Advanced HP-UX Patch Topics

Bob Campbell
Engineer
Hewlett-Packard

HP WORLD 2003
Solutions and Technology Conference & Expo

Todays Topics

- Depots
  - Depot overview
  - Tips
- Patch Dependencies
  - Dependency types and enforcement
  - Special Installation Instructions
- Committed Patches
  - What are they?
  - Benefits and drawbacks of patch commitment
  - Removing a committed patch

Tape Depots

While the use of tape media has dwindled, the format is still in use today

- Designed for use with serial media
- Cannot be registered for remote access
- Is easily transferred via ftp, web pages, or email

```
# ls -l PHNE_28476*
-rw-r--r-- 1 root sys 2938 Mar 24 1:06 PHNE_28476.depot
```

- Created using the **`swpackage(1m)`** command
- Built in the **`tar(1)`** format using **`pax(1)`**

Directory Depots

Generally the most useful format, directory depots are optimized for SD-UX

- Designed for random/multiple access
- Can be registered to allow network access
- Exists as a directory hierarchy (exploded tape depot)

```
# ls -l PHNE_28476*
  PHNE_28476:
  total 16
  dr-x------ 8 root  sys     1024 Jul 12 17:36 PHNE_28476
  dr-x------ 4 root  sys     1024 Jul 12 17:36 catalog
  -rw-r--r-- 1 root  sys     6020 Jul 12 17:36 swagent.log
```

- Files contained under ***Product/Fileset/Path***, IPD information & scripts found under **catalog/*Product***

Depot Registration

Two methods to register

```
   swcopy -x register_new_depot=true …
   swreg -l depot /MyDepot
```

Two methods to unregister

```
   swremove \* @ /BadDepot
   swreg -u -l depot /BadDepot
```

Fun Facts

- Registration not required for access from local host
- Registration not required for remote swlist of depot contents (network and tape format depots!)
- Registration does not require depot to exist

SD-UX Access Control Lists

Everything you ever wanted to control …

- Can control access to depots or selected contents within a depot
- Can control access from selected systems or users
- A whole new command to learn (**`swacl(1m)`**)

But not enough time today…

- Interworks 2001 proceedings contain an excellent paper “Understanding SD-UX ACLs” by Al Miller.
- Available online (Interex membership required) from http://www.interex.org/conference/iworks2001/proceedings/home.html

Patch Depot Tips

Keep them complete

- Design for a single install session
  - can include multiple bundles, superseded patches OK
  - do not worry about duplicated content
- Always include dependencies, but can choose to use the original (low-water mark) patches
- Installation should use a matching operation

```
swinstall -x patch_match_target=true …
```

Patch Depot Tips

Keep them for specific purposes

- Limit risk by limiting change
  - Patches you don't want are risks you don't need
  - Mega-depots may use different dependencies each time
- Faith in depot limits analysis time
  - Matching operations give consistent, valid results
- Smaller depots are quicker depots
  - SD-UX overhead
  - Selections made at depot creation

Patch Depot Tips

Keep them compressed

- Disk is cheap, but not free
  - GOLDQPK11i stats, 760 Mb versus 286 Mb
  - 62% compression
- Better performance
  - Less network traffic/faster transfers
  - Less disk I/O on server
  - Does require additional CPU on target systems
- Compression must be done at depot creation

```
    swcopy -s /Fat -x compress_files=true \* @ /Thin
```

- Cannot be done when copying from a tape-style depot

Patch Depot Tips

Keep them centrally managed

- Allow shared cost for management and infrastructure
  - Patch selection expertise and warning assessments
  - Dedicated (and fast) networks and servers
  - Testing, testing, testing
- Leverage knowledge across teams
  - Consistent patch levels means everyone is testing the same environment
  - Reactive patching depot contains fixes for all problems seen company-wide or just with specific applications

Patch Dependencies

In theory, releases are big and patches are small

- The scope of a patch can vary in size
  - a single library object
  - a single file
  - a fileset
  - complete product.
- Dependencies can be used to fight patch growth
  - without dependencies, certain changes would require multiple patch streams to combine (bubbling)
  - leads to unsupportable mega-patches
- Trade off between granularity and number of patches

Patch Dependencies

Working with patch dependencies takes some effort

- Tool improvements have simplified the dependency world for both patch selection and installation
  - Patch Database recommends any patch dependencies
  - Switch from recalls to warnings
- Certain aspects and releases remain a manual operation

Enforced Patch Dependencies (HP-UX 11.11 & later releases only)

SD-UX can now enforce patch dependencies

- Could not be used in earlier releases due to inability to handle the supersession of the dependency
- Dependencies are recorded between patches
- Requisites are used between specific filesets
- Two known issues, both fixed in current SD-UX patch
  - **`swconfig(1m)`** errors with superseded dependency
  - explicit selection of multiple patch bundles might exclude a patch while including all of its dependencies

Manual Patch Dependencies

Why do they exist

- Patch supports multiple releases including those that predate SD-UX support for patch requisites
  - HP-UX 11.0
  - Core-OS patches excluded from this set
- A conditional dependency exists
  - Documented as "Other Dependencies" (ODep)
  - Patch must function without ODep
  - Often involves optional functionality

Manual Patch Dependencies

Marked for identification

- patches created with the **`manual_dependencies`** category tag
- Can be found using **`swlist(1)`**

```
swlist -d *,c=manual_dependencies @ /MyDepot
```

Special Installation Instructions

Special Installation Instructions are intended to document operator actions required to properly install certain patches

- Have been mixed with Other Dependencies (Odeps) in the past
- Are being screened by the Patch Clearinghouse
  - moving to automated scripts when possible
  - clarifying distinction between S.I.I. and ODeps
  - will work to improve this area in the future
- The following script is offered until we can do better

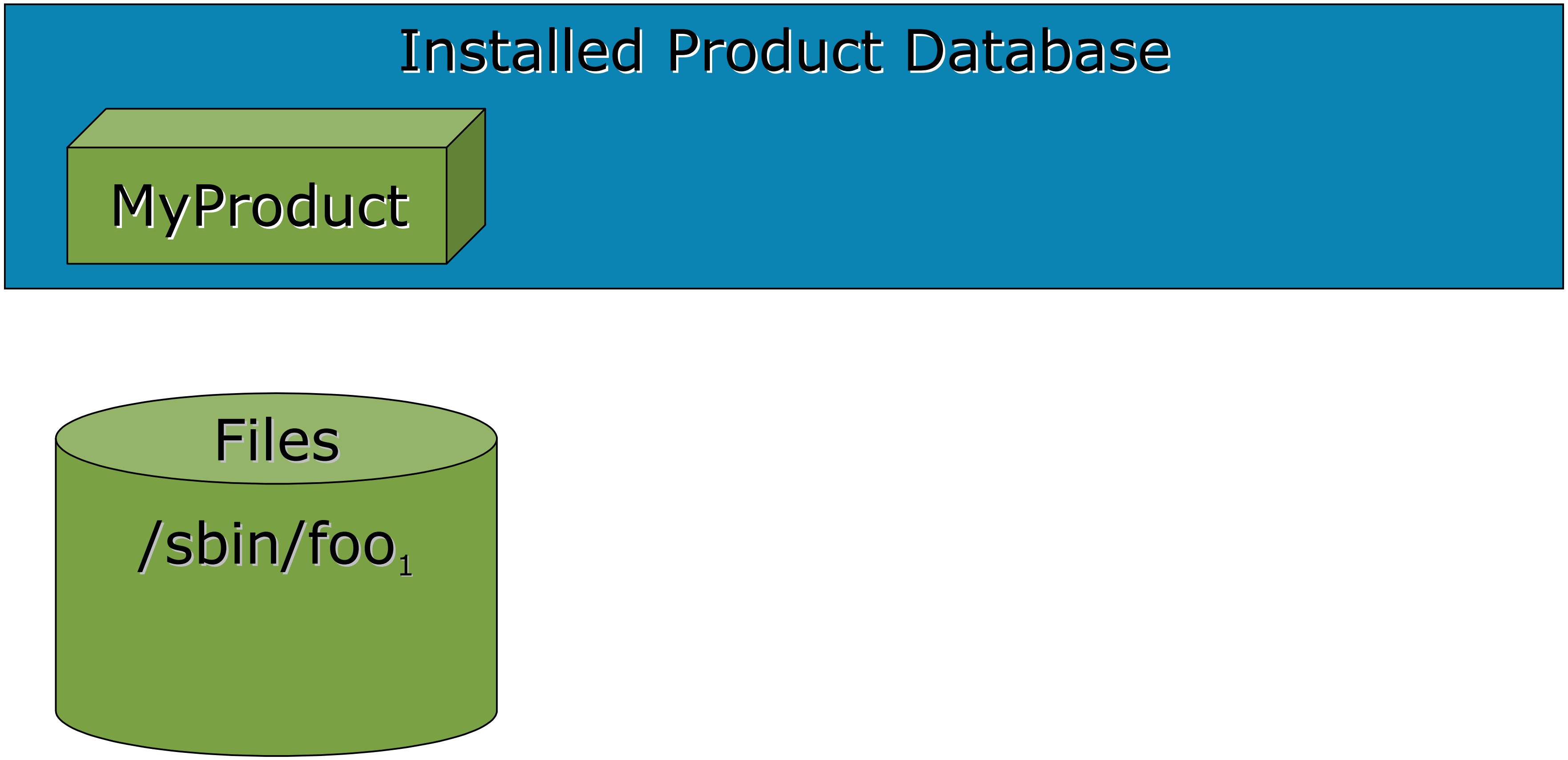
listSII.sh

This script will list the Special Installation Instructions found within a local HP-UX 11.x depot (root user req'd):

```
#! /sbin/sh
DEPOT=${1#*:}
for X in ${DEPOT}/catalog/*/pfiles/README
do
  grep -sq "^Special Installation Instructions: None" $X
  if [[ $? -ne 0 ]]
  then
    Y=${X#${DEPOT}/catalog/}
    print ${Y%/pfiles/README}
    sed -e "1,/^Special Installation Instructions:/d" $X
  fi
done
```

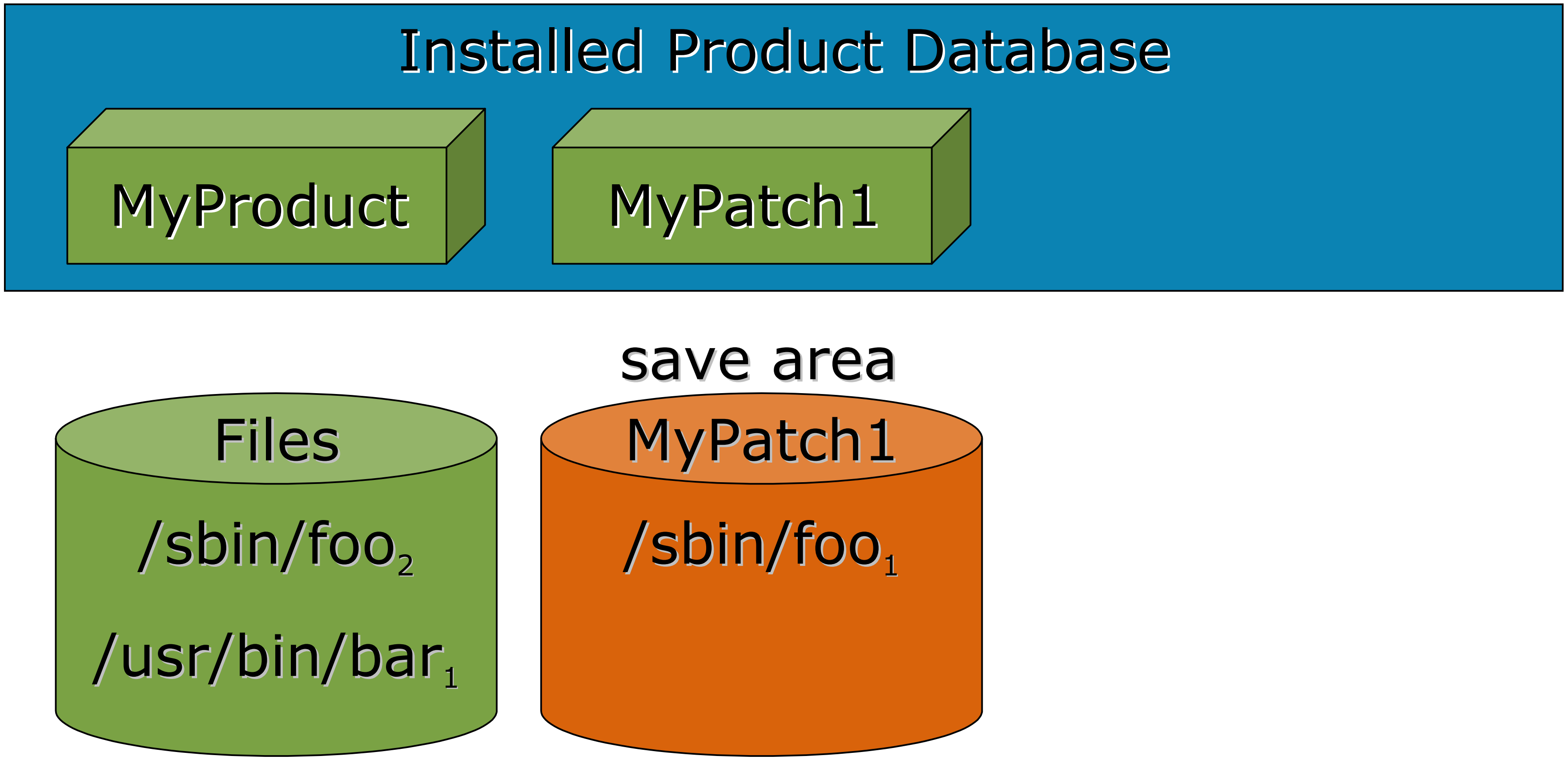
Patch Commitment

MyProduct delivers the first version of the command foo, with MyProduct recorded in the IPD



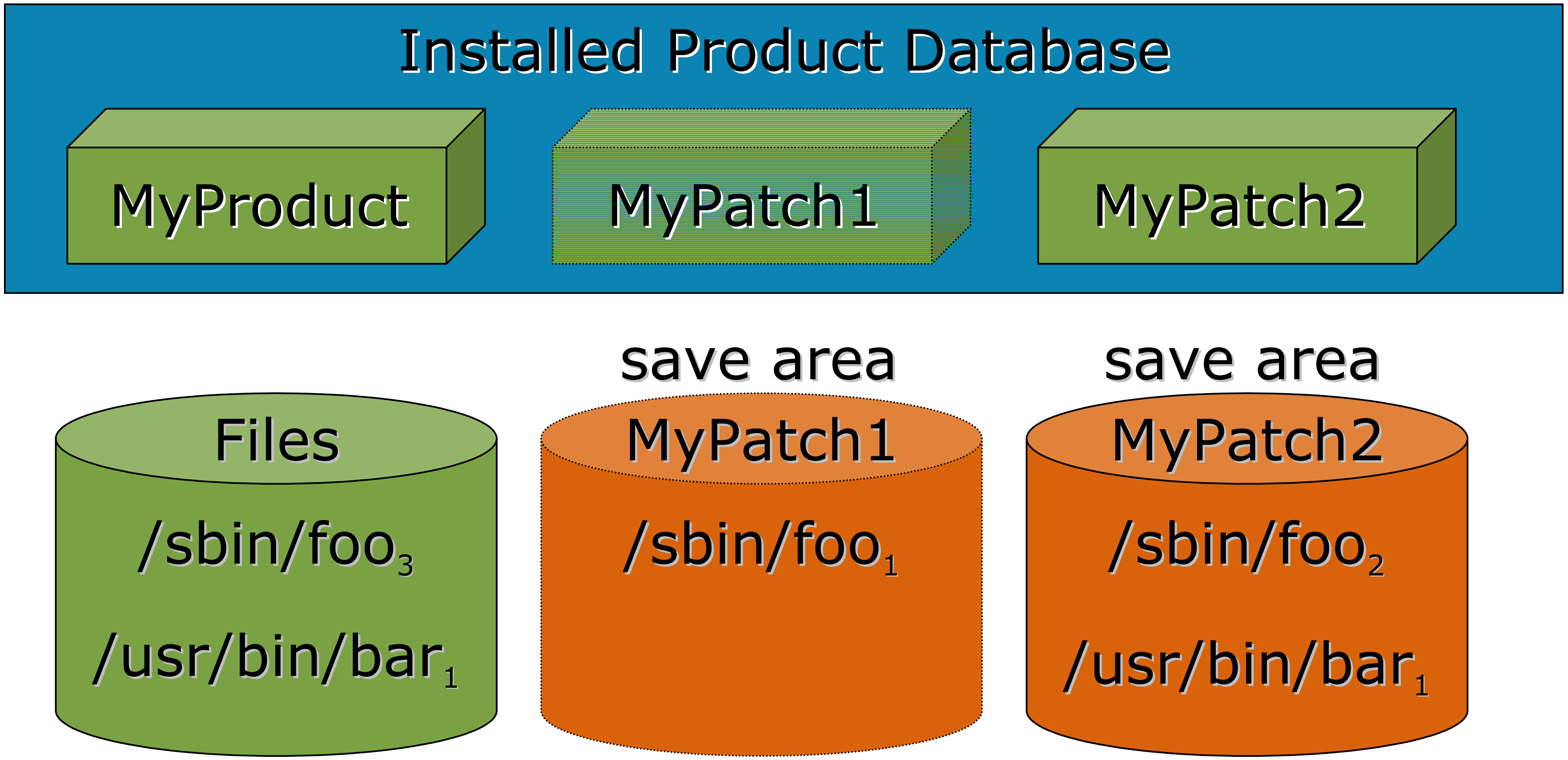
Patch Commitment

MyPatch1 delivers a new version of foo, and a new command bar. The previous version of foo is placed in the save area associated with MyPatch1



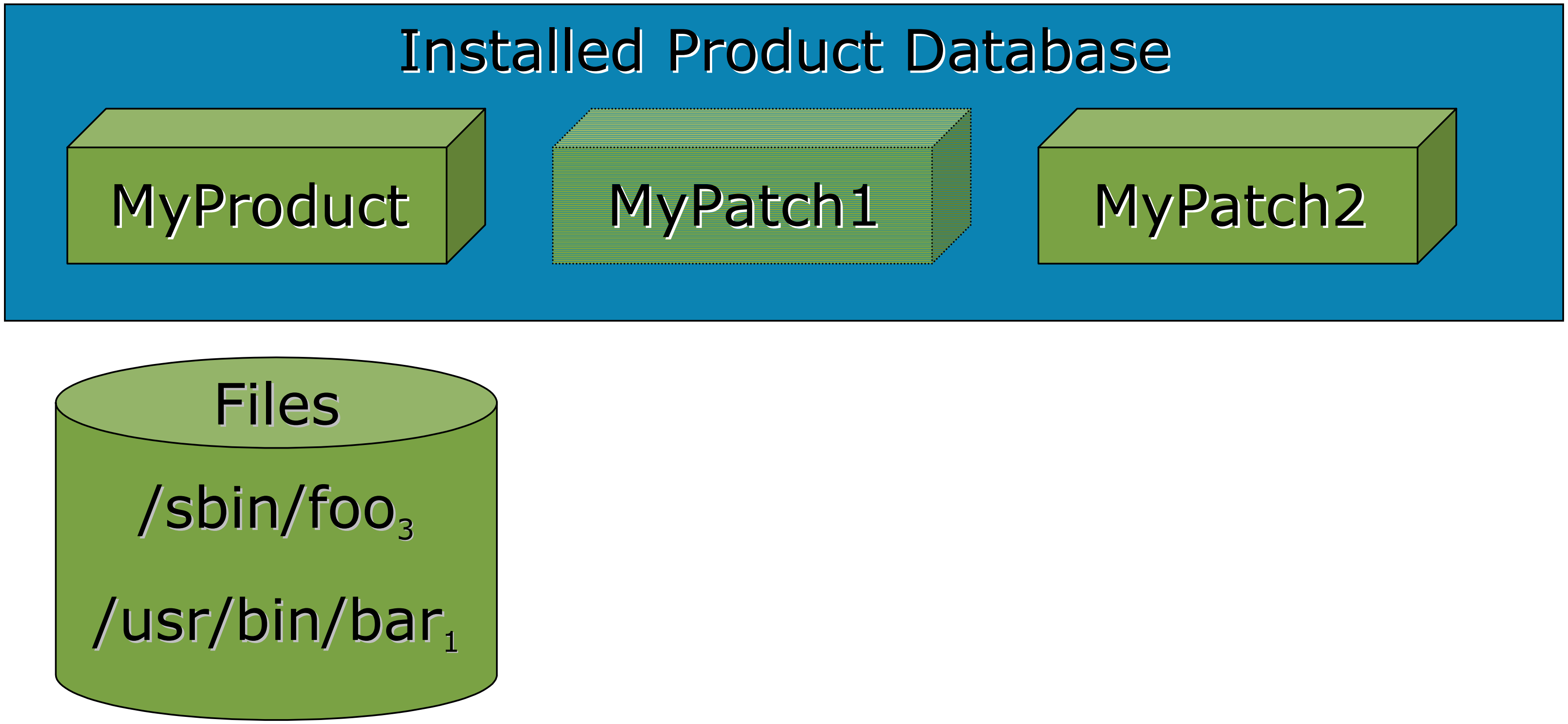
Patch Commitment

MyPatch2 delivers a 3rd version of foo, and redelivers bar. MyPatch1 is in the IPD, but is hidden from view.



Patch Commitment

When the patches are committed, they remain recorded in the IPD (with attributes changed) but the save areas are deleted. Patch removal is no longer possible…



Reasons to Commit Patches

Disk Space Recovery

Action	Size of `/var` (Mbytes)	Size Change (Mbytes)
Base System	43.5	N/A
GOLDQPK11i June 2001	82.3	38.8
GOLDQPK11i June 2002	271.9	189.6
GOLDQPK11i June 2003	625.4	353.5
Commit all patches	50.2	-575.2

Reasons to Commit Patches

Enforcing minimum patch levels (low-water marks)

- Hardware enablement patches
  - Don’t remove, halt the system!
- Security Bulletins
  - Got lawyers???
- Critical Defects
  - Known problems seen in your environment
  - Severe issues you want to avoid at all costs

Reasons to Commit Patches

Installation/Recovery Performance

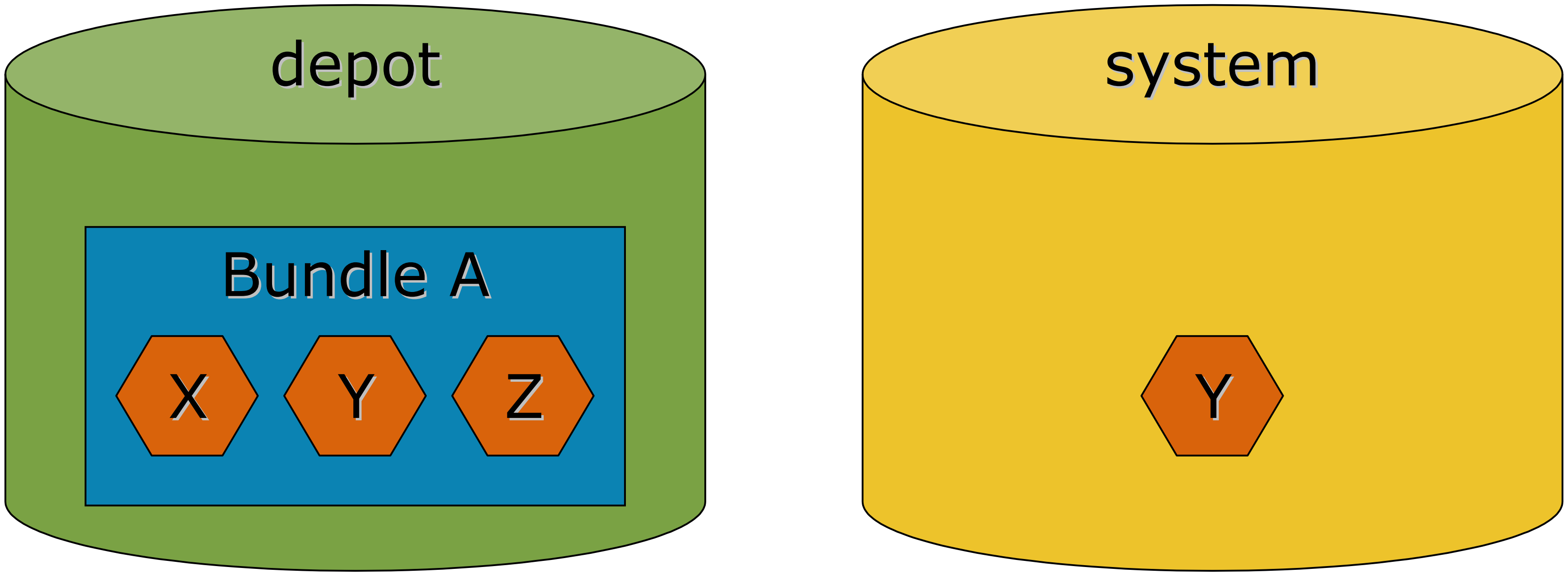
- Speed of **`swinstall(1m)`**
  - Using "**`-x patch_save_files=false`**"
  - Less disk I/O
  - ~10% speed improvement
- Ignite-UX
  - Less network traffic
  - Savings on image load or creation
  - recovery image file ~25% smaller
  - ~10% speed improvement

Stats determined using system from disk space recovery example. Your mileage will vary!!!

Reasons to Commit Patches

Support for Bundle Rollback

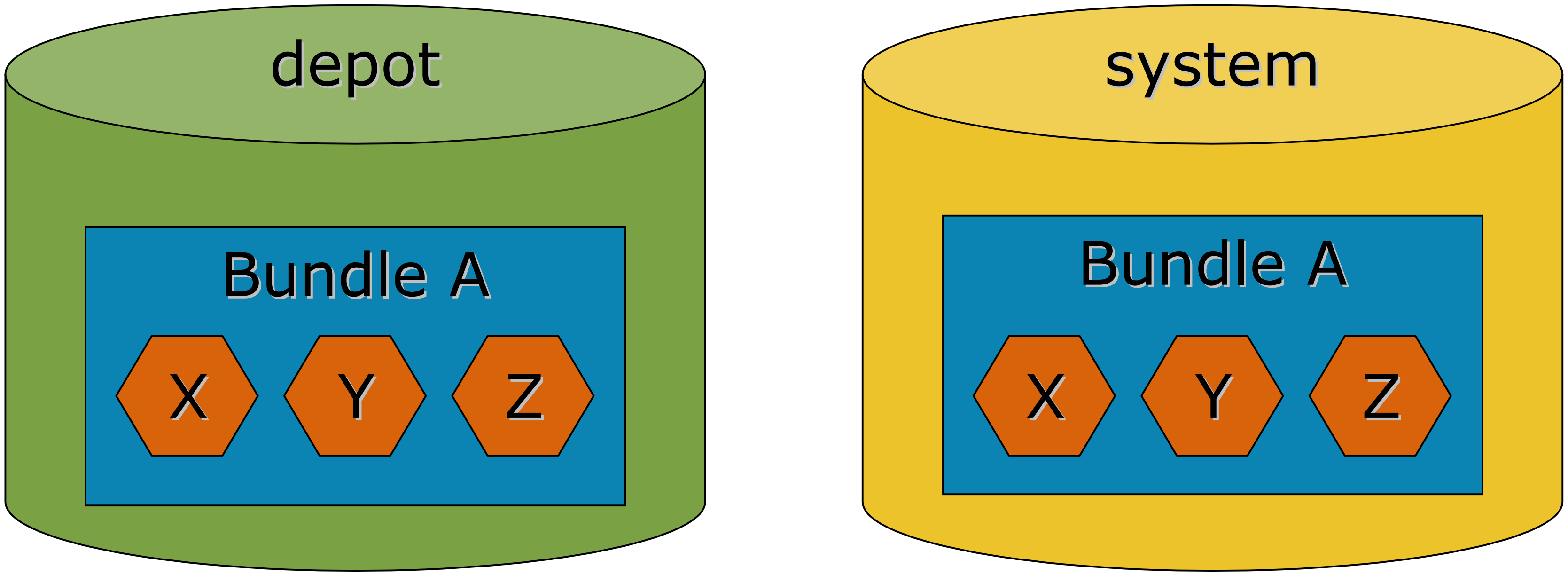
- Bundle A contains patches X, Y, and Z
- Patch Y is already installed on system



Reasons to Commit Patches

Support for Bundle Rollback

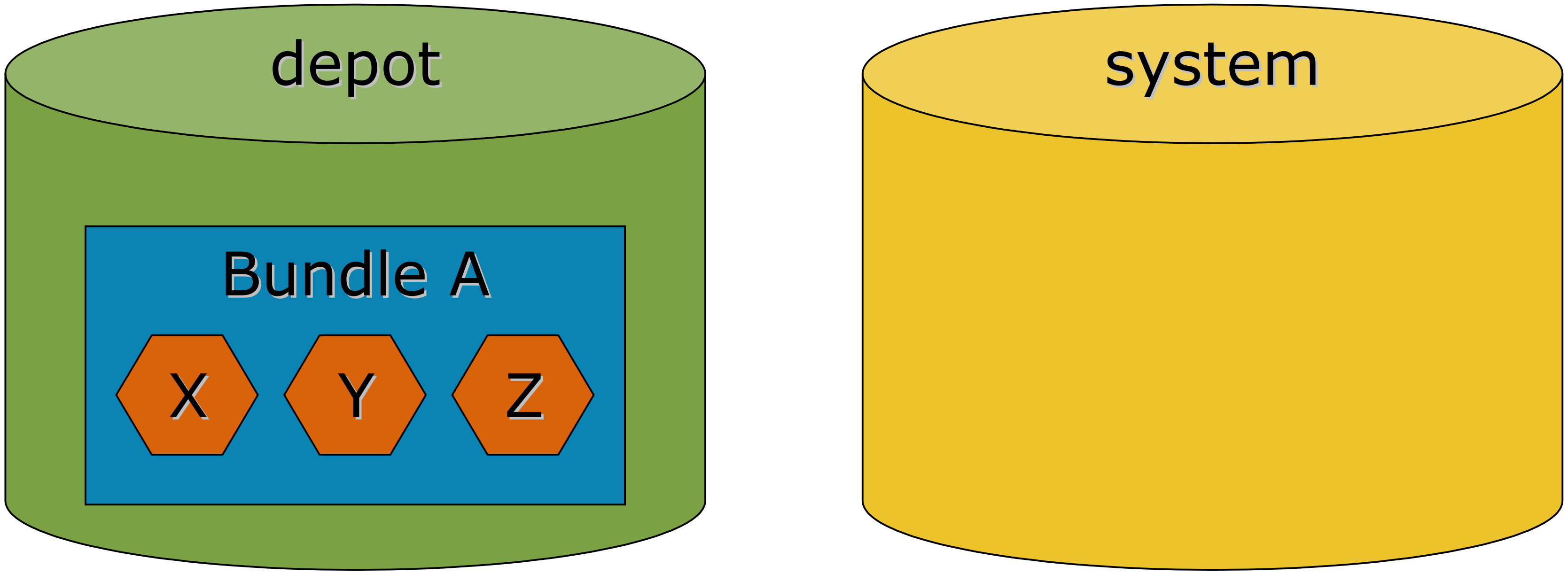
- When Bundle A is installed, it inherits the preexisting Y



Reasons to Commit Patches

Support for Bundle Rollback

- Removal of Bundle A does not return us to original state
- If Y is contained in other bundles, it should be anchored and would not be removed with the rest of Bundle A



Reasons to NOT Commit

There are lots of reasons to commit a patch, but really just one reason not to

- When you can't take it off, you may suddenly have to
  - *Patch warnings*
  - *Security bulletins*
  - *Management told you to*

Bad days happen……

Committed Patch Removal

I have five methods for committed patch removal

- The *Short* Answer
- The *Somewhat Longer* Answer
- The *If I Had A Time Machine* Answer
- The *Brute Force* Answer
- The *Brain Surgery* Answer

Committed Patch Removal
The *Short* Answer

Don't

you saw that one coming, right???

Committed Patch Removal
The *Somewhat Longer* Answer

All mechanisms for removing committed patches are untested and may result in new and exciting problems, try other methods to resolve the issue

- Roll forward, not back
  - Compare current issue with joy of rollback
  - Tools and processes are set up to install, not remove
  - Can commit new patch to avoid future exposure
- Consider non-SD temporary solutions
  - Make manual changes until patch available
- Reinstallation allows you to fix all of the mistakes made the first time around….

Committed Patch Removal
The *If I Had A Time Machine* Answer

There are things you can do before committing a patch that can save you later on

- Ignite-UX recovery images
  - the big red reset button
  - deterministic load time
  - can preserve multiple checkpoints
  - can create new root to preserve current state
- Create backups of Save Area and IPD before commit
  - Beware old backups!
  - /var/adm/sw directory holds both
  - If you have the backup, support knows what to do
  - More info under *Brain Surgery*

Committed Patch Removal
The *Brute Force* Answer

Forced reinstallation is possible using override options within **`swinstall(1m)`** *(bigger hammers)*

- Not Cold Install
  - Existing files systems retained
  - "Extra" files not removed
- Reinstallation of the patch's ancestor product
  - Native patching support automatically cleans IPD
  - Superseded patch can install with base product
- Can be done on a system-wide scale or surgically
  - Reloading all of core system safest (my opinion)
  - Reloading specific areas saves time, but takes work

Committed Patch Removal
The *Brute Force* Answer (continued)

This method requires a depot that contains all of the products and patches needed to cold install

```
swinstall -s /MyDepot \
          -x auto_reboot=true \
          -x allow_downdate=true \
          -x reinstall=true \
          -x reinstall_files=true \
          HPUXBase64 HPUXBaseAUX HPUX11i-OE-MC
```

Any patch left in a committed state should be due to the ancestor product not installing

Committed Patch Removal
The *Brute Force* Answer (continued)

The same method can be done for specific products or even filesets, but risks increase with granularity!

- Can significantly cut install time
- Identify patch ancestors using **`swlist(1m)`**
- May require a reverse dependency analysis

```
# swlist -l fileset -a applied_patches | grep PHCO_24846
# PHCO_24846
 PHCO_24846.SECURITY
 SecurityMon.SECURITY PHCO_24504.SECURITY PHCO_24846.SECURITY
```

(this example edited to fit, SD-UX really likes attributes)

Committed Patch Removal
The *Brute Force* Answer (continued)

If you are attempting to install only selected filesets additional options may be needed to limit bloat

```
   swinstall -s /MyDepot \
             -x auto_reboot=true \
             -x allow_downdate=true \
             -x reinstall=true \
             -x reinstall_files=true \
             -x autoselect_dependencies=false \
             SecurityMon
```

The more of SD-UX disabled, the greater the risk and the preparation that may be needed!

Committed Patch Removal
The *Brain Surgery* Answer

So you actually want to understand what is going on? The Installed Product Database is the key

- **`/var/adm/sw/products/INDEX`**
  - created from product and fileset INDEX files located in subdirectories
  - regenerated each time software is installed or removed (preview mode doesn’t count)
  - If you only restore this file, the component INDEX files will rebuild the file you didn’t want!

Committed Patch Removal
The *Brain Surgery* Answer

You have restored the IPD subdirectory for the patch, but without the save area you won't go far

- **`/var/adm/sw/save/`*`PatchID`***
  - all files preserved by SD when directly overwritten
  - saved using full path
- OK, other areas exist but we don't tell SD about them
  - **`/var/adm/sw/save_custom`**
  - **`/var/adm/sw/save-custom`**
  - **`/var/adm/sw/tmp`**

Committed Patch Removal
The *Brain Surgery* Answer

Getting really close, just one more trick…

- You have restored the IPD subdirectory for the patch and the save area, but it is still won't remove!
  - Is it superseded?
  - Did you rebuild the master INDEX?
- Dang, I don't want to install or remove anything….
  - **`swmodify(1m)`** can't help here
  - directly editing the master INDEX will work

Committed Patch Removal
The *Brain Surgery* Answer

```
product
tag PHCO_24504
data_model_revision 2.40
instance_id 1
control_directory PHCO_24504
revision 1.0
title "audisp patch for IPv6 and unix sockets"
description "Patch PHCO_24504: audisp IPv6 & sockets patch"
mod_time 1058072261
create_time 1057861856
… SNIP
fileset
tag SECURITY
data_model_revision 2.40
… SNIP
state configured
ancestor SecurityMon.SECURITY,fr=B.11.11,v=HP
is_sparse true
patch_state committed
applied_to SecurityMon.SECURITY,l=/,r=B.11.11,a=HP-
  UX_B.11.11_32/64,v=HP,fr=B.11.11,fa=HP-UX_B.11.11_32/64
end
```

Committed Patch Removal
The *Brain Surgery* Answer

Can I “forget” a patch by deleting the IPD directory and master entries?

No, but nice try. More surgery needed for that!

- Ancestor filesets **`applied_patches`** attribute
  - records all installed patches modifying each fileset
- Superseded patches **`superseded_by`** attribute
  - records installation order
  - needs to be replaced with entry for forgotten patch
- Superseded patches **`patch_state`** attribute
  - is patch still superseded?

This is not an area even I have played in, beware!

Closing thoughts

- Recovery images of a bad state may prove valuable when your experiments find a worse state
- Like strong medicine, these tricks should only be used when needed
- A little knowledge is a dangerous thing
- If you get into trouble, my name is Scott McNealy
- Most important of all, Hewlett-Packard disavows any knowledge of me or my team, these slides will self-destruct in 5 seconds

HP WORLD 2003
Solutions and Technology Conference & Expo

Interex, Encompass and HP bring you a powerful new HP World.

interex
shared knowledge • shared power

encompass
AN HP USER GROUP

hp
invent