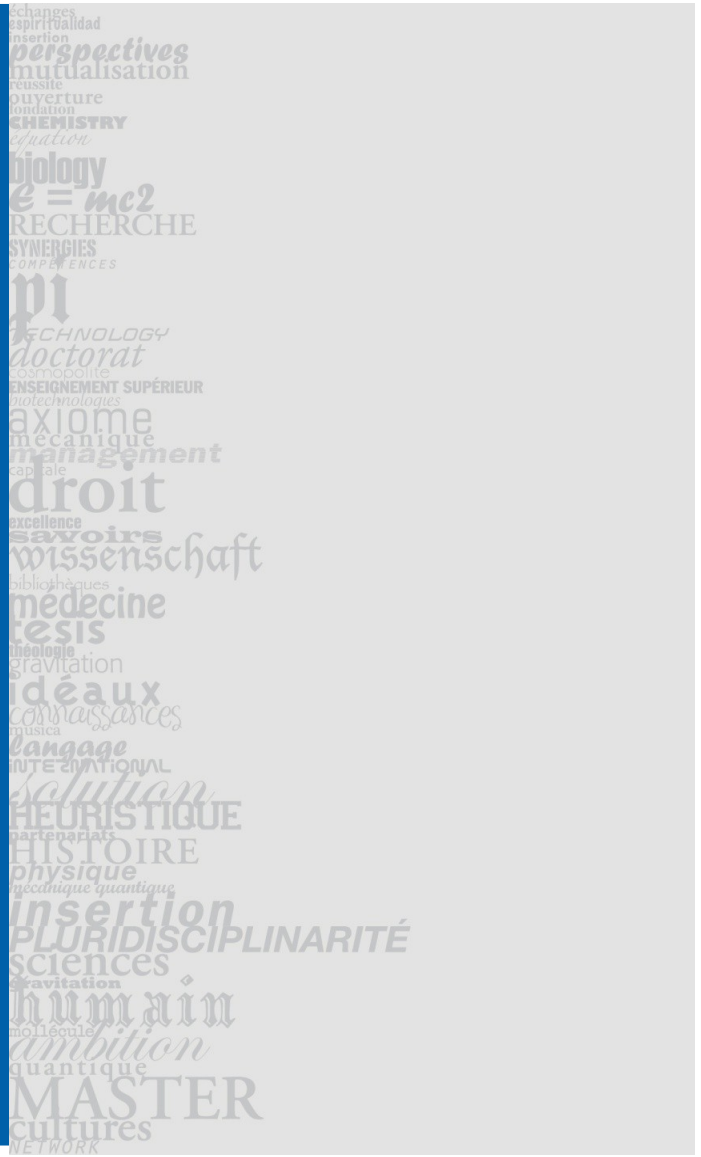
Taming network ops complexity

Jean Benoit
Université de Strasbourg
2014/04/25



UNIVERSITÉ DE STRASBOURG

Roadmap

- Approach
- ITIL and Visible Ops
- Implementation

Approach

- Goal: make your network reliable
- Combine those 3 practices
  - Change monitoring
  - Asset Inventory
  - Automated deployment
- This approach comes from
  - Putting those 3 things in practice
  - Thinking about how and why they work
- ITIL ?

Road map

- Approach
- ITIL and Visible Ops
- Implementation

ITIL Core (v3)

"ITIL manuals are like **kryptonite** to enthusiasts"

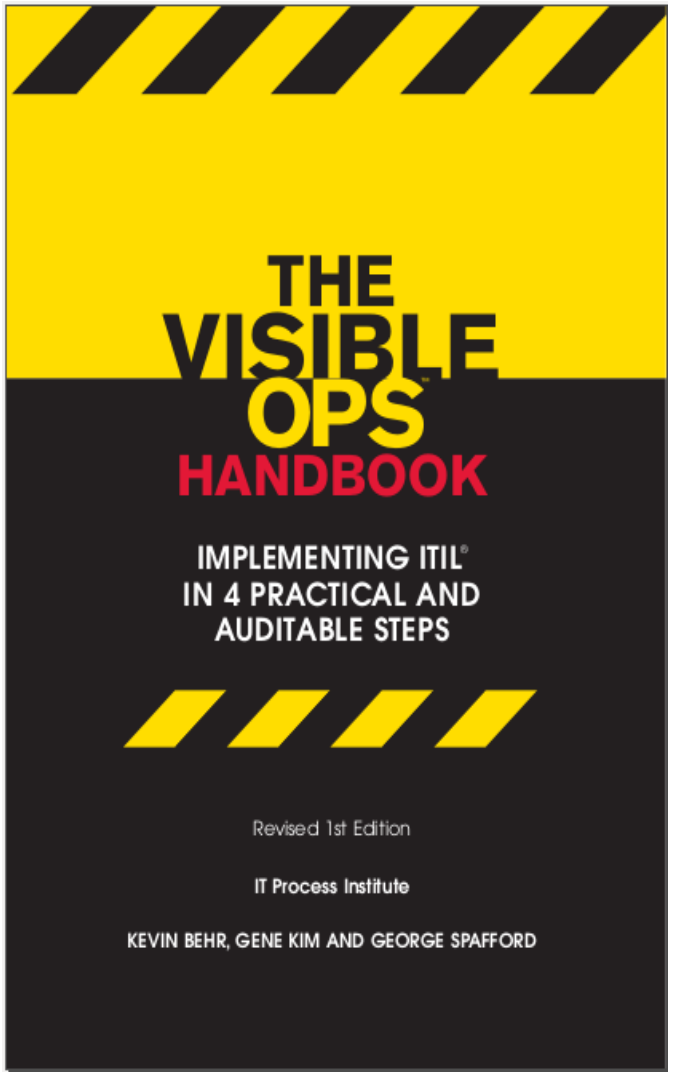
BOFH, episode 34

ITIL Core (v3)

- Set of best practices
  - 5 books, over 1500 pages in total
- Lots of interesting concepts
  - Configuration management, Change management, Capacity management etc.
- But difficult to apply
  - complex: lots of roles, processes, risk of creating silos etc.
  - ITIL is non-prescriptive ; it doesn't tell you where to start nor what is important

Another approach: The Visible Ops Handbook

- 100 pages
- Concrete examples
- 4 phases
- Origin of the book :
  - Peculiar use of Tripwire



Origin of Visible Ops : Tripwire

- Tripwire is a security tool that checks the integrity of a system
  - Computes checksum of all files
  - Compare to reference values (baseline is established when the system is installed)
  - Detect changes
- If a change is detected, Tripwire raises an alert

Tripwire → Visible Ops

```
Modified object name  /etc/passwd-

  Property:           Expected                Observed
  ------------        -----------             -----------
* Inode Number        2099877                 2101089
* Size                2714                    2772
* Modify Time         Sun Nov 18 13:32:14 2012 Thu Dec 13 20:40:28 2012
* CRC32               C2Oayv                  AdbhBT
* MD5                 B4D5f3WjCmExJ+clnPNB5u  B8wE3wV3ojnztY7Vp4XUn6


Modified object name:  /etc/shadow-

  Property:           Expected                Observed
  ------------        -----------             ----------
* Size                1762                    1887
* Modify Time         Thu Nov  1 19:23:18 2012 Sun Nov 18 13:32:22 2012
* CRC32               BHLRTD                  DsS9ba
* MD5                 Df9YtwPJbrd2TLZcajWpVY  BU9hvSUSEUj39LGpRQGmfj



-------------------------------------------------------------------------------
Rule Name: Security Control (/etc/passwd)
Severity Level: 66
-------------------------------------------------------------------------------
  ---------------------------------------
  Modified Objects: 1
  ---------------------------------------

Modified object name:  /etc/passwd

  Property:           Expected                Observed
  ------------        -----------             -----------
* Inode Number        2101057                 2101094
```

Tripwire → Visible Ops

- Gene Kim, author of Tripwire wonders why:
  - Tripwire is not used as a security tool by some companies...
  - … but is used to increase IT services reliability
- When he compared different companies in this area, he noticed that some companies are more efficient than others
- The most efficient ones have specific practices
  - Change detection, Asset inventory, Automated deployment
- This lead him to the writing of " Visible Ops"

The 4 phases of Visible Ops

1. Stabilize the patient
2. Inventory / catch fragile artifacts
3. Build a library of repeatable builds
4. Continual improvement

Phase 1 : stabilize the patient

Phase 1 : stabilize the patient

- List critical systems generating the greatest amount of unplanned work
- Protect them against uncontrolled changes
  - New policy : no changes on those systems without approval
  - Publish this new policy

Phase 1 : stabilize the patient

- "Trust but verify"
  - Sysadmin and netadmin can apply changes
  - Changes are monitored
- Changes are visible
  - Monitoring and detecting changes = **catalyst to understand and resolve incidents**

    → **Analyzing** the change logs often leads to find the root cause of the incident
- Maintenance window

Phase 2 : “catch & release”

Phase 2 : “catch & release”

- Rangers in national parks catch all the animals, weigh them, tag them, and release them
- Do the same with servers, network equipments, applications, etc.
- Questions : what is it used for? What happens if it crashes? Is it backed up? Dependencies? Etc.

  → Detailed asset inventory

Phase 3 : library of repeatable builds

Phase 3 : library of repeatable builds

- Rebuilding is simpler than repairing
  - Automate deployment
  - “Bare-metal build”
- Deployment: assembling standard components
  - Web server: OS component + Apache component
  - DB server: OS component + Mysql component
- Deployment = code
  - Factorization, Versioning, Tests etc.

Phase 4 : continual improvement

- Measure
- Extend the perimeter

Road map

- Approach
- ITIL and Visible Ops
- Implementation

How to apply this on a campus network?

- In this context, ops teams are small ; a more pragmatic approach is needed
- Change monitoring with change events broadcasted to sysadmin/netadmin
- Federation of inventory systems
- Targeted deployment automation

Change monitoring sources

- Network change monitoring: use RANCID
- Use GIT for server change monitoring:

```
cd /etc ; git init ; git add . ; git commit -m 'premier commit'
```

- Every minute in cron:

```
(cd /etc; git status; git diff) | keepstate | mail sysadmins
```

Change monitoring destinations

- When a change is detected, it is sent
  - To the sysadmins or netadmins interested in receiving it
  - To Nagios
- In Nagios, a service named “change” is defined for every hosts:
  - Passive check
  - Volatile service
  - Output contains URL of change diff

Effects of change

- “Great, I can see what others are changing!”
- “Damn, others can see what I am changing!”
- Rounds of beer/croissants
  - If an unplanned change failed
  - Payed for by the person who did it

- This leads everybody to announce and prepare changes
  - Maintenance window
  - Greater confidence
- **Cultivate autonomy and expertise**

Federating inventory systems

- There will always be **several referential data sources**
  - Network inventory
  - Server inventory
  - Other data : network prefix, contacts
- Is it possible to centralize all this data (CMDB) ?
  - Complex and costly, difficult to keep up to date
- Be minimalist: what is the **least** amount of referential data needed?

Federating inventory systems

- Required functions : enumerate inventory items, tag items
- Use existing tools and all available sources
  - GLPI, configuration management software, RANCID, etc.
  - a naming scheme in the DNS (campus-west-core-sw01 etc.)
- Those tools make up your **Infrastructure Information System**
- Interactions between tools : loose coupling, web services

Targeted automated deployment

- Targets
  - Critical servers, clusters of servers
  - Network equipments
- Write deployment script/recipes
  - Takes more time but has a quick ROI
  - Result is repeatable and testable
  - What was done by hand before is now documented in code
  - Traceability of configuration changes with versioning

Automated deployment platform

- Use an existing configuration management tool:
  - Chef, Puppet, Ansible etc.
- Manageable server resources : file, template, package...
- Recipe = function
  - reusable

Network automated deployment platform

- 2 approaches
  - Full configuration: network model pushed on equipments
    - Need out-of-band management
  - Partial configuration: modify non-critical part of the configuration
- Puppet, Chef, Ansible... can be used on Juniper
  - Netconf
- Netmagis
- Manageable network resource : port, vlan ...
- Unstructured: custom scripts + Rancid to send commands

Automated deployment recommendations

- Extend automated deployment gradually
- Peer-programming of recipes/scripts
- Write infrastructure tests
  - Test frameworks (Cucumber for example)
- Configure it everywhere even if the device is not deployed automatically
  - Deploy the agent on all servers:
    - Manage configurations for ssh, syslog, monitoring etc.

Associating the 3 practices

- The 3 practices are often associated with each other
- Asset inventory is the foundation
  - many automations are based on it
- Inventory can provide a list of servers and network equipment filtered by a criteria (location, type etc.)
  - To target automated deployment
  - Test if change detection software is deployed
- Deployment can update inventory with data
  - Redeploy app with a new database server name
  - Creates an up to date dependency in the inventory
- Change monitoring detects configuration drift
  - Integrate configuration diff into deployment recipe

Associating the 3 practices

- Change monitoring and inventory facilitate incident resolution
  - Explore change logs close to incident location
  - Expand investigation in concentric circles using inventory
- Change monitoring and automated deployment make restoring the service quicker and easier
  - You know exactly what has changed
  - Re-deploy the elements that have been modified

Associating the 3 practices

- Network equipment life-cycle management
- Inventory is key
- Initial deployment is scripted with configuration templates
- Populates the monitoring system automatically
- See Campus Best Practice document

  Referential Data and Network Management Automation

Conclusion

- Understand the concepts
  - Read ITIL, Visible Ops etc.
- Confront concepts and reality = practice
- Maintain your infrastructure Information System
  - Foundation
- Monitor changes
  - To understand what happened
- Automate deployment
  - To stabilize infrastructure