This work is funded under National Data Storage 2 project (2011-2013), Project number NR02-0025-10/2011. http://nds.psnc.pl

Full Polish name of the project: System bezpiecznego przechowywania i współdzielenia danych oraz składowania kopii zapasowych i archiwalnych w Krajowym Magazynie Danych
Agenda

• Context: NDS1 & PLATON Popular Archive Service
• Why version 2 of NDS needed?
• **NDS2:**
  – New functionality:
    • secure sharing, publishing and exchanging files
    • versioning, point in time recovery
  – New features:
    • enhanced security,
    • performance scalability,
    • multi-user readiness
• Some observations / open issues
Projects – where we are?

2007             2008             2009            2010            2011            2012            2013

R&D: NDS Design & implementation

Tenders...

Internal tests of NDS system

Tests with users

Production

R&D: NDS2 Design & implementation
High-level aim:

To support scientific and academic community in protecting and archiving the data.

Detailed aims:

– Adressing secondary storage applications:
  • Long-term data archival
  • Short-term backup

– Assumptions:
  • people do have their own primary storage
  • people use another tools for data exchange and CM
NDS1 Design assumptions

- **Focus on specific system features and functionality:**
  - **Long-term data durability** and **consistency**:
    - Physical protection of the data
    - Replication + **safe storage**
    - Keeping **consistency** of the data
  - **Confidentiality** and safety of the data
    - To be supported *(not able to solve all issues)*
  - **Easy usage:**
    - **standard** access methods
    - possible **integration** with existing tools
    - **Transparent** data replication
  - **Stable & reliable** product and service!
    - HA, Trust....
NDS1 – Features & challenges

**HA:**
- Geographically distributed system
- Synchronous and asynchronous replication (reliability vs performance)

**Scalability:**
- performance,
- storage capacity,
- number of users

**Challenges:**
- fault tolerance
- Consistency vs high performance
NDS1 - Architecture

- NDS1 user application
- Access Methods Servers (SSH, HTTPS, WebDAV+)
- Virtual filesystem for data and meta-data (FUSE)
- NDS system logic
- Replica access methods servers
- Storage Node file system
- Replication

Components:
- Meta-data DB
- Users DB
- Accounting & limits DB
- Database Node
- Storage Node
- Slave Meta-data DB
- Access Node
- HSM System
- NDS1 user application
**NDS system – Architecture comments**

- **Data durability** and service availability
  - *Sync & async replication*
  - Multiple data access & storage sites
  - *Monitoring & faults detection*
  - *Limits: no data consistency checking inside the system at the moment*

- **Meta-data durability and consistency:**
  - Multiple meta-data databases instances
  - *Semi-synchronous replication of meta-data*
NDS– Architecture comments/limits

• Data confidentiality
  – Dedicated name spaces
  – Data sharing possible among designated users, limited to a given institution/profile
  – Limit:
    • No support for secure data sharing among institutions
  – NDS1 uses encryption where possible; means: not everywhere!
    • Data access:
      – Scp/Sftp, httpS, WebDAV over httpS
    • Storage:
      – Encryption-enabled tapes (in fact external to system)
  – Encryption outside the system:
    • supported by client application (details later)
  – System-side encryption & data consistency checks to be considered to increase the security of data not encrypted by user
NDS– Architecture comments/limits

- Client side encryption & automation
  - User side B/A application that supports security and automation
    - **Limits: user side encryption is CPU intensive**
      - Some hardware-aid solutions might be necessary for users having a lot of data
      - Additional tools needed:
        - Management features for keys
        - Automation of security-related features
• **Scalability:**
  – **performance:**
    - Many ANs, SNs
    - Many storage devices
    - *Data access optimisation:*
      - Load balancing
      - Monitoring
  – **Limits:**
    - Metadata handling is... centralised for a given logical name space!
    - Consistency vs performance...
  – **Storage capacity:**
    - Many SNs
    - Many storage devices
    - *Cost-effective approach:*
      *HSM as the storage backend*
• **Scalability:**
  - Number of users
    - We can configure multiple system instances when the single system limit is reached
    - Architecture is virtualization-ready
  - Limits:
    - The more users, the more metadata and more complicated user management
    - No real experience from the production system yet
    - Some level of the user management decentralisation is needed
• **Ease of integration / usage:**
  
  **Standard user interfaces:**
  
  - We support: SCP, HTTP/WebDAV, GridFTP
  - Integration with existing tools easy
  - **NDS logic details hidden from the user**

  **Limitations:**
  
  - No ‘special features’ for users through standard interfaces (except meta-data fs)
  - Extra features are to be provided by additional tool / interface:
    - Client backup/archive application
    - Web/GUI Interface
    - E.g. No advanced tools to manage ACLs and sharing

  **Single sign-on:**
  
  - Based on X.509 certificates stored in LDAP
  - Keys and certificates distributed automatically to access methods servers (sshd, apache, gridftp) and converted to appropriate format on-the-fly by KeyFS solution
NDS2 - summary of issues to address

• In NDS2 we need to address (functionalities):
  
  • Advanced features for long-term B/A:
    • Versioning – point in time recovery
  
  • Security and data safety related:
    • Data consistency checks
    • Strong and efficient encryption:
      • on the client side (hardware aid, automated + tools)
  
  • Sharing:
    • Inside NDS (some trust to users assumed)
    • NDS <-> external world (one side of sharing not trusted)
  
  • Publishing data using our infrastructure:
    • e.g. for Digital Libraries
      • they store archives in the in NDS already
  
  • Extra functionalities to be offered by extended (non-standard) interface
    • e.g. versions management
    • We still keep standard interfaces working
NDS2 - summary of issues to address

• We need to address (features):

  • **Scalability:**
    • Deal with *metadata handling* scalability
    • but keep consistency untouched!
    • common logical view is needed for all users going to share data
      • => *De-centralise* logical name space management:

  • *De-centralise* users management:
    • hierarchical management (not covered in this presentation)
NDS1 – Scalability improvements

- De-centralised logical name space management:
  - Step 1: divide the namespace into parts distributed across **multiple metadata DBs** (dCache-like approach?)

++ load distribution  
++ consistency  
-- single point of failure
NDS1 – Scalability improvements

- De-centralised logical name space management:
  - Step 2: combine distribution with replication

++ load distribution
++ consistency
++ no single point of failure

SSR – Semi Synchronous Replication of meta-data

Meta-data DB A instance 1
Meta-data DB B instance 1
Meta-data DB C instance 1

Meta-data DB B instance 2
Meta-data DB instance 2

Database Node

Users DB
Accounting & limits DB
NDS1 – Scalability improvements

- De-centralised logical name space management:
  - Step 3: combine distribution with replication + provide automated failover

SSR – Semi Synchronous Replication of meta-data

++ load distribution
++ consistency
++ no single point of failure
++ automated failover
NDS2 – New functionalities

• In NDS2 we need to address (functionalities):

  • Advanced features for long-term B/A:
    • Versioning – point in time recovery

  • Security and data safety related:
    • Data consistency checks
    • Strong and efficient encryption:
      • on the client side (hardware aid, automated, + tools)

  • Sharing:
    • Inside NDS (some trust to users assumed)
    • NDS <-> external world (one side of sharing not trusted)

  • Publishing data using our infrastructure:
    • e.g. for Digital Libraries
      • they store archives in the in NDS already

  • Extra functionalities to be offered by extended (non-standard) interface
    • e.g. versions management
    • We still keep standard interfaces working
NDS2 – Versioning + Point-in-time recovery

NDS virtual filesystem structure

open (..., ..., O_RDWR)

NDS virtual filesystem structure

• Versioning is:
  • Transparent to users
    • they may access to old versions through a special directory or mountpoint; similarly (protocol, access point) as the current replica
  • Efficient:
    • We can use one of many existing replicas to handle current operations immediately
NDS2 – Security and data safety functions

- **Data consistency checks:**
  - **Cryptographics shortcuts** calculated for replicas and put to Metadata DB
  - Can be presented to users (to be compared – manually or automatically)
  - **Background checks** made periodically (at least while storing and retrieving the file)
  - Shortcuts calculated close to data (on the storage nodes)
  - *Redundancy coding to be considered*
NDS2 – Security and data safety functions

• Strong and efficient encryption – on the client side
  • **Software-based** – application for individual users
  • **Hardware-aid** – appliance for those having really huge amounts data

• **Both approaches automated + tools provided**
NDS2 – Secure sharing and publishing

- Secure sharing the data inside the NDS2 system

Maciej – trusted NDS2 user

Jan – trusted NDS2 user

NDS virtual filesystem structure

Copy (or link) the file
+ set the ACL

+ setting the ACLs enables the other user to SEE the file

+ key exchange needed to really READ its contents

+ again, we can use multiple replicas to handle modifications
NDS2 – Secure sharing and publishing

- Secure sharing the data with the external World (FileSender-like use case)

Maciej – trusted NDS2 user

Copy the file to the publishing sandbox

'Brian... – UNTRUSTED user

URL provided to Brian
+ some additional info:
  - user/login
  - key for the file...

Sandboxed part of the system

Simplified NDS logic

NDS virtual filesystem structure

Copy the file to the publishing sandbox

System side replication and decryption
- user not involved!

+ sharing effective
- Maciej discloses the file vs the system
+ Maciej’s secret key still not disclosed
NDS2 – Secure sharing and publishing

- Secure publishing... (a special case of sharing)

Maciej – trusted NDS2 user

Copy the file to the publishing sandbox

Anonymous UNTRUSTED users

Public URL generated and published

People can access the file

Sandboxed part of the system

Simplified NDS logic

NDS virtual filesystem structure

+ publishing effective

+ multiple replicas can be effectively served by multiple servers (e.g. Apache) to many users

- Maciej discloses the file vs the system
NDS2 – Secure sharing and publishing

- Secure publishing...
  (practical use case for digital libraries)

**Diagram:**
- Data source
- User / CMS operator
- End user (publication reader)

**Flow:**
1. Data source
2. Meta-data
3. CMS
4. MASTER data
5. NDS2 services
6. High-res presentation versions
7. ‘Regular’ (low-res) presentation versions
   + link to high-res presentation version put to library portal
8. NDS2 sandbox
9. End user (publication reader)
NDS2 – Secure sharing and publishing

• Some observations:
  • Proper key management mechanisms needed:
    • User (e.g. Maciej) owns the **private/public „master” key pair**
    • Each stored file is encrypted using the a **separate file’s symmetric key**
    • Symmetric keys are safely stored in NDS2 Metadata DB (encrypted by master key known only to the user)
  • Sharing steps:
    • Sharing file physically (ACL...) – enables to see the file
    • Retrieving the symmetric file key from the system
    • Decrypting the file’s key (on the user side)
    • Passing the decrypted file’s key to the target user
    • Decrypting the file on the user’s side – enabling to understand the file contents
  • Publishing steps:
    • Retrieving the file’s symmetric key from the system
    • Passing the decrypted key back to the system
    • The system copies the file to sandbox (and decrypts it)
    • The others can see the file and understand it
  • **Sharing mgmt. and key handling tools should be provided to users**
Further observations / open issues:

- Improving system scalability is necessary to implement new features
  - internal sharing require ‘single’ logical view of the shared structure
  - additional metadata processing

- Two versions of NDS2 logic:
  - Regular – for trusted part of the system, feature-full
  - Light – for the sandbox – hack-proof, non feature-full, simple!

- Q: are we going to end-up with two completely different products?

- We have to consider new interfaces:
  - Sandbox’s public interface for downloading/uploading data
    - What about reusing part of Filesender project findings?
  - New features require going beyond standard interfaces:
    - Web GUI
    - (Java?) client-side application
NDS2 – Summary

• In NDS 2 we try to:
  
  • **Address more sophisticated B/A-related needs**
    • Provide transparent versioning
    • Increased security (encryption) and performance (hardware-aided)
    • Improve overall system scalability
  
  • **Provide sharing facilities:**
    • Internal sharing – 2 lines of defence – ACLs on file access + keys for contents
    • File-Sender-like features for NDS users
  
  • **Support publishing:**
    • ‘Private’ publishing use case
    • ‘Massive’ publishing use case
THANK YOU!

More information: nds.psnc.pl

Note: most of the pictures used in the presentation come from sxc.hu service