Implementing a Simple SMF Service: Lessons Learned

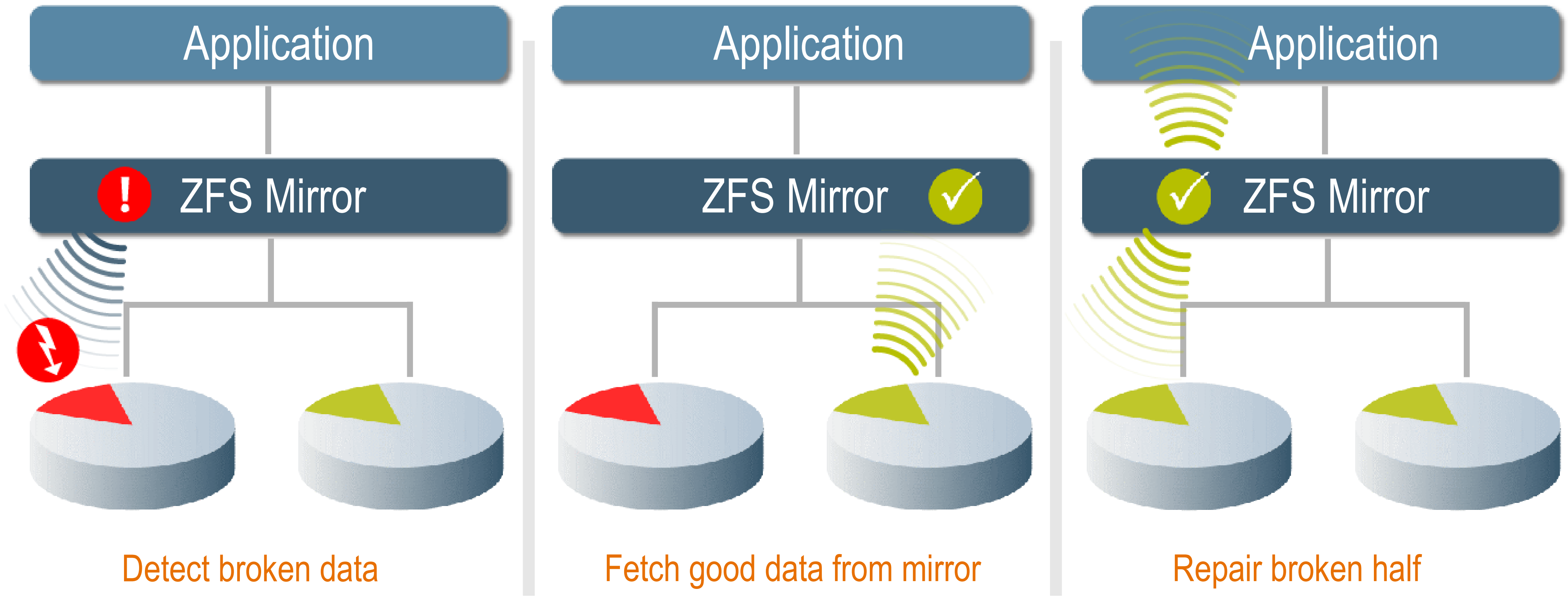
OSDevCon09, October 30th, 2009

Constantin Gonzalez
Principal Field Technologist
Sun Microsystems Germany

Goals

- Make „ZFS pool hygiene“ a 1-click experience
  - Implement a simple SMF service that periodically scrubs pools.
- Learn about SMF and other Solaris features:
  - ksh93, ZFS, SMF, RBAC, IPKG, Visual Panels.
- Motivate more users to use SMF more often
- Have some fun, too!

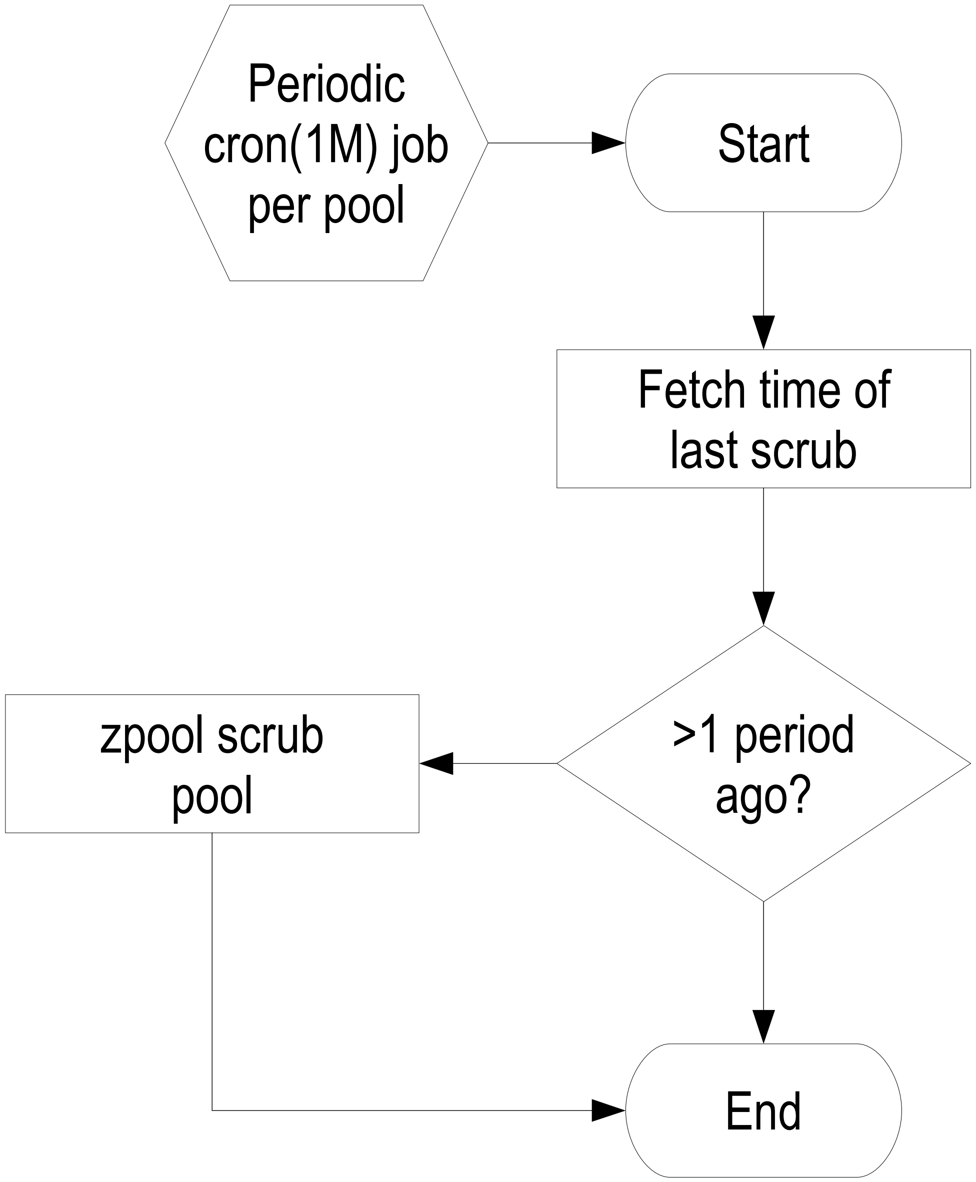
ZFS Self Healing



- You only can repair errors that you see.
- `zpool scrub <pool>` checks all blocks systematically.
- Recommendation: Scrub all your pools periodically.
- Even if you don't use mirroring or RAID-Z.

Can this be done automatically?

A Simple Idea



There's Already Something Similar

- ZFS Auto-Snapshot Service
- Shipped with OpenSolaris
- Basis for the popular ZFS Time-Slider feature
- Let's help ourselves here :).

Service Management Facility

- Since Solaris 10
- Manages all Services of the System (and more)
- Controls boot process and replaces run-levels
- Comfortable framework for:
  - Start/Stop scripts
  - Dependencies
  - Configuration of multiple instances
  - Status/Error messages and resolutions
- Most important commands:
  - `svcs(1), svcadm(1M), svccfg(1M)`

Our Service Therefore Needs

- A start/stop script
- A script for `cron(1M)`

} Can be done as one

- A manifest for SMF (XML-file)

… and we can just borrow, then adapt them from the ZFS Auto-Snapshot Service!

Lesson #1:

It's ok to ~~steal~~ borrow stuff!

What if someone hacks into our script?

Making Our Service More Secure

- Role-based Access Control (RBAC)
- New role `zfsscrub`:
  - Allowed to administer ZFS Pools (not file systems)
  - Allowed to administer the ZFS Auto-Scrub Service
  - Allowed to use normal commands (like a user)
  - Nothing else
- A hacker would only be able to:
  - Destroy/manipulate pools,
  - but **not** take over the system!

Our Service Therefore Needs

- An SMF service “`zfs/auto-scrub`”:
  - A start/stop Script
  - A script for `cron(1M)`
  - A manifest for SMF

$\left.\begin{matrix}\\\\\end{matrix}\right\}$ Combined into one single script

- A new `zfsscrub` role

Lesson #2:

RBAC makes establishing a least-privilege model easy!

How do we want to install our new service?

Scriptless Installation, `pkg(1)`-style

- We may only:
  - Copy files
  - Activate SMF services
- We may not:
  - Directly start scripts
- Why?
  - Less complexity, less errors during installation
    - No special treatment for VMs, zones, hands-off, etc.
    - Simplified installtion
  - More secure
  - Better serviceable

Can't Start Scripts Directly?

- But we may install and activate SMF-Services!
- Therefore: Let's do a new SMF-Service for
  - Creating the new role upon activation,
  - Deactivating itself when done.

Our Service Therefore Needs

- An SMF service "`zfs/auto-scrub`":
  - A start/stop script,
  - A script for `cron(1M)`,
  - A manifest for SMF.

} Combined into one script

- Another SMF service "`zfsscrub-roleadd`":
  - A start/stop script,
    - creates the role `zfsscrub`, then deactivates itself,
  - A manifest for SMF.

Lesson #3:

We can cheat around IPKG by packing our install scripts Into SMF services.

Let's Get Started, Then!

Wait, when did that last scrub happen, BTW?

zpool(1M) status

```
constant@fridolin:~$ zpool status testpool
  pool: testpool
 state: ONLINE
 scrub: scrub completed after 0h0m with 0 errors on
Wed Sep 16 09:33:42 2009
config:

     NAME                     STATE     READ WRITE CKSUM
     testpool                 ONLINE       0     0     0
       /export/stuff/disk1    ONLINE       0     0     0

errors: No known data errors
```

After Reboot or `zpool export`:

```
constant@fridolin:~$ zpool status testpool
  pool: testpool
 state: ONLINE
 scrub: none requested
config:

        NAME                    STATE     READ WRITE CKSUM
        testpool                ONLINE       0     0     0
          /export/stuff/disk1   ONLINE       0     0     0

errors: No known data errors
```

New CR 6878281 opened:
“zpool should store the time of last scrub/resilver and other zpool status info in pool properties.”

What do we do now?

Add Another SMF Service

- `zfs/scrub-track`
  - Runs once per hour (through `cron(1M)`)
  - Until `zpool scrub` is finished
  - Stores finish time in a ZFS property in the topmost ZFS filesystem of the pool
    - Needs „ZFS Filesystem Mgmt“ profile for `zfsscrub`

    and deactivates itself
- `zfs/auto-scrub`
  - checks `zpool status` **and** the new property.
  - Activates `zfs/scrub-track` at every `scrub`

BTW

- Zpool supports properties:

-
  ```
  constant@fridolin:~$ zpool get all testpool
  NAME      PROPERTY       VALUE        SOURCE
  testpool  size           504M         -
  testpool  used           243M         -
  testpool  available      261M         -
  testpool  capacity       48%          -
  testpool  altroot        -            default
  testpool  health         ONLINE       -
  testpool  guid           4748598414767023039  default
  testpool  version        18           default
  testpool  bootfs         -            default
  testpool  delegation     on           default
  testpool  autoreplace    off          default
  ...
  ```
- But no user-defined ones!
- Workaround: Use the top-level ZFS filesystem
- Bug? RFE? Not an issue?

Our Service Therefore Needs

- An SMF service “`zfs/auto-scrub`”
  - A start/stop/cron script
  - A manifest for SMF
- Another SMF service “`zfsscrub-roleadd`”
  - A start/stop script for creating `zfsscrub`
    - `ZFS Storage Management,`
      `ZFS File System Management`
  - A Manifest for SMF
- Yet another SMF-Service “`zfs/scrub-track`”
  - A start/stop/cron script, similar to `zfs/auto-scrub`
  - A manifest for SMF

Lesson #4:

Bugs and RFEs show up in unexpected places...

Lesson #5:

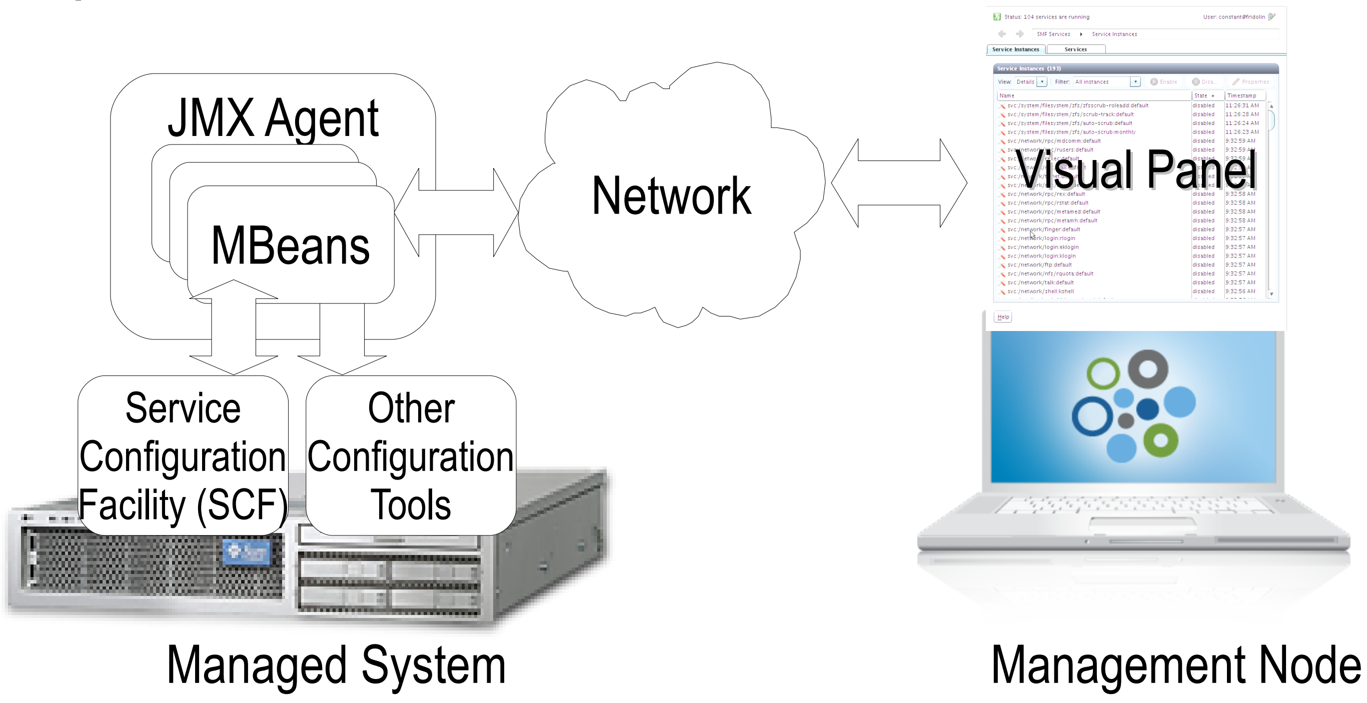
If in doubt, do it in SMF!

Now, let's take a look!

Final touches: A GUI!

OpenSolaris Visual Panels Project

- New framework for central management of system configurations
- Based on Java Management Extensions (JMX)
- Interacts with Service Configuration Framework (SCF), a part of SMF, and others



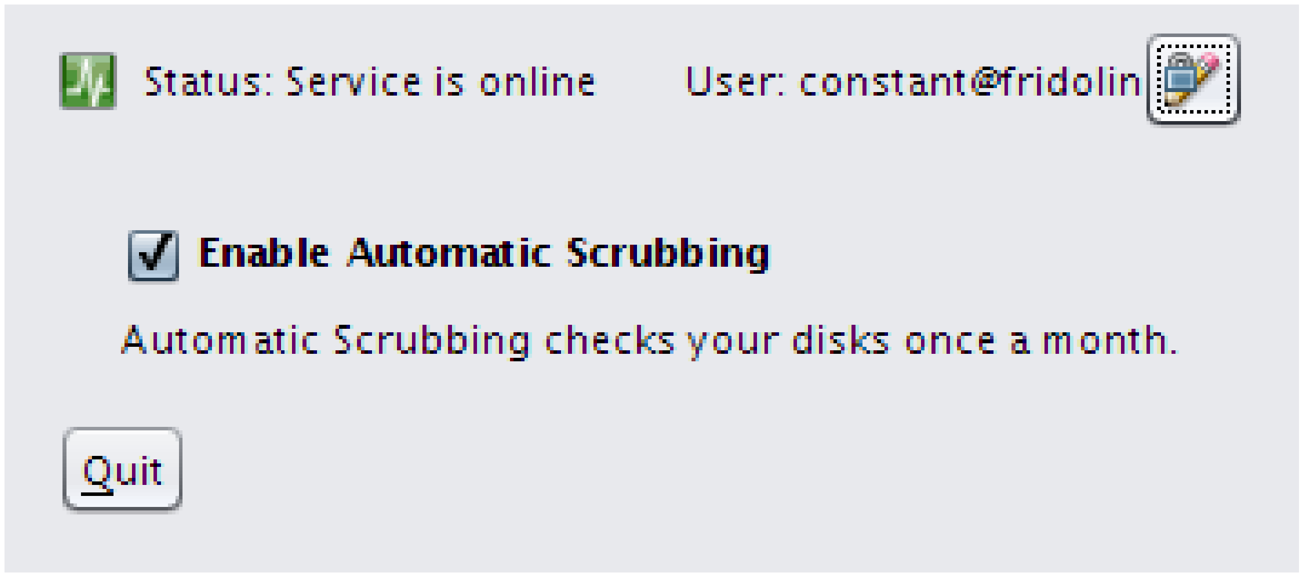
Visual Panel Components

- Java-Classes in a .jar-file
  - Panel Descriptor
    - Describes the panel to the system
    - Pivot point for the panel
  - Controller
    - Connects the GUI with management-beans
  - Panel
    - Presents the actual GUI
  - Other classes (optional)
    - Depending on the complexity of your panel
- XML file, describing the panel

Our Service Therefore Needs

- An SMF service "`zfs/auto-scrub`"
  - A start/stop/cron script and a manifest for SMF
- Another SMF service "`zfsscrub-roleadd`"
  - A start/stop script for creating the `zfsscrub` user and a manifest for SMF
- Yet another SMF service "`zfs/scrub-track`"
  - A start/stop/cron script, similar to `zfs/auto-scrub`
  - A manifest for SMF
- A visual panel
  - A .jar-File with Java classes etc.
  - An XML file with a description

Done!


Status: Service is online

User: constant@fridolin

[✓] Enable Automatic Scrubbing

Automatic Scrubbing checks your disks once a month.

Lesson #6:

Little things (like GUIs) please little minds...

Future Features

- Black/White lists for scrub times
- Expand the GUI
  - Current scrub status and statistics
  - User-defined instances
    - Pool specific
    - With different scrubbing intervals
  - Simple/complex view
- Store scrub preferences in ZFS Properties instead of SMF properties
  - Will travel with the pool
- Publish as IPKG through a repository

Lessons Learned

- SMF ist easy to program, if you ~~steal~~ re-use from examples.
  - `/lib/svc/method`
  - `svccfg export <service>`
- When in doubt, use SMF
- Easy ideas can become surprisingly complex, if you try to implement them right.
  - But you learn a lot about the rest of the system.
- GUIs with Visual Panels are still kinda wonky, but they seem to work.

Links

- Tim Foster's ZFS Auto-Snapshot Service
  - `http://blogs.sun.com/timf`
- SMF
  - `man smf`
  - `http://opensolaris.org/os/community/smf/`
- Visual Panels
  - `http://opensolaris.org/os/project/vpanels/`
- Download from my Blog
  - `http://blogs.sun.com/constantin`

opensolaris™

THANK YOU!

Constantin Gonzalez
constantin@sun.com
blogs.sun.com/constantin
twitter.com/zalez