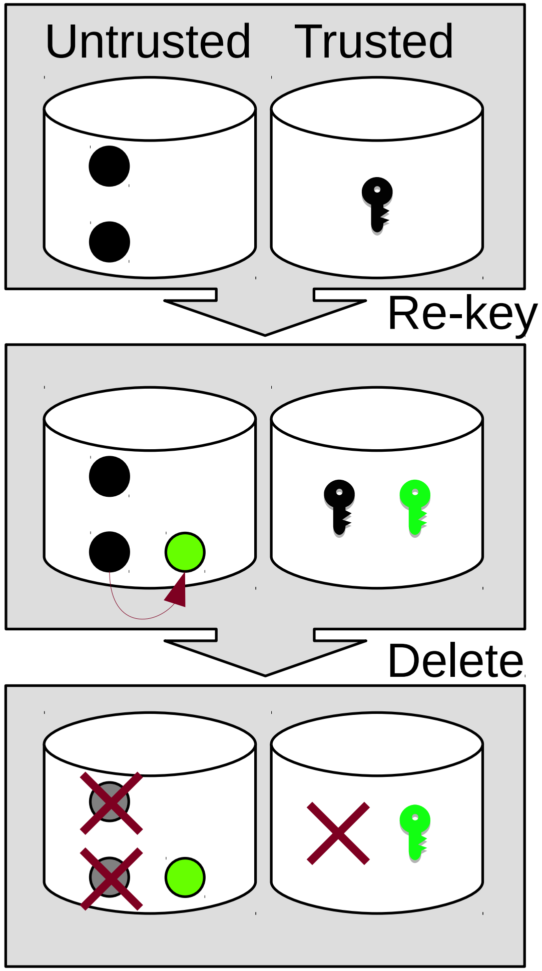


→ **The secure storage system needs to hold a prohibitively large amount of objects**

Our Projects – SDOS *(see poster session)*

Second approach: *single key*

- Generate a single, random key for all objects
- Store key in a separate, secure storage system that provides secure deletion
- **Delete operation**
  - Generate a new key
  - Decrypt all objects (except the one to-delete)
  - Re-encrypt them with the new key
  - Replace key in the secure storage system
  - Remove all old objects
- **Overhead**
  - Put: encryption
  - Get: key retrieval, decryption
  - Delete: re-keying of all kept objects
  - Secure storage system with a capacity of s_k



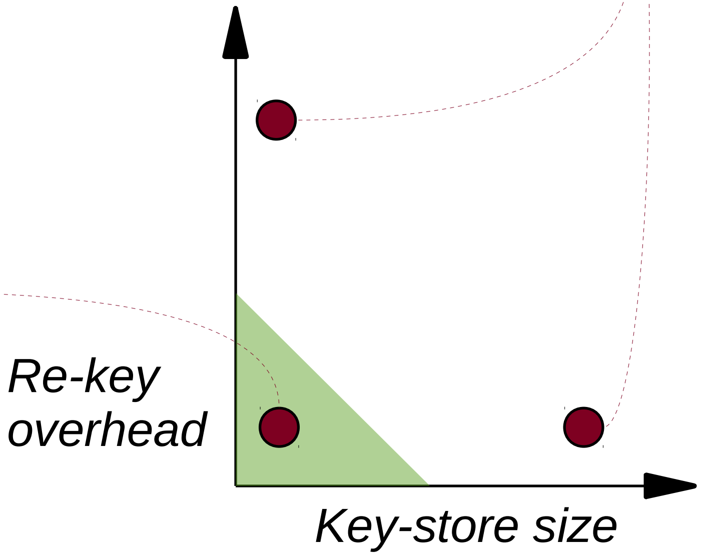
→ **The re-keying operation will become prohibitively expensive in large object stores**

Our Projects – SDOS *(see poster session)*

- The **individual-key approach** requires a prohibitively large trusted key-store
  - But avoids re-keying overhead
- The **single-key approach** has a high re-keying overhead
  - But requires only a small trusted key-store

→ The **key-cascade approach** provides a mechanism for cryptographic deletion with

- Minimal re-keying overhead
- Small key-store size



Thank You!

Tim Waizenegger
University of Stuttgart,
Institute of Parallel and Distributed Systems,
Universitätsstr. 38,
70569 Stuttgart, Germany
tim.waizenegger@ipvs.uni-stuttgart.de
http://www.ipvs.uni-stuttgart.de