



# HomeLinks Roaming Profiles in Math MacOSX Labs

**Jeff Kopmanis**

Manager, MathIT

MacSIG - December 1, 2004

# A Little History...

- Previously, all Solaris 8 Labs (103 machines)
- LSA AFS Home Directories
- Fully managed
- Access only for Math Course Students
- Maple, MATLAB, Netscape 4.78, etc...

# Solaris 8 Problems

- GUI (Sun CDE) was unfamiliar and hard for students to use

# Solaris 8 Problems

- GUI (Sun CDE) was unfamiliar and hard for students to use
- Printing through lp/lpr was difficult

# Solaris 8 Problems

- GUI (Sun CDE) was unfamiliar and hard for students to use
- Printing through lp/lpr was difficult
- StarOffice 6 was not 100% compatible with Microsoft Office

# Solaris 8 Problems

- GUI (Sun CDE) was unfamiliar and hard for students to use
- Printing through lp/lpr was difficult
- StarOffice 6 was not 100% compatible with Microsoft Office
- Sparse numbers of applications and application availability for 64-bit Solaris 8

# Solaris 8 Problems

**SLOW**

# Spring 2003

- LSA Instructional Technology Grant (Mar/03)



# Spring 2003

- LSA Instructional Technology Grant (Mar/03)
- UNIX-centric: synctree? RsyncX? Radmind?

# Spring 2003

- LSA Instructional Technology Grant (Mar/03)

**BZZT!** UNIX-centric: synctree? RsyncX? Radmind?

# Spring 2003

- LSA Instructional Technology Grant (Mar/03)
- BZZT!** ● UNIX-centric: synctree? RsyncX? Radmind?
- Standalone, CD Load

# Spring 2003

● LSA Instructional Technology Grant (Mar/03)

**BZZT!** UNIX-centric: synctree? RsyncX? Radmind?

**BZZT!** Standalone, CD Load

# Spring 2003

- LSA Instructional Technology Grant (Mar/03)

**BZZT!** UNIX-centric: synctree? RsyncX? Radmin?

**BZZT!** Standalone, CD Load

- Apple Computer: John Hickey & Interns

- Netboot + Workgroup Manager

- Radmin as fallback option

# Spring 2003

- LSA Instructional Technology Grant (Mar/03)

**BZZT!** ● UNIX-centric: synctree? RsyncX? Radmin?

**BZZT!** ● Standalone, CD Load

- Apple Computer: John Hickey & Co.

**WINNER!!!**

● Netboot + Workgroup Manager

● Radmin as fallback option

# May, 2003

- 24 iMac G4 800MHz 512MB RAM
- Xserve G4 1GHz 512MB RAM
- 100Mbps to each iMac, 1Gbps to Xserve
- First image: MacOS 10.2.3 (Jaguar) <3GB
- 2 minute or less boot time!!
- Online by mid-June!

# 2003-2004

- June-July, 2003 & 2004: Michigan Math & Science Scholars Camp ("Math Camp")
- F03, W04, F04: Math 216 - EXCLUSIVE, Math 215 - *REQUESTED*
- 3 LSA-IT Training Sessions
- 2 Apple Certification Courses
- Lots of misc. Math Dept orientations and instructional sessions



Evaluation: 1 Year Later...

A Very Successful Project

# Evaluation: 1 Year Later...

## A Very Successful Project

***Thank You***, John, Sheri  
and all of Apple Computer!

# Evaluation: 1 Year Later...

## A Very Successful Project

But, we also learned some things...

# Evaluation: 1 Year Later...

- Saving Bookmarks: highly unusual and misunderstood by students

# Evaluation: 1 Year Later...

- Saving Bookmarks: highly unusual and misunderstood by students
- A Volatile Desktop: erased on logout

# Evaluation: 1 Year Later...

- Saving Bookmarks: highly unusual and misunderstood by students
- A Volatile Desktop: erased on logout
- Image bloat: 6+ GB (still 2 minute boot times!)

# Evaluation: 1 Year Later...

- Saving Bookmarks: highly unusual and misunderstood by students
- A Volatile Desktop: erased on logout
- Image bloat: 6+ GB (still 2 minute boot times!)
- Cross-realm K5 trusts not supported in Jaguar

# Evaluation: 1 Year Later...

- Saving Bookmarks: highly unusual and misunderstood by students
- A Volatile Desktop: erased on logout
- Image bloat: 6+ GB (still 2 minute boot times!)
- Cross-realm K5 trusts not supported in Jaguar
- Jaguar getting very long in tooth, both on server and client



# Evaluation: 1 Year Later...

- 118 machines on a single G4 Xserve *doesn't* work (Apple's 100 machine limit is **TRUE**)

# Evaluation: 1 Year Later...

- 118 machines on a single G4 Xserve *doesn't* work (Apple's 100 machine limit is **TRUE**)
- Server Settings app is hopelessly broken under Jaguar

# Evaluation: 1 Year Later...

- 118 machines on a single G4 Xserve *doesn't* work (Apple's 100 machine limit is **TRUE**)
- Server Settings app is hopelessly broken under Jaguar
- 512MB RAM in Xserve is inadequate for anything over 50 netboot clients

# Evaluation: 1 Year Later...

- 118 machines on a single G4 Xserve *doesn't* work (Apple's 100 machine limit is **TRUE**)
- Server Settings app is hopelessly broken under Jaguar
- 512MB RAM in Xserve is inadequate for anything over 50 netboot clients
- **UPGRADE TIME!!!**

# Summer, 2004

- Panther Load (MacOSX 10.3.5)
- Easy to reproduce image: LSA SNI-based
- OpenAFS 1.2.10a
- Layered LDAP: UMOD & Math
- LSA LabHomeDirs.pkg (Sites-based)
- “Roaming Profiles” via **HomeLinks**

# HomeLinks

- Dynamically maps local structure onto network storage

# HomeLinks

- Dynamically maps local structure onto network storage
- Can create network locations, if not present

# HomeLinks

- Dynamically maps local structure onto network storage
- Can create network locations, if not present
- XML (plist) files used for configuration



# HomeLinks

- Dynamically maps local structure onto network storage
- Can create network locations, if not present
- XML (plist) files used for configuration
- PropertyList Editor becomes an easy-to-use configuration tool

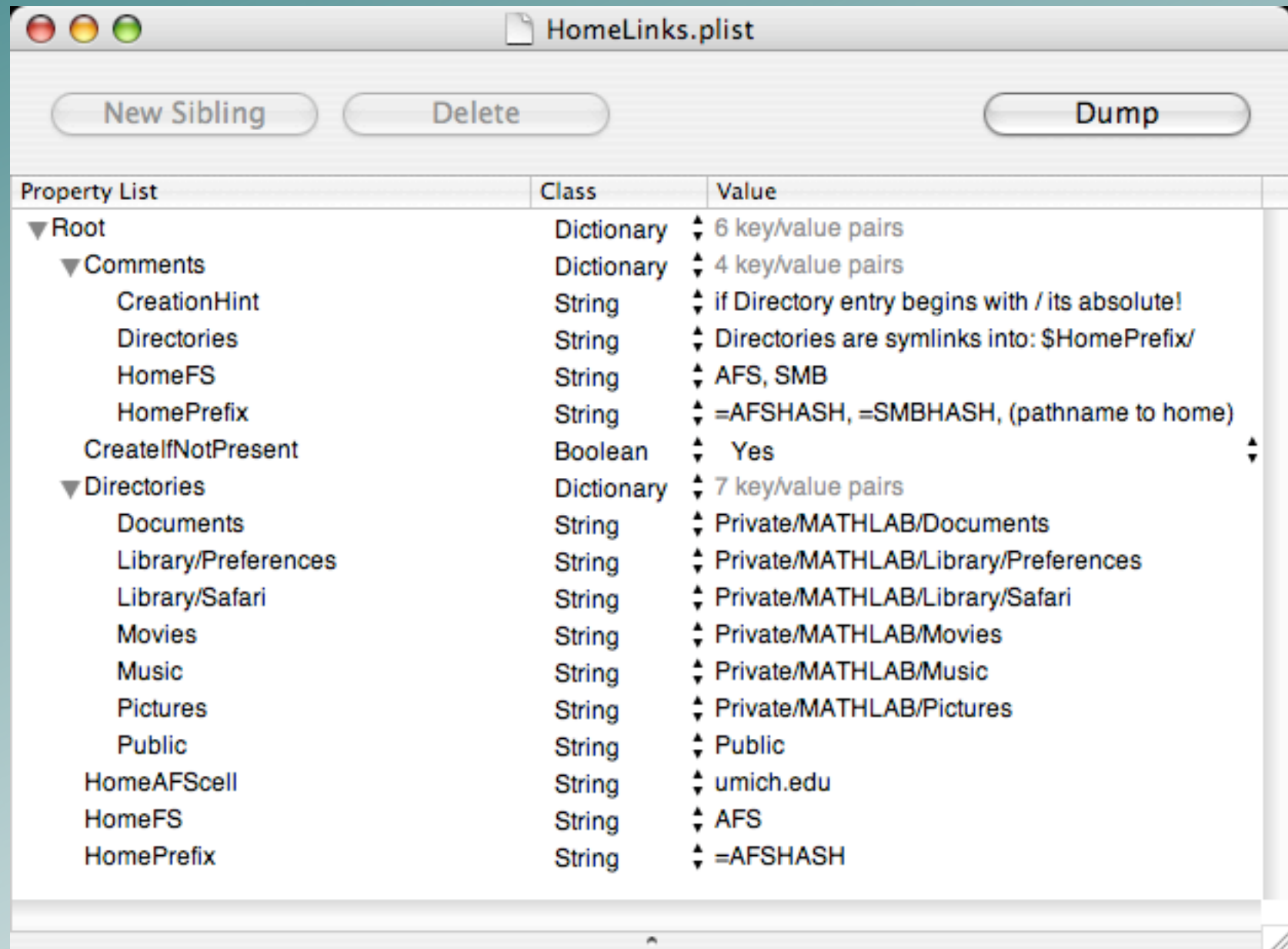
# HomeLinks

- Dynamically maps local structure onto network storage
- Can create network locations, if not present
- XML (plist) files used for configuration
- PropertyList Editor becomes an easy-to-use configuration tool
- Needs **plistbuddy**, but with some work, could probably use **defaults** instead

# HomeLinks: Overview

- Build a network path prefix (`$PREFIX`) for the user logging in
- For each of the Directories in the Dictionary...
- use **ditto** to copy from `/Users/username/key` to `$PREFIX/value`

# Sample HomeLinks.plist file



Property List	Class	Value
▼ Root	Dictionary	▲ 6 key/value pairs
▼ Comments	Dictionary	▲ 4 key/value pairs
CreationHint	String	▲ if Directory entry begins with / its absolute!
Directories	String	▲ Directories are symlinks into: \$HomePrefix/
HomeFS	String	▲ AFS, SMB
HomePrefix	String	▲ =AFSHASH, =SMBHASH, (pathname to home)
CreatelfNotPresent	Boolean	▲ Yes
▼ Directories	Dictionary	▲ 7 key/value pairs
Documents	String	▲ Private/MATHLAB/Documents
Library/Preferences	String	▲ Private/MATHLAB/Library/Preferences
Library/Safari	String	▲ Private/MATHLAB/Library/Safari
Movies	String	▲ Private/MATHLAB/Movies
Music	String	▲ Private/MATHLAB/Music
Pictures	String	▲ Private/MATHLAB/Pictures
Public	String	▲ Public
HomeAFScell	String	▲ umich.edu
HomeFS	String	▲ AFS
HomePrefix	String	▲ =AFSHASH

# Constructing a Prefix

HomeAFScell	String	↕ umich.edu
HomeFS	String	↕ AFS
HomePrefix	String	↕ =AFSHASH

- HomeFS is a flag for how to interpret things. Currently only AFS is implemented.
- HomePrefix determines what comes before the user's unickname in the path.
- =AFSHASH denotes the UM double-hash
- HomeAFScell is the cell name

# Prefix examples:

HomeAFScell	String	↕ umich.edu
HomeFS	String	↕ AFS
HomePrefix	String	↕ =AFSHASH

Constructs: `/afs/umich.edu/user/k/o/kopmanis`

HomeAFScell	String	↕ lsa.umich.edu
HomeFS	String	↕ AFS
HomePrefix	String	↕ =AFSHASH

Constructs: `/afs/lsa.umich.edu/user/k/o/kopmanis`

HomeFS	String	↕ NFS
HomePrefix	String	↕ /home/exports

Constructs: `/home/exports/kopmanis`

# Directory Dictionary

- Works with key-value pairs
- keys are the “from” or “source” location
- key maps to /Users/username/key
- values are the “to” or “destination” location
- value maps to \$PREFIX/value

# key-value examples:

▼ Directories	Dictionary	▲ 7 key/value pairs
Documents	String	▲ Private/MATHLAB/Documents
Library/Preferences	String	▲ Private/MATHLAB/Library/Preferences
Library/Safari	String	▲ Private/MATHLAB/Library/Safari
Movies	String	▲ Private/MATHLAB/Movies
Music	String	▲ Private/MATHLAB/Music
Pictures	String	▲ Private/MATHLAB/Pictures
Public	String	▲ Public

With our previous umich.edu AFS prefix:

`/Users/kopmanis/Documents`

`/afs/umich.edu/user/k/o/kopmanis/Private/MATHLAB/Documents`

`/Users/kopmanis/Public`

`/afs/umich.edu/user/k/o/kopmanis/Public`

`/Users/kopmanis/Library/Preferences`

`/afs/umich.edu/user/k/o/kopmanis/Private/MATHLAB/Library/Preferences`



# CreateIfNotPresent

- HomeLinks will build a structure according to the plist configuration file
- If “Yes”, this flag signals that directories should be created in the target location if not present
- Each directory is tested along the way
- If \$PREFIX is not present, no creation will be possible, so no mapping is performed (*Dec04*)

# HomeLinks: Overview

- Build a network path prefix (`$PREFIX`) for the user logging in
- For each of the Directories in the Dictionary...
- use **ditto** to copy from `/Users/username/key` to `$PREFIX/value`

# **Wait a Minute!!!**

**What about Kerberos tickets  
and AFS tokens and file ACLs  
and permissions ?!**

# Panther Security Sessions

- OSX processes run as root or other admin users

MacOS  
10.3.x

# Panther Security Sessions

- OSX processes run as root or other admin users
- User authenticates at login to create a Security Session, which includes AFS tokens.



MacOS  
10.3.x

AFS tokens

User

# Panther Security Sessions

- OSX processes run as root or other admin users
- User authenticates at login to create a Security Session, which includes AFS tokens.
- HomeLinks, run by MacOSX, uses **sudo** to reach into user's Security Session to use AFS tokens of that user



# Panther Security Sessions

- HomeLinks uses ITCS-Sites/LSA LoginHooks package to provide /etc/hooks structures
- LI81.HomeLinks is run at login time
- The magic sudo call is made in LI81.HomeLinks
- LI81.HomeLinks is a wrapper for the real script in /etc/HomeLinks/HomeLinks

# /etc/hooks/L181.HomeLinks

```
#!/bin/bash
```

```
sudo -u $1 /etc/hooks/HomeLinks/Homelinks $1
```



# /etc/HomeLinks/HomeLinks

```
#!/bin/bash
#####
# First arg should be user shortname
#####

export UNIONNAME=$1
export UNIQUIID=`/usr/bin/id -u $UNIONNAME`
export DEBUG=0
export CONFIGFILE="/etc/hooks/HomeLinks/HomeLinks.plist"
export PLB=/usr/bin/PlistBuddy
export PLB_ARGS="/etc/radmind.defaults"

#####
function PLB_get () {
    $PLB -c "print $1" $CONFIGFILE | grep -v "Not Exist"
}

#####
function PLB_get_Prefix () {
    user=$1
    HomePrefix=`PLB_get ":HomePrefix"`
    if [ "$HomePrefix" = "=AFSHASH" ]
    then
        echo /afs/$AFSCell/user/${user:0:1}/${user:1:1}/$user/
        if [ "$DEBUG" = "1" ]; then /usr/bin/logger -is HomePrefix0: /afs/$AFSCell/user/${user:0:1}/
    {user:1:1}/$user/; fi
    elif [ "$HomePrefix" = "=SMBHASH" ]
    then
        # Nothing defined yet for SMB Homedir hashes
        echo $HomePrefix
        if [ "$DEBUG" = "1" ]; then /usr/bin/logger -is HomePrefix1: $HomePrefix; fi
    else
        echo $HomePrefix
        if [ "$DEBUG" = "1" ]; then /usr/bin/logger -is HomePrefix2: $HomePrefix; fi
    fi
}

#####
# Create links to a directory mapping in /Users/$UNIONNAME/$1 to $2
function link_directory () {
    # $APPLE is the "Apple-standard" directory location we're going to map
    APPLE=$1
    # $REAL is the "real" location we'll be mapping to
    REAL=$2

    APPLEPATH=/Users/$UNIONNAME/$APPLE
    if [ "$DEBUG" = "1" ]; then /usr/bin/logger -is "APPLEPATH=$APPLEPATH"; fi

    if [ ${REAL:0:1} = "/" ]
    then
        PREFIX=""
    else
        PREFIX=$HomePrefix
    fi
    if [ "$DEBUG" = "1" ]; then /usr/bin/logger -is "PREFIX=$PREFIX"; fi
    BUILDPATH=$PREFIX$REAL
    if [ "$DEBUG" = "1" ]; then /usr/bin/logger -is "BUILDPATH=$BUILDPATH"; fi

    # # and, if it doesn't (and we have $CreateIfNotPresent is set), create
    # the target directory and permit it to the user
    if [ "$CreateIfNotPresent" = "true" ]
    then
        if [ "$DEBUG" = "1" ]; then /usr/bin/logger -is " Create: $BUILDPATH"; fi
        mkdir -p $BUILDPATH
        if [ "$DEBUG" = "1" ]; then /usr/bin/logger -is " Copy: $APPLEPATH to $BUILDPATH"; fi
        ( cd $APPLEPATH ; tar cf - . ) | ( cd $BUILDPATH ; tar xf - )
    fi

    # remove any existing directory first
    if [ -d $APPLEPATH ]
    then
        rm -rf $APPLEPATH
        if [ "$DEBUG" = "1" ]; then /usr/bin/logger -is " Remove: $APPLEPATH"; fi
    fi

    # FINALLY, make the link
    if [ "$DEBUG" = "1" ]; then /usr/bin/logger -is " Link: ln -s $BUILDPATH $APPLEPATH"; fi
    ln -sf $BUILDPATH $APPLEPATH
}

#####
## M A I N #####
#####

if [ ${UNIQUIID} -lt 1000 ]
then
    /usr/bin/logger -is -t HomeLinks -p user.info "Nothing to be done for $UNIONNAME (id=$UNIQUIID)"
    exit
fi

export CreateIfNotPresent=`PLB_get ":CreateIfNotPresent"`
HomeFS=`PLB_get ":HomeFS"`
if [ "$HomeFS" = "AFS" ]
then
    /usr/bin/logger -is -t HomeLinks -p user.info "Using AFS"
    if [ "$DEBUG" = "1" ]; then /usr/bin/logger -is " AFS: `tokens`"; fi
    export AFSCell=`cat /var/db/openafs/etc/ThisCell`
    export plistAFSCell=`PLB_get ":HomeAFSCell"`
    if [ "$plistAFSCell" != "" ]
    then
        export AFSCell=$plistAFSCell
        /usr/bin/logger -is -t HomeLinks -p user.info "Overriding ThisCell setting with: $AFSCell"
    fi
    export HomePrefix=`PLB_get_Prefix $UNIONNAME`

    /usr/bin/logger -is -t HomeLinks -p user.info "Creating directory structures for $UNIONNAME..."
    # $UNIONNAME directory should already be there

    # dump all Directories into a temp file
    $PLB -c "print :Directories" $CONFIGFILE | grep -v "}" | grep -v "Dict" | sort > /tmp/$UNIONNAME.login

    # read each line and build the directory
    while read variable equals value
    do
        # ready to link directory
        link_directory $variable $value
        done < /tmp/$UNIONNAME.login
        rm /tmp/$UNIONNAME.login
    elif [ "$HomeFS" = "SMB" ]
    then
        /usr/bin/logger -is -t HomeLinks -p user.info "Using SMB"
    else
        /usr/bin/logger -is -t HomeLinks -p user.info "ERROR: Unsupported Home FS!"
    fi
fi
```

# HomeLinks - *Version 2 (Dec04)*

- Needs to test for the existence of HomeDir:  
A non-existent HomeDir results in a login that  
“jumps off a cliff”

# HomeLinks - *Version 2 (Dec04)*

- Needs to test for the existence of HomeDir:  
A non-existent HomeDir results in a login that “jumps off a cliff”
- Needs to check for the success of the ditto runs: quotas or other limitations result in an incomplete template copy, possibly fatal to the session

# HomeLinks.mpkg

Complete UM-Installable Meta-Package containing:

HomeLinks v2  
LoginHooks  
LabHomeDirs  
plistbuddy

**AVAILABLE SOON**

<https://www.math.lsa.umich.edu/software>

(requires UMICH Kerberos)

# Questions?

**Jeff Kopmanis, MathIT**

kopmanis@umich.edu

<https://www.math.lsa.umich.edu/software>