Sudo: You're Doing It Wrong

BSDCan 2014

Michael W Lucas
https://www.MichaelWLucas.com
http://blather.MichaelWLucas.com
@mwlauthor

About Me

- Latest tech books: Absolute OpenBSD, DNSSEC Mastery, Sudo Mastery
- This class based on Sudo Mastery

My teaching style

- Based on your reactions
- How often I've done this class

Topics

- Sudo and sudoers
- Editing and Testing sudoers
- Lists and Aliases
- Options and Defaults
- Shell Escapes and Editors
- Configuring
- Shell Environments
- Sudo as IDS
- Complex policies
- LDAP
- Logging & Debugging

"Welcome to hell, kid"

- There are many ways to use sudo (or any tool)
- I recommend some
- I discourage others
- I offer choices on still more

Choose the pain that fits your threat and risk models

What's the Problem?

- Permissions and access control on Unix-like systems is a right pain
- UID- and GID-based accessed developed for campus environments
- root owns everything
- setuid and setgid? Sometimes
- ACLs? Kill me now.

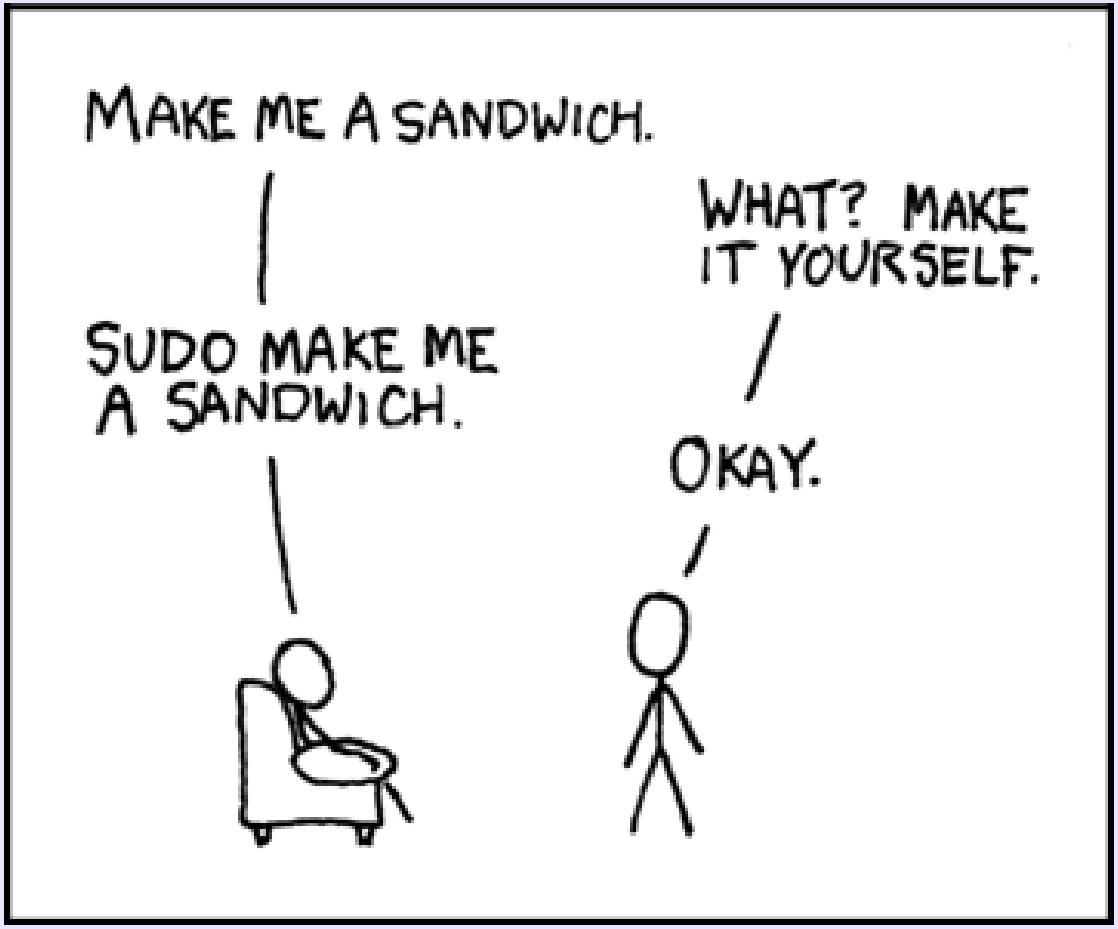
What is Sudo?

- Allows delegation of specific root privileges
- Allows users to run specific commands as a different user
- Cross-platform
- Works within traditional Unix privilege schematics
- ACLs? Selinux? Pfexec? Roles? All platform-specific.

What's wrong with sudo?

- Sudo is a specific solution to a specific use case
- Sudo adds another layer of system administration
- Some proprietary UNIX(tm) include their own delegation tool (pfexec, pbrun, etc).

You can stop sending me this now



http://xkcd.com/149/

Wrong use of sudo

- replacing su
- Avoiding authentication
- Not understanding what you're doing

Avoiding sudo

- Don't use sudo to edit files – use file permissions and groups.
- Create new groups for shared files (zone files, web server config, etc)
- Use sudo for service management

Proper Use of sudo

- Dividing privilege protects and permits:
  - Responsibility
  - Blame
  - Protection
  - Passwordless vs Password Privilege
  - Making the Boss Do His Job

Learning sudo

- If sudo is your only way to get privileged access, and you break sudo, you get to do the "single user mode shuffle"
- Set a root password
- Make sure you can use it

Avoiding sudo

- Don't use sudo to give users access to a file – that's what groups are for
- Groups? You know, the middle 3 permissions bits

  -rw-r--r--
- Create a group. Put users in the group. Give the group permissions.
- Use id(1) to see which groups you are in

Programs versus Groups

- Programs that perform privileged actions can't use groups to get that access
- You can't give httpd privilege to bind to port 80 and 443
- You can give a user privileges to run httpd and bind it to those ports – this is where sudo comes in.

Sudo 101

- Run sudo followed by the privileged command

```
$ sudo mount fileserver:/home/mike /mnt
Password:
mount_nfs: fileserver: host not known
```

- Enter your password, not the root password

Run a Command as Another User

- Use the -u flag

```
$ sudo -u oracle sqlplus
```

- Enter your password, not the root password

Run a Command as Another Group

- Some software requires a specific primary group
- Usually stupid commercial stuff
- Use the -g flag

```
$ sudo -g operator dump
$ sudo -g #5 dump
```

Sudoers 101

- Sudo seems simple?
- The complex stuff is in /etc/sudoers, aka sudoers
- Learning from the example provided with your OS is boring and limiting. Copy it to a safe place and start from scratch.

Sudo Policy Format

```
User Host=(RunAs) Command
```

- User = who can do this
- Host = which host this applies to
- RunAs = target user (optional)
- Command = the privileged command

Default Sudo Policy

```
%wheel ALL = ALL
```

- If you're in the `wheel` group, you get total access.
- Fine for system owners, not for enterprises
- No root password=no recovery

Smaller Sudo Policies

```
mike ALL = ALL
```

- User `mike` can run anything

```
mike dns1 = ALL
```

- User `mike` can run anything on the host dns1

```
mike dns1 = /usr/sbin/service named
```

- User `mike` can manage the named service on host dns1

Multiple Rules

Each unique combination needs its own sudoers rule

```
mike dns1 = /sbin/reboot
mike dns1 = /sbin/dump
rwatson dns1 = /sbin/reboot
rwatson dns1 = /sbin/dump
```

- This gets cumbersome quickly, so...

Lists

- Separate list items with a comma.
- Split long lines with a backslash

```
mwlucas,rwatson  dns1,dns2 = \
 /sbin/service named, /sbin/service syslogd
```

Multiple Privilege Levels

- Each level of access needs its own sudoers rule

```
mwlucas,rwatson dns1 = /sbin/reboot
dlangille ALL = ALL
```

- Dan is in charge

Select Users

- Use the optional RunAS to specify a target user

  `kate db1 = (oracle) ALL`

- Kate can do anything as the user `oracle`.

Negation

- ! means "everything but"
- Usable for usernames and hosts, not for commands

%wheel,!mwlucas ALL = ALL

Negation on Commands

- Negating commands parses but is not useful

```
mwlucas ALL = ALL,!/bin/sh
```

Negation on Commands

- Negating commands parses but is not useful

```
mwlucas ALL = ALL,!/bin/sh
```

```
$ cp /bin/sh /tmp/myscript
$ sudo /tmp/myscript
```

Sudoers Processing

- Rules processed in order
- Last matching rule wins
- Sudoers must end in a blank line

Editing Policy File

- Use visudo to edit /etc/sudoers

- Copies policy file to temp file
- You edit temp file
- Parses edited file
- Installs or rejects file

Editing Policy File

- Use visudo to edit /etc/sudoers

- Copies policy file to temp file
- You edit temp file
- Parses edited file
- Installs or rejects file
- Don't like vi? Set $EDITOR

Testing sudoers

- What access do I have?

  ```
  $ sudo -l
  ```

- What access do other users have?

  ```
  $ sudo -U username -l
  ```

- Test your changes before telling user they're set

Pattern Matching in sudoers

- ? – any single character
- [1-5] – numbers 1-5
- [0-9]* – any number of numbers
- * – match everything (except / in command names)

```
mwlucas dns[1-4]=ALL

mwlucas ALL = /usr/local/sbin/*
```

Pattern Matching in sudoers 2

- [acQ] – single character a, c, or Q
- Backslash *, ?, [, and] to literally match
- "" – disallow arguments

```
mwlucas ALL=/opt/bin/program -[acQ]
mwlucas ALL=/opt/bin/program2 ""
```

Dangers of Wildcards

```
Pete ALL=/bin/cat /var/log/messages*
```

- So you can view all the /var/log/messages archives...

Dangers of Wildcards

```
Pete ALL=/bin/cat /var/log/messages*
```

- So you can view all the /var/log/messages archives...

```
$ sudo cat /var/log/messages /etc/shadow
```

- ...and all the other files in the system

Aliases

- A named list of items

```
Cmnd_Alias BACKUP = /sbin/dump,
/sbin/restore,   /usr/bin/mt
```

- Fill any place in a rule

```
mwlucas ALL=BACKUP
```

Alias Names

- Alias names include capital letters, numbers, and underscores

  `CUSTOMERS` – ok

  `_CUSTOMERS` – not ok

  `2CUSTOMERS` – not ok

Aliases 2

- User and Host aliases

```
User_Alias TAPEMONKEYS = mwlucas,
jgballard

Host_Alias WWW = web1, web2, web3

TAPEMONKEYS WWW=BACKUP
```

RunAs Aliases

- List of users to run commands as

```
Runas_Alias DB=oracle, pgsql, mysql
```

- Use as a target user

```
fred DB = (DB) ALL
```

User Information Sources

- Users by username
- UID, prefaced by #
- Group names, prefaced with %
- Group ID, prefaced by %#
- Netgroup, prefaced by +

Non-Unix User Information Sources

- Non-Unix username (AD), prefaced by `%:`

  enclose in quotes if there's a space
- Non-Unix group ID, prefaced by `%:#`

```
User_Alias WHINERS = "%:Domain Users" \
   %operator, MINIONS
```

Hostnames

- Parsed from hostname(1)
- If hostname returns FQDN, drop the domain
- IP addresses
  - If host has multiple IPs, any address triggers rules
- CIDR 192.0.2.0/24
- Netgroups – preface with +

```
carl 192.0.2.0/25,+db = ALL
```

Excess Access

- Ease of rules versus concise rules

```
carl DB=(oracle, postgres, mysql) ALL
```

Negation in Lists

- Use negation to remove items from a list

```
User_Alias NOTSCUM = %wheel, !mike
NOTSCUM ALL = ALL
```

Sample Sudoers Policy

```
%wheel  ALL = ALL
DBA DB = (DBUSERS) ALL
TAPEMONKEYS DB=/sbin/reboot
TAPEMONKEYS ALL=BACKUP, /sbin/reboot
mwlucas ALL = /usr/sbin/visudo
```

Aliases with sudo

- sudo -l shows expanded aliases

```
$ sudo -l
Password:
User mike may run the following
commands on this host:
(root) ALL, !/bin/sh, /bin/bash,
/bin/tcsh, /usr/bin/su
```

Options and Defaults

- Traditional sudo insults users who cannot type their password
- Disabled by boring Unixes seeking public acceptance
- Use Defaults in sudoers to set as default

```
Defaults insults
```

Integer Options

- A sudoers option that takes a number as an argument

```
Defaults insults, passwd_tries=9
```

- Some sudoers options use 0 to disable

```
Defaults insults, passwd_timeout=0
```

String Options

- A sudoers option that takes a string

```
Defaults insults, passwd_tries=9,
badpass_message = "Wrong password.
I have noted your incompetence in
the log. Don't think you're fooling
anyone."
```

Per-User Options

- Set sudoers options on a per-user basis

  ```
  Defaults:mwlucas insults
  ```

- Set options on per-group basis.

  ```
  Defaults:%wheel !lecture
  ```

Per-Host Options

- Set sudoers options for a specific host

```
Defaults@dev !lecture
Defaults@prod lecture=always
```

Per-Command Options

- Set options based on command run:

```
Defaults:%wheel !lecture

Defaults!/sbin/fdisk
lecture=always,
lecture_file=/etc/fdisk-lecture
```

- Tags (we'll get there) can be defaults

```
Defaults!ALL = noexec
```

Other Defaults

- RunAs – separate with >

```
Defaults>DBA insults
```

Conflicting Defaults

```
Defaults:mwlucas insults
Defaults!/usr/bin/su !insults
mwlucas ALL = /usr/bin/su
```

- What happens?

Conflicting Defaults

```
Defaults:mwlucas insults
Defaults!/usr/bin/su !insults
mwlucas ALL = /usr/bin/su
```

- What happens?

- Last match wins. Sudo does not insult me.

What's wrong with this?

```
$ sudo more /var/log/auth.log
```

What's wrong with this?

```
$ sudo more /var/log/auth.log
```

- Need privilege to view this log file
- More breaks you to a command prompt with !
- You're running commands as root – oops!

Preventing Escapes

```
mwlucas ALL = NOEXEC: ALL
```

- Commands may not execute commands
- Sudo visudo doesn't work – visudo tries to run an editor

Allowing Escapes

- The EXEC and NOEXEC control a program's ability to execute programs

```
Defaults!ALL NOEXEC

Cmnd_Alias MAYEXEC =
/bin/newaliases, /sbin/fdisk

mwlucas ALL = ALL, EXEC: MAYEXEC
```

- I cannot run commands that execute commands... except for ones that may.
- Visudo needs to execute a text editor.

Editing Files

```
mwlucas ALL=/usr/bin/vi /etc/named.conf
```

- This sucks.
  - Trapped in one editor
  - Editor can't exec, but I might legitimately want a shell escape

sudoedit

```
mwlucas ALL = sudoedit /etc/rc.conf
```

- Copies file to temp file
- Opens $EDITOR
- Edit file as yourself
- Copies temp file over original

Configuring sudo(1)

- Configure the sudo client behavior in sudo.conf
- View configuration with sudo -V

Sudo Plugins

- Plugins change core sudo behavior
- Parsing sudoers is a plugin

```
Plugin sudoers_policy sudoers.so
Plugin sudoers_io sudoers.so
```

- You can only use one policy engine at a time

Sudo Paths

- Paths tie a feature to a shared library

```
Path noexec
/usr/libexec/sudo/sudo_noexec.so
```

- This example is bogus

Sudo Settings

- Lots of sudo options are set to true or false. Sudo won't dump core unless you:

  ```
  Set disable_coredump false
  ```

- Operating system must support coredump of setuid programs.

Sudo Environment

- Sudo starts a carefully sanitized instance of your shell to run privileged commands in.
- So, what's wrong with:

```
$ sudo cd /opt/secret/bin
```

Sudo Environment

- Sudo starts a carefully sanitized instance of your shell to run privileged commands in.
- So, what's wrong with:

  ```
  $ sudo cd /opt/secret/bin
  ```

- New shell starts. Working directory changes. New shell exits. You're in your regular shell's working directory.

Exposing Secrets

- How do you look in a secret directory?

```
$ sudo ls /opt/secret/bin
```

```
$ sudo sh -c "cd /opt/secret ; du -d0 | sort -rnk 6"
```

- Lots of ways to skin this problem

Retained Environment Variables

- Sudo retains these environment variables.

- `$TERM`
- `$PATH`
- `$HOME`
- `$MAIL`
- `$SHELL`
- `$LOGNAME`
- `$USER`
- `$USERNAME`

Sanity Check Environment

- Some environment variables get checked for stupidity

  `$TERM`

  `$LINGUAS`

  `$LC_*`

  `$LANGUAGE`

  `$LANG`

  `$COLORTERM`

Environment Variables

- You want environment variables preserved? Enumerate them in sudoers with env_keep

```
Defaults env_keep+="HOME SSH_CLIENT
SSH_CONNECTION SSH_TTY
SSH_AUTH_SOCK"
```

- Note the +=

Environment and Users

- Retain environment variables for specific users:

```
Defaults%wheel env_keep+="HOME
SSH_CLIENT SSH_CONNECTION SSH_TTY
SSH_AUTH_SOCK"
```

Don't Purge Environment

- Disable env_reset to not purge environment

```
Defaults !env_reset
```

- Purge specific variables

```
Defaults env_delete += "LD_LIBRARY_PRELOAD"
```

- This is enumerating badness. Don't do it.
- Environment settings sudo is hard-coded to remove (sudo -V) are always removed – you cannot override them

Environment vs Shells

- Running sudo sh?
- Can read in environment from config file

Users Overriding Environment

- Use the SETENV and NOSETENV environment variables to allow a user to ***request*** that sudo not reset their environment.

```
dan dbtest = (oracle) SETENV: /opt/oracle/bin
```

- Lets the user test environment on test server
- Request by using the -E flag

```
$ sudo -E -u oracle sqlplus
```

Users Environment Option

- Use SETENV and NOSETENV as options.

  ```
  Defaults:mike setenv
  ```

- This user can ask sudo to not reset their environment anywhere.
- Remember, environment filtering is not just to control users – it limits the damage an intruder can inflict

Target User Environment

- Some programs need a specific environment, configured for a specific user.
- Completely ditch your environment with -i.

```
$ sudo -u oracle -i sqlplus
```

- Uses target user shell.
- Common for application servers

Adding Environment Variables

- You might want a specific environment for certain users

```
Defaults env_file="/etc/sudoenv"
```

- File contains a list of environment variables, i.e.:

```
HTTP_PROXY=http://proxyhost:8080
```

What does your sudo do with environment?

- Run sudo -V as root to list how your sudo binary treats the environment
- Lists all variables to preserve, sanity-check, and remove

Sudo-specific Environment

- Four sudo-specific environment variables:
  - $SUDO_COMMAND (exact command you ran)
  - $SUDO_USER (original username)
  - $SUDO_UID (original UID)
  - $SUDO_GID (original GID)

Managing $PATH

- Intruders often try to sabotage $PATH

```
Defaults secure_path="/bin /sbin \
/usr/bin /usr/sbin /usr/local/bin \
/usr/local/sbin"
```

- If a command is not in the secure path, sudo fails

Sudo without Terminals

- Why run sudo without a terminal? Window manager menu item.
- No terminal, so cannot ask for password
- Set askpass in sudo.conf to prompt for alternate password program

```
Path askpass /usr/bin/openssh-askpass
```

Sudo for Intrusion Detection

- Sudo 1.8.7 and later can verify file checksums.
- Why?
  - Intruders
  - "I know how to fix this, I just need root!"
- Two steps:
  - Generate digests
  - Write sudo rule

Generate Digest

- Use openssl (or libressl)

```
$ openssl dgst -sha224 /usr/bin/passwd
SHA224(/usr/bin/passwd)= c6eab09e527dc…
```

- 56-character string

Sudoers Rule

- Use openssl (or libressl)

```
mike ALL = sha224:c6eab09e527d…
/usr/bin/passwd
```

- If checksum doesn't match, you'll get the generic "not allowed" message
- Write a script to generate these for all permitted programs

Hard Links

- Use an alias

  Cmnd_Alias SENDMAIL = sha224:89fc1... \

  /usr/bin/sendmail, /usr/bin/mailq, /usr/bin/hoststat, \

  ...

  `mike ALL = SENDMAIL`

Multiple Operating Systems

- Still use aliases

```
Cmnd_Alias FBSD-10-SENDMAIL = \
sha224:89fc1...  /usr/bin/sendmail, \
/usr/bin/mailq...

Cmnd_Alias PRECISE-SENDMAIL = \
sha224:89fc1...  /usr/bin/sendmail, \
/usr/bin/mailq...

Cmnd_Alias SENDMAIL = FBSD-10-SENDMAIL,\
PRECISE-SENDMAIL
```

Multiple Operating Systems

- Still use aliases

```
Cmnd_Alias FBSD-10-SENDMAIL = \
sha224:89fc1...  /usr/bin/sendmail, \
/usr/bin/mailq...

Cmnd_Alias PRECISE-SENDMAIL = \
sha224:89fc1...  /usr/bin/sendmail, \
/usr/bin/mailq...

Cmnd_Alias SENDMAIL = FBSD-10-SENDMAIL,\
PRECISE-SENDMAIL
```

Policy Guidelines

- Deny commands by default
- Do not exclude/negate commands
- Use NOEXEC
- Use aliases
- Remember: last match
- Visudo recovery rule last
- If tight restrictions aren't possible, don't try

Common Sudoers

- Write one sudoers policy, put it on all systems
- Must:
  - Rationalize hostname(1)
  - Consider DNS

Hostname

- Sudo compares the output of hostname(1) to the hostname in sudoers
- What does hostname on your systems say?

DNS?

- Sudo can use the hostname as given in forward and reverse DNS/first hosts table entry

```
Defaults fqdn
```

- Enabling DNS tells compare the fully qualified domain name to sudoers

```
%helpdesk www8.michaelwlucas.com =\
/usr/bin/passwd, !/usr/bin/passwd root
```

IP Addresses

- A rogue sysadmin could change a machine's hostname, but IP address is more difficult

```
Host_Alias WEBSERVERS 192.0.2.0/24
Host_Alias DBSERVERS 203.0.113.0/24
```

- Which host naming method should you use? Whichever will annoy you the least.

Including Files in Sudoers

- Include other files by reference, at location of include statement
- Include location is vital – you're hosed if the last line in your sudoers says:

```
%wheel ALL = !ALL
```

- *Always* put a visudo recovery rule at the end of sudoers.

Including Files

- For a specific file

  ```
  #include /etc/sudoers.local
  ```

- Per-host include file

  ```
  #include /etc/sudoers.%h
  ```

Including Directories

- Suck in everything in a directory

  ```
  #includedir /etc/sudoers.local
  ```

- Per-host include file

  ```
  #include /etc/sudoers.%h
  ```

- Files processed in lexical (ls(1)) order
  - Numbers before upper case
  - Upper before lower case
  - Lower case before accented

Oops! Errors in Include files

- Visudo only checks for /etc/sudoers errors
- To check or edit include files

  ```
  $ visudo -f /etc/sudoers.local
  ```

- Errors in one include file blow up the whole policy

Validating Distributed Sudoers

- Edit the centralized sudoers file
- Push it out to servers, and:
  - Sudo version mismatch?
  - Syntax errors?
  - Random stupidity?
- Have the various servers validate the new sudoers before they install it, or else...


/etc/sudoers

/etc/sudoers vs LDAP

- Sudoers file is local
- LDAP is remote
- If you have a lot of machines, you're probably already using it
- Compromise the machine?
  - Intruder owns sudoers
  - Intruder can't touch LDAP
- Choose your pain based on your threat model

LDAP sudo policies

- LDAP does not support aliases
  - Use LDAP groups for users and servers
- LDAP replies are nondeterministic
  - You cannot order attributes within a single LDAP sudo rule
  - You can order sudo rules, but you must remember this step
- Negations? No, no, no.

Sudo/LDAP Prerequisites

- You have LDAP working for authentication
- You can import LDIFs
- You can use an LDAP browser to edit
- You have a working sudoers-based policy
- I assume OpenLDAP in this class
  - Active Directory and Netscape schemas exist
- If you don't meet this, take 10 and come back for logging

LDAP-aware sudo(8)

- You need sudo linked against OpenLDAP
  - RedHat/CentOS – enabled by default
  - Debian – sudo-ldap package
  - FreeBSD – enable in port
  - OpenBSD – ask the devs when OpenBSD will integrate OpenLDAP into base
- Once sudo(8) can know about LDAP, configure LDAP to know about sudo

Sudo Schemas

- A schema is an LDAP data structure
- Sudo comes with schemas for:
  - OpenLDAP
  - Active Directory
  - Netscape iPlanet
- I assume OpenLDAP in this class

Sudo Schemas

- A schema is an LDAP data structure
- Sudo comes with schemas for:
  - OpenLDAP
  - Active Directory
  - Netscape iPlanet
- I assume OpenLDAP in this class

OpenLDAP sudo scheming

- Get the sudo OpenLDAP schema files
- Copy them to your schema directory

```
include /etc/openldap/schema/sudo.schema
index sudoUser eq
```

- Slap slapd

Sudoers Container

- Ask your LDAP Administrator
- OpenLDAP default is `ou=SUDOers,dc=example,dc=com`
- Sample container for mwlucas.org:

```
dn: ou=SUDOers,dc=mwlucas,dc=org
objectClass: top
objectClass: organizationalUnit
ou: SUDOers
```

Convert Sudoers to LDAP

- Use sudoers2ldif script
- Set $SUDOERS_BASE

```
$ SUDOERS_BASE=ou=SUDOers,dc=mwlucas,dc=org
$ export SUDOERS_BASE
$ sudoers2ldif /etc/sudoers > /tmp/sudo.ldif
```

- Fills in rule order for you
- You can blindly import this script

Dilapidated sudoers

- Start with a simple sudoers policy:

```
Defaults env_keep += "HOME SSH_CLIENT \
   SSH_CONNECTION SSH_TTY SSH_AUTH_SOCK"

%wheel,%sysadmins ALL=(ALL) ALL
```

- Creates a longer LDIF file
- Let's dissect it...

Line 1 LDIF

```
dn: cn=defaults,ou=SUDOERS,dc=mwlucas,dc=org
objectClass: top
objectClass: sudoRole
cn: defaults
description: Default sudoOption's go here
sudoOption: env_keep += "HOME SSH_CLIENT
SSH_CONNECTION SSH_TTY SSH_AUTH_SOCK"
sudoOrder: 1
```

Line 2 LDIF

```
dn: cn=%wheel,ou=SUDOERS,dc=mwlucas,dc=org
objectClass: top
objectClass: sudoRole
cn: %wheel
sudoUser: %wheel
sudoUser: %sysadmins
sudoHost: ALL
sudoRunAsUser: ALL
sudoCommand: ALL
sudoOrder: 2
```

Ordering

- Note these two entries – order is nondeterministic

```
sudoUser: %wheel

sudoUser: %sysadmins
```

- If rule order is important, use sudoOrder, i.e.:

```
sudoOrder: 2
```

Activating sudo(8) LDAP

- Ask sudo where it expects to find its config. (Some operating systems have sudo-specific LDAP config files.)

```
$ sudo -V

...

ldap.conf path: /etc/ldap.conf

ldap.secret path: /etc/ldap.secret
```

- Set server info in this file

ldap.conf and sudo

- sudoers_base – mandatory – the location of the sudoers container
- sudoers_search_filter – optional – LDAP filter to reduce number of results returned
- sudoers_timed – optional – enables checking policy expiration

Throwing the switch

- /etc/nsswitch.conf

  ```
  sudoers: ldap files
  ```

- If you should never check the local policy, remove "files" from the list
- Or, add "ignore_local_sudoers" option to LDAP policy
- Different failure modes

sudoRoles

- A one-line sudoers rule becomes a single LDAP sudoRole
- Both LDIF showed a few slides ago are sudoRoles
- LDAP import validates – no need for visudo
- Remember, "validates" != "what you want"

SudoRole Attributes

- Distinguished Name (DN)
- ObjectClass = sudoRole
- Common Name (CN)
- sudoUser
- sudoHost
- sudoCommand

sudoUser

- User name
- Can include
  - Operating system groups
  - Group IDs
  - Netgroups
  - Non-system groups need a client-side plugin – use LDAP groups instead

sudoUser

- Each username must appear in its own sudoUser entry within a sudoRole

```
sudoUser: %wheel

sudoUser: %sysadmins

SudoUser: mwlucas
```

sudoHost

- Hosts this rule applies to.
- Use hostnames, IP, and netgroups
- ALL matches all hosts

```
SudoHost: 192.0.2.0/24

sudoHost: www

SudoHost: 192.0.1.4
```

sudoCommand

- Full path to a command, plus args & wildcards
- ALL matches all commands

```
SudoCommand: /usr/bin/passwd
SudoCommand: sha224:d1... /usr/bin/passwd
SudoCommand: sudoedit /etc/named.conf
```

Optional Attributes

- sudoRunAsUser
- sudoRunAsGroup
- sudoOptions
- sudoOrder

sudoRunAsUser

- User to run the command as
- Just like a RunAs list

```
SudoRunAsUser: oracle
SudoRunAsUser: postgres
SudoRunAsUser: mysql
```

sudoRunAsGroup

- Group to run the command as

```
SudoRunAsGroup: operator
```

sudoOption

- Options to apply

```
SudoOption: !lecture, insults
```

sudoRuleOrder

- Emulates line ordering in sudoers
- Rules without sudoOrder are processed in random order
- Rules without sudoOrder appear first, all numbered rules override them

```
SudoRuleOrder: 87
```

SudoRule Timing

- LDAP-based sudo policy lets you have rules with activation & expiration times
- Uses sudoNotBefore and sudoNotAfter sudoRole attributes
- Times appear in UTC
- YYYYMMDDHHMMSSZ

```
SudoNotBefore: 20150101130000
SudoNotAfter: 20150201130000
```

SudoRule Timing Conflicts

- If you have multiple sudoNotBefore and sudoNotAfter, sudo uses the most permissive interpretation.
- Useless times are never matched

Disabling sudoers

- Do you want a local sudoers file?
  - Users might edit it
  - If LDAP fails, you lose sudo
- ignore_local_sudoers sudoOption disables sudoers – use at top of tree

  `cn=defaults,ou=sudoers,dc=example,dc=org`
- /etc/sudoers as backup? LDAP client without LDAP is basically hosed

Learning sudoRole schema

- Managing LDAP is Not My <bleeping> Job (tm)
- Managing sudo is not the <bleeping> LDAP Administrator's Job, either
- Use sudoers2ldif to convert snippets of /etc/sudoers to LDIF
- Modifying LDIF is much easier than writing from scratch

LDAP Caching

- You have a fully dilapidated network? Cool.
- LDAP now owns you.
- As of sudo 1.8.4, sudo supports SSSD (System Security Services Daemon).
- LDAP caching is its own special hell

Logging and Debugging

- Sudo has three logs:
  - Who ran what?
  - Sudo debugging log
  - Full session logs

sudo syslog

- You've seen these messages in the log:

```
Aug 27 23:34:44 www9 sudo: mike:
TTY=pts/1 ; PWD=/home/mike ;
USER=root ; COMMAND=/usr/bin/passwd
carl
```

- And the matching failure message

```
Aug 27 23:35:25 pestilence sudo: mike
: command not allowed ; TTY=pts/1 ;
PWD=/home/mike ; USER=root ;
COMMAND=/usr/bin/passwd root
```

sudo syslog messages

- You've seen these messages in the log:

```
Aug 27 23:34:44 www9 sudo: mike:
TTY=pts/1 ; PWD=/home/mike ;
USER=root ; COMMAND=/usr/bin/passwd
carl
```

- And the matching failure message

```
Aug 27 23:35:25 pestilence sudo: mike
: command not allowed ; TTY=pts/1 ;
PWD=/home/mike ; USER=root ;
COMMAND=/usr/bin/passwd root
```

sudo syslog facility & priority

- Sudo defaults to facility LOCAL2
- Success messages are level notice
- Failure messages are level alert

```
local2.=notice  /var/log/sudo
local3.=notice  /var/log/sudofail
```

- Use options syslog, syslog_badpri, and syslog_goodpri to change settings

Sudo and email

- Sudo sends email to root when a user doesn't use sudo correctly.
- Control with options:
  - mail_always
  - mail_badpass
  - mail_no_host
  - mail_no_perms
  - mail_no_user

Paranoia is a tool. Use it

- Have sudo failures emailed individually to helpdesk
- The phone call "I see you're having trouble. What's going on?" is *powerful*
  - They think you want to help
  - They think you know what they're doing
  - They think you're watching

Sudo Debugging Log

- Lets you watch sudo process your policy
- Configure in sudo.conf
- Divided by level (facility) and priority
- Priorities: debug, trace, info, diag, notice, warn, err, crit
- Lots and lots of levels...

Sudo Logging Levels

- Args
- Conv
- Edit
- Sudoedit
- Exec
- Main
- Pcom
- Plugin
- Selinux
- Utmp
- Alias
- Audit
- Auth
- Defaults
- Env
- Ldap
- Logging
- Match
- Nss
- Parser
- Perms
- Plugin
- Rbtree
- All
- Netif
- Pty
- util

Configure Debugging

- Needs 4 entries:
  - Debug statement
  - Program or plugin to be debugged
  - Log file location
  - Level and priority
- Don't know which level to debug? Start with "all"

```
Debug sudo /var/log/sudo_debug all@notice
```

Example: debugging LDAP

- Most of the rest of sudo only needs debugging when things go bad
- LDAP needs debugging when you sneeze

```
Debug sudo /var/log/sudo_debug
                    all@notice,ldap@info
```

How Useful is Debugging?

- For LDAP: very
- For most problems: not very
- Produce info for bug reports

Sudo Activity Log

- You know someone ran sudo /bin/sh, but not what they did in that session?
- Sudoreplay is your friend!
- Sudo can log all input and output from programs run under sudo, and replay them in real time.
- Enable in sudoers

Sudoreplay Hints

- Don't log sudoreplay sessions
- Don't log shutdown & reboot commands

```
Defaults log_output
Defaults!/usr/bin/sudoreplay !log_output
Defaults!/sbin/reboot !log_output
```

Sudoreplay Input Logs

- Also record what the user enters
- Only records what's echoed back to the users, but not all programs hide everything they should

```
Defaults log_input
```

Per-Command Logging

- Use the tags
  - LOG_INPUT
  - NOLOG_INPUT
  - LOG_OUTPUT
  - NOLOG_OUTPUT
- Apply to individual commands

Sudo Session Logs

- Stored in /var/log/sudo-io
- Change with iolog option
- Logs are tamper-evident but not tamper-proof
- Export off machine, append-only filesystem, sshfs, etc.

Listing Sudo Sessions

```
# sudoreplay -l
Sep 1 19:53:42 2013 : mike :
TTY=/dev/pts/1 ; CWD=/usr/home/thea ;
USER=root ; TSID=000001 ;
COMMAND=/usr/bin/passwd

Sep 1 20:04:42 2013 : thea :
TTY=/dev/pts/2 ; CWD=/usr/home/thea ;
USER=root ; TSID=000002 ;
COMMAND=/usr/local/bin/emacs
/etc/rc.conf
```

Replaying Sudo Sessions

```
# sudoreplay 000001

Replaying sudo session:
/usr/bin/passwd

Changing local password for root

New Password:

Retype New Password:
```

Replaying Notes

- Replays in real time
- < slows replay speed by 50%
- > doubles replay speed

Searching sudoreplay logs

- Who ran passwd?

```
# sudoreplay -l command passwd
```

- Who ran commands in /etc/?

```
# sudoreplay -l cwd /etc
```

- What did Lucas do this time?

```
# sudoreplay -l user mwlucas
```

Searching sudoreplay logs 2

- Who ran commands as a group member?

```
# sudoreplay -l group operator
```

- Who ran command as oracle?

```
# sudoreplay -l runas oracle
```

- Who was on the console

```
# sudoreplay -l tty console
```

sudoreplay searches by date

- What happened last week?

  ```
  # sudoreplay -l fromdate "last week"
  ```

- Who ran command on or after a date?

  ```
  # sudoreplay -l todate today
  ```

- What happened between 8PM and 11:59PM on 14 May 2014?

  ```
  # sudoreplay -l fromdate "8pm 14 May 2014" todate "11:59pm 14 May 2014"
  ```

Logical operators

- What did Lucas do last week?

  ```
  # sudoreplay -l fromdate "last week" user mwlucas
  ```

- Who ran command on or after a date?

  ```
  # sudoreplay -l command /bin/sh or command /bin/bash
  ```

- All together now!

  ```
  # sudoreplay -l (command bin/sh or command /bin/bash) user mwlucas
  ```

Sudo Authentication

- Lots of options to control how sudo handles authentication
- Lots of alternate authentication sources
- Choose wisely

Password Retries

- passwd_tries – how many times can user try to enter their password

```
Defaults passwd_tries=5
```

Password Prompt

- passprompt – text to tell the user to enter their password

```
Defaults passprompt "Enter YOUR password:"
```

- Escape characters can make this useful

```
Defaults passprompt "Your password on %h:"

Defaults passprompt "Enter %u's password
to run this command as %U:"
```

Authentication Cache

- Sudo does not remember your password
- It remembers that you successfully authenticated

```
# sudo -V | grep timestamp
Authentication timestamp timeout: 5.0 minutes
Path to authentication timestamp dir: /var/db/sudo
```

Changing the Time

- Use timestamp_timeout to tell sudo how long to cache a successful authentication
- Set to 0 to disable the cache

```
Defaults timestamp_timeout=0
```

- Using a negative values makes the timestamp immortal. Don't do that.
- Change timestamp directory with timestamp_dir

Invalidating the Cache

- Going to lunch? Invalidate your auth cache

```
$ sudo -K
```

Disabling Authentication

- In general, bad practice
- Makes sense for limited use, i.e., dhclient and ifconfig on your laptop

```
Defaults!
sbin/ifconfig,/sbin/dhclient !
authenticate
```

- Or:

```
mike ALL = NOPASSWD: /sbin/ifconfig
```

Shared Auth between Terminals

- Most sudo installs use username and terminal device in auth cache
- Isolating two processes owned by the same user is really hard – ptrace and gdb can break this
- Option ttytickets means "don't bother separating by terminal, it's bogus"
- Default on OpenBSD

The Lecture

- Option lecture
  - once – lecture each user once per host
  - always – lecture each user every time
  - never – don't lecture
- lecture_file – file containing personalized lecture

PAM

- This is no more a PAM class than an LDAP class.
- PAM is a black art
- Be *sure* you know what you're doing

PAM modules

- Google Authenticator? Removes the source of trust from your network
- Windows domain?
- RSA tokens?
- SSH agent is knock-off two-factor authentication, sort of.
- pam_ssh_agent_auth attach your SSH agent to PAM

pam_ssh_agent_auth

- Must have authorized_keys on local machine
- SSH agent forwarding enabled ($SSH_AUTH_SOCK)
- Sudo resets the auth cache whenever you authenticate. Disable the cache with timestamp_timeout=0

Activating SSH agent auth

- /etc/pam.d/sudo
- pam_unix module handles password auth
- Replace this entry with

```
auth sufficient pam_ssh_agent_auth.so
file=~/.ssh/authorized_keys
```

- In theory, you now need an SSH agent to authenticate to sudo
- pam_ssh_agent_auth must know the key file location and acceptable permissions

Testing new auth

- Flush the credentials and try sudo

  ```
  $ sudo -K
  $ sudo touch /tmp/test
  ```

- If broken, you'll get password prompts
- If fails silently, configure sudo logging

Freedom!

- Questions?

  Obligatory Advertisements:
- Blog: blather.michaelwlucas.com
- Twitter: @mwlauthor
- Mailing list: tech+subscribe@mailinglist.michaelwlucas.com