



# Securing Management Tools

Users Do Terrible Things

November 2015

# Keeping Your Management Tools Running

“If a user repeatedly disables your management tools, you no longer have a technical problem, you have a personnel problem.”

Me

“Never attribute to malice that which is adequately explained by stupidity.”

Hanlon's Razor

# Google's Setup

# How we do it

Configuration Management: puppet – ruby, with a python wrapper

Package Deployment: munki – python

Inventory, Logging, Security, &c.: python, Go, Objective-C

# How we do it

Configuration Management: puppet – ruby, with a python wrapper

Package Deployment: munki – python

Inventory, Logging, Security, &c.: python, Go, Objective-C

Did he just say “shebang”?



“ In computing, a shebang (also called a sha-bang, hashbang, pound-bang, or hash-pling), is the character sequence consisting of the characters number sign and exclamation mark (that is, "#!") at the beginning of a script. ”

# Shebang

Shell:

```
#!/bin/sh
```

Python:

```
#!/usr/bin/python
```

or

```
#!/usr/bin/env python
```

Ruby:

```
#!/usr/bin/ruby
```

# Troubles They Cause

# Breaking the system interpreters

```
$ sudo pip install pyobjc
```

# Breaking the system interpreters

```
$ sudo pip install pyobjc
```

```
[...]
```

```
File "/Library/MegaCorpSupport/bin/crankd.py", line 85, in init
```

```
    self = super(NotificationHandler, self).init()
```

```
AttributeError: 'super' object has no attribute 'init'
```

# Breaking the system interpreters

Installing Python 3

# Breaking the system interpreters

Installing Python 3

... and pointing `/usr/bin/python` to it.

Everything with `#!/usr/bin/python` will break in fascinating ways

# Breaking the system interpreters

Upgrading the system ruby to 2.2



# Breaking the system interpreters

Upgrading the system ruby to 2.2

... which breaks puppet 3.x

# Breaking the system interpreters

- Installing a malformed egg
- All sorts of crazy one-off damage to system frameworks
- **Deleting** system python/ruby entirely

# Actual support ticket

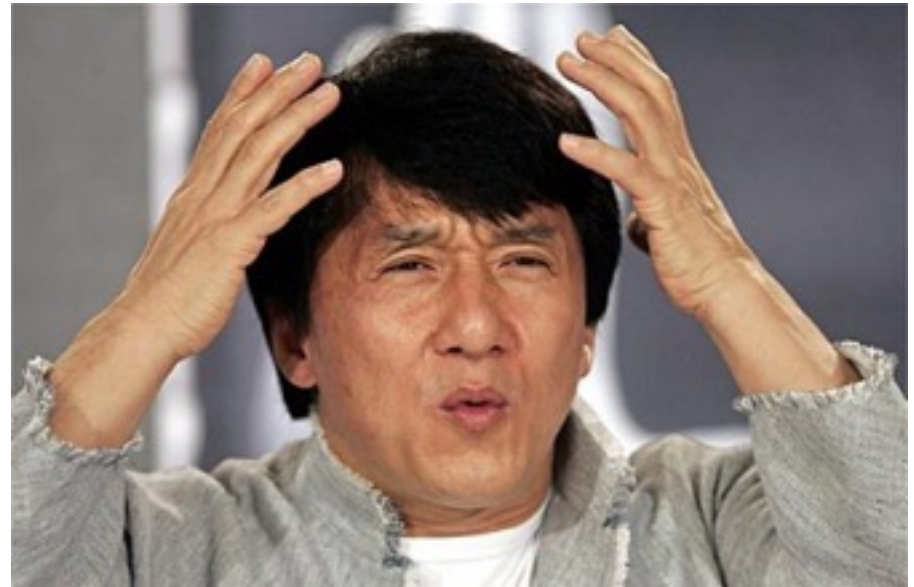
Is there a way for me to install the new certificate without using puppet? Many of us mac users are trying to avoid the google **puppet** and other related programs since **they come with several viruses.**

Anonymous

## Actual support ticket

Is there a way for me to install the new certificate without using puppet? Many of us mac users are trying to avoid the google **puppet** and other related programs since **they come with several viruses.**

Anonymous



# Actual support ticket

Is there a way for me to install the new certificate without using puppet? Many of us mac users are trying to avoid the google **puppet** and other related programs since **they come with several viruses.**

Anonymous



# Actual support ticket

Is there a way for me to install the new certificate without using puppet? Many of us mac users are trying to avoid the google **puppet** and other related programs since **they come with several viruses.**

Anonymous



# So what can we do?

# Users with admin access

Do your users have local admin rights?

No? So you're all good!



# Users with admin access

Do your users have local admin rights?

No? So you're all good!

Do they have physical access to the machine?

Then yes, they do have admin access.

# Users with admin access

Do your users have local admin rights?

No? So you're all good!

Do they have physical access to the machine?

Then yes, they do have admin access.

And they will unwittingly (and sometimes wittingly) break things.

That just made me nervous

No, really, what can you do?

# Attempt 1

Using configuration management to ensure `/usr/bin/python2.7` is a symlink to `/System/Library/Frameworks/Python.framework/Versions/2.7/bin/python2.7`

(Similarly for `/usr/bin/ruby`)

# Attempt 2

Let's create package of system python and ruby frameworks!

Add a script on each machine for support to use that installs the packages

# Attempt 3

Compile our own Python!

And Ruby!

And install them somewhere clever!

# An aside on OpenSSL



“OpenSSL is written by monkeys”

Marco Peereboom

# Secure Transport

# Secure Transport

```
if ((err = SSLHashSHA1.update(&hashCtx, &serverRandom)) != 0)
    goto fail;
if ((err = SSLHashSHA1.update(&hashCtx, &signedParams)) != 0)
    goto fail;
    goto fail;
if ((err = SSLHashSHA1.final(&hashCtx, &hashOut)) != 0)
    goto fail;
```

“ If you use OpenSSL in your publicly shipping apps, you must provide your own copy of the OpenSSL libraries, preferably as part of your app bundle; **the OpenSSL libraries that OS X provides are deprecated.** ”

Apple Cryptographic Services Guide, "Transmitting Data Securely"

# Apple is not good at this

OpenSSL is still used in many important areas, like Python and Ruby

```
$ python -c 'import ssl; print ssl.OPENSSL_VERSION'
```

OpenSSL 0.9.8zg 14 July 2015

```
$ ruby -r openssl -e 'puts OpenSSL::OPENSSL_VERSION'
```

"OpenSSL 0.9.8zc 19 Mar 2015"

# Apple is not good at this

OpenSSL is still used in many important areas, like Python and Ruby

```
$ python -c 'import ssl; print ssl.OPENSSL_VERSION'
```

OpenSSL **0.9.8zg** 14 July 2015

```
$ ruby -r openssl -e 'puts OpenSSL::OPENSSL_VERSION'
```

"OpenSSL **0.9.8zc** 19 Mar 2015"

So we're doomed

# Why should you care about OpenSSL?

Your management tools almost surely talk to networked resources.

Of course, you are using SSL everywhere, right?

Using an outdated and unpatched build of OpenSSL is not good



# So let's compile our own

```
build-openssl: l_Library_MegaCorp_openssl
/bin/mkdir "${OPENSSL_BUILD}"
cd "${OPENSSL_BUILD}"&& \
${CURL} -Lsf "${OPENSSL_DIST}" | ${TAR} -xzc "${OPENSSL_BUILD}" --strip-components=1 && \
perl ./Configure \
    --prefix="${WORK_D}/Library/MegaCorpSupport/openssl" \
    --openssldir="${WORK_D}/Library/MegaCorpSupport/openssl" \
    no-ssl2 no-ssl3 no-idea no-zlib no-comp shared \
    darwin64-x86_64-cc enable-ec_nistp_64_gcc_128 && \
make -s depend && \
sed -i.b -e "s%/tmp/the_luggage/openssl-[0-9]*/root%%" ./crypto/opensslconf.h \
./tools/c_rehash && \
make -s all && \
sudo make install && \
sudo install_name_tool -id \
    "/Library/MegaCorpSupport/openssl/lib/libcrypto.1.0.0.dylib" \
    "${WORK_D}/Library/MegaCorpSupport/openssl/lib/libcrypto.1.0.0.dylib" && \
sudo install_name_tool -id \
    "/Library/MegaCorpSupport/openssl/lib/libssl.1.0.0.dylib" \
    "${WORK_D}/Library/MegaCorpSupport/openssl/lib/libssl.1.0.0.dylib" && \
sudo install_name_tool -change \
    "${WORK_D}/Library/MegaCorpSupport/openssl/lib/libssl.1.0.0.dylib" \
    "/Library/MegaCorpSupport/openssl/lib/libssl.1.0.0.dylib" \
    "${WORK_D}/Library/MegaCorpSupport/openssl/bin/openssl" && \
sudo install_name_tool -change \
    "${WORK_D}/Library/MegaCorpSupport/openssl/lib/libcrypto.1.0.0.dylib" \
    "/Library/MegaCorpSupport/openssl/lib/libcrypto.1.0.0.dylib" \
    "${WORK_D}/Library/MegaCorpSupport/openssl/bin/openssl" && \
sudo install_name_tool -change \
    "${WORK_D}/Library/MegaCorpSupport/openssl/lib/libssl.1.0.0.dylib" \
    "/Library/MegaCorpSupport/openssl/lib/libssl.1.0.0.dylib" \
    "${WORK_D}/Library/MegaCorpSupport/openssl/lib/libcrypto.1.0.0.dylib" && \
sudo install_name_tool -change \
    "${WORK_D}/Library/MegaCorpSupport/openssl/lib/libcrypto.1.0.0.dylib" \
    "/Library/MegaCorpSupport/openssl/lib/libcrypto.1.0.0.dylib" \
    "${WORK_D}/Library/MegaCorpSupport/openssl/lib/libssl.1.0.0.dylib"
```

Just kidding

<https://github.com/google/macops/tree/master/packages/openssl>

# Ruby!

<https://github.com/google/macops/tree/master/packages/ruby>

```
configure-ruby: ${RUBY_BUILD}  
    cd "${RUBY_BUILD}" && \  
    ./configure --prefix=${WORK_D}/Library/MegaCorpSupport/Ruby \  
        --with-opt-dir=/Library/MegaCorpSupport/openssl \  
        --enable-load-relative
```

# Python!

<https://github.com/google/macops/tree/master/packages/python>

```
configure-python: ${PYTHON_BUILD} /Library/MegaCorpSupport/openssl
    cd "${PYTHON_BUILD}" && \
    echo "_socket socketmodule.c timemodule.c" >> Modules/Setup.dist && \
    echo "_ssl _ssl.c -DUSE_SSL "\
    "-I/Library/MegaCorpSupport/openssl/include " \
    "-I/Library/MegaCorpSupport/openssl/include/openssl " \
    "-L/Library/MegaCorpSupport/openssl/lib -lssl -lcrypto" >> \
    Modules/Setup.dist && \
    ./configure \
        --prefix="${WORK_D}/${PYTHON_PATH}" \
        --enable-ipv6 \
        --with-system-expat \
        --with-threads \
        --datarootdir="${WORK_D}/${PYTHON_PATH}/share" \
        --datadir="${WORK_D}/${PYTHON_PATH}/share"
patch ${PYTHON_BUILD}/Lib/ssl.py < log_cert.patch
```

# Python!

<https://github.com/google/macops/tree/master/packages/python>

```
${PYTHON_BUILD}: configure.ed setup.py.ed readline.c.ed
    /bin/mkdir -p ${@}
    ${CURL} -Lsf "${PYTHON_DIST}" | ${TAR} -xzc "${@}" --strip-components=1
    /bin/ed - ${@}/configure < configure.ed
    /bin/ed - ${@}/setup.py < setup.py.ed
    /bin/ed - ${@}/Modules/readline.c < readline.c.ed
```

# .ed files?

<http://www.opensource.apple.com/source/python/python-89/2.7/fix/configure.ed?txt>

```
g/-Wl,-search_paths_first/d
g/ -lSystemStubs -arch_only [^"]*/s///
g/-current_version $(VERSION)/s//current_version $(PYTHON_CURRENT_VERSION)/
g/-O3/s//-Os/g
/^OTHER_LIBTOOL_OPT$/i
EXTRA_CFLAGS
LIBTOOL
.
/# Calculate the right deployment target for this build\./i
    function vers2str() {
        printf '%02d%02d' `echo $1 | sed 's/\./ /'`
    }
    function cmpvers() {
        test `vers2str $1` "$2" `vers2str $3`
    }
.
/if test ${cur_target} '>' 10\.2/s/test/cmpvers/
/LIBTOOL_CRUFT=""/i
    LIBTOOL='$(CC) -dynamiclib -all_load $(CFLAGS)'
.
/# Use -undefined dynamic_lookup whenever possible (10\.3 and later)\./a
.
/if test ${MACOSX_DEPLOYMENT_TARGET} '>' 10\.2/s/test/cmpvers/
/LIBFFI_INCLUDEDIR="/s,","*/usr/include/ffi",
/^# check for endianness/i
# Enable dtrace
EXTRA_CFLAGS="$EXTRA_CFLAGS -DENABLE_DTRACE"
.
w
```

# ed is the standard text editor

```
$ ed
?
help
?
?
?
quit
?
exit
?
bye
?
eat flaming death
?
^C
?
^C
?
```

# Where was I?



# Python Virtual Environments

`virtualenv` is a tool that creates isolated, hermetic Python environments

All executables and libraries you need will be in the virtual environment.

# Python Virtual Environments

[https://github.com/google/macops/tree/master/packages/python\\_env](https://github.com/google/macops/tree/master/packages/python_env)

```
# Main python virtual environment, build from custom python+openssl
build-venv: l_Library_GoogleCorp_Python_Env
    sudo ${PYTHON_BASE}/bin/virtualenv -p ${PYTHON_BIN} ${WORK_D}/${VENV}

# Update to latest pip
build-pip: build-venv
    sudo ${WORK_D}/${VENV}/bin/pip install --upgrade --no-cache-dir --index-url ${DISTURL}/python_venv/simple pip

# pyopenssl and xattr are used by munki
build-pyopenssl: build-venv
    sudo ${WORK_D}/${VENV}/bin/pip install --no-cache-dir --index-url ${DISTURL}/python_venv/simple pyopenssl

build-xattr: build-venv
    sudo ${WORK_D}/${VENV}/bin/pip install --no-cache-dir --index-url ${DISTURL}/python_venv/simple xattr

# Python-ObjectiveC bridge
build-pyobjc: build-venv
    sudo ${WORK_D}/${VENV}/bin/pip install --no-cache-dir --index-url ${DISTURL}/python_venv/simple pyobjc-core
    sudo ${WORK_D}/${VENV}/bin/pip install --no-cache-dir --index-url ${DISTURL}/python_venv/simple pyobjc

# For locally caching pip data in $DISTURL/python_venv
build-pip2pi: build-venv
    sudo ${WORK_D}/${VENV}/bin/pip install --no-cache-dir --index-url ${DISTURL}/python_venv/simple pip2pi

build-readline: build-venv
    sudo ${WORK_D}/${VENV}/bin/pip install --no-cache-dir --index-url ${DISTURL}/python_venv/simple readline
```

# Python Virtual Environments

[https://github.com/google/macops/tree/master/packages/python\\_env](https://github.com/google/macops/tree/master/packages/python_env)

```
# Main python virtual environment, build from custom python+openssl
build-venv: l_Library_GoogleCorp_Python_Env
    sudo ${PYTHON_BASE}/bin/virtualenv -p ${PYTHON_BIN} ${WORK_D}/${VENV}

# Update to latest pip
build-pip: build-venv
    sudo ${WORK_D}/${VENV}/bin/pip install --upgrade --no-cache-dir --index-url ${DISTURL}/python_venv/simple pip

# pyopenssl and xattr are used by munki
build-pyopenssl: build-venv
    sudo ${WORK_D}/${VENV}/bin/pip install --no-cache-dir --index-url ${DISTURL}/python_venv/simple pyopenssl

build-xattr: build-venv
    sudo ${WORK_D}/${VENV}/bin/pip install --no-cache-dir --index-url ${DISTURL}/python_venv/simple xattr

# Python-ObjectiveC bridge
build-pyobjc: build-venv
    sudo ${WORK_D}/${VENV}/bin/pip install --no-cache-dir --index-url ${DISTURL}/python_venv/simple pyobjc-core
    sudo ${WORK_D}/${VENV}/bin/pip install --no-cache-dir --index-url ${DISTURL}/python_venv/simple pyobjc

# For locally caching pip data in $DISTURL/python_venv
build-pip2pi: build-venv
    sudo ${WORK_D}/${VENV}/bin/pip install --no-cache-dir --index-url ${DISTURL}/python_venv/simple pip2pi

build-readline: build-venv
    sudo ${WORK_D}/${VENV}/bin/pip install --no-cache-dir --index-url ${DISTURL}/python_venv/simple readline
```

# Putting it all together

```
#!/Library/MegaCorpSupport/bin/{python,ruby}
```

Just use your new custom python and ruby in all of your  
management tools

# Deploying this solution

Get the custom interpreter(s) installed everywhere

... using multiple delivery and enforcement mechanisms

Change the shebang of one of your tools

... and roll it out slowly

Change more of your tools

Change all of your tools

# Plan B

<https://github.com/google/macops/tree/master/planb>

# Self-repairing Management Tools

As much as possible, have your management tools try to repair themselves.



# They deleted the custom directory entirely!

Your monitoring should catch this

(You do have monitoring, right?)

# They deleted the custom directory entirely!

Your monitoring should catch this

(You do have monitoring, right?)

“ You’re giving a presentation  
on how to point to a different  
directory?”

My wife

THANK YOU

# Contact

[crc@google.com](mailto:crc@google.com)

[@salajander](#)

[plus.google.com/+ClayCaviness](https://plus.google.com/+ClayCaviness)

[github.com/google/macops/](https://github.com/google/macops/)

BeyondCorp

Lisa presentation: [goo.gl/lgYXoM](https://goo.gl/lgYXoM)

White paper: [goo.gl/6YdVjv](https://goo.gl/6YdVjv)