

# OS 9 Remote Shutdown

Jason Healy, Director of Networks and Systems

Last Updated Mar 18, 2008



# Contents

<b>1</b>	<b>OS 9 Remote Shutdown</b>	<b>5</b>
1.1	Introduction . . . . .	5
1.2	Requirements . . . . .	5
1.3	Configuring the Target . . . . .	6
1.3.1	Installation of Scripts . . . . .	6
1.3.2	Receive AppleEvents . . . . .	7
1.4	Configuring the Source . . . . .	8
1.5	Configuring NUT as a Trigger . . . . .	8
1.5.1	Setting Up NUT . . . . .	8
1.5.2	Sample Configuration Files . . . . .	9
1.5.3	Creating a Remote Shutdown Script . . . . .	10
1.5.4	Calling the Remote Shutdown Script . . . . .	11



# Chapter 1

## OS 9 Remote Shutdown

Last updated 2008/03/18

### 1.1 Introduction

We use **NUT** to manage our UPSes and automated shutdown of servers (see [our NUT documentation](#) for more information). However, NUT has not been ported to run on Mac OS 9, so our older servers would not automatically shut down when a power failure took place.

To work around this issue, we built a small set of scripts that send remote AppleScript events to a Mac running OS 9. While basic, these commands provide enough functionality to shut a machine down remotely.

While these scripts focus on power failure shutdowns, they are very basic and easily adapted to other uses. See the code in the following sections for more information.

### 1.2 Requirements

The full shutdown package relies on three main components:

1. A machine running Mac OS 9 (9.2 is assumed; earlier versions should work). **Note:** Mac OS versions 8 and below do **not** support these scripts; program linking only works over AppleTalk in these systems, but Mac OS X only supports program linking via IP, so they are not compatible. This

machine is the **target**; it will be shut down when it receives a remote command. AppleScript support must be enabled on this machine.

2. A machine running Mac OS X (10.3 or 10.4 is assumed; 10.2 may work). This machine is the **source**; it will send the remote shutdown command to the target.
3. A **trigger** program that will cause the remote command to be sent. In our case, the UPS software triggers the remote shutdown command, but any scriptable event can act as the trigger.

The **source** and **target** machines talk to each other via TCP/UDP port 3031 (eppc), so any firewall software must be configured to allow traffic *from* the source *to* the target.

## 1.3 Configuring the Target

To configure the target, we must install a few scripts on the system volume of the machine. We must also configure the system to receive and process remote AppleEvents.

### 1.3.1 Installation of Scripts

In the `scripts/target` directory of the project, you will find two scripts:

1. `CommTest.txt`
2. `ForcedShutDown.txt`

Copy these scripts onto the **target** machine (the one running Mac OS 9).

On the **target** machine:

1. Create a folder at the root level of the system hard drive called **RemoteShutdown**.
2. Open these scripts using the **Script Editor** application that comes with OS 9. You may need to open the scripts from within the application, or by dragging the scripts onto the application's icon.
3. For each script, choose **Save As Run-Only...** from the **File** menu. For the name, use the script's existing name, omitting the `.txt` extension (*e.g.*, `CommTest` and `ForcedShutDown`).
4. For the format, choose **Application**. Leave the **Stay Open** check box *unselected*. Make the **Never Show Startup Screen** check box *selected*.

5. Save the compiled script into the `RemoteShutdown` folder you created at the root level of the hard drive.
6. Run each compiled script by double-clicking on it. You should receive a dialog box with buttons for each script. Dismiss the dialog box when you get it.

You have now compiled the scripts correctly. Quit `Script Editor`. If you wish, you may delete the original script files (only the compiled versions are needed for regular operation).

### 1.3.2 Receive AppleEvents

The **target** machine must be configured to receive remote events from other computers:

1. Begin by opening the `File Sharing` control panel on the target.
2. Select the **Start/Stop** tab, ensure that **Program Linking** is *on*, and that the **Enable Program Linking clients to connect over TCP/IP** check box is *selected*.
3. Switch to the **Users & Groups** tab, and click on the **New User** button.
4. Choose a name and password for the new user. You may want to *deselect* the **Allow user to change password** check box. **Note:** passwords in OS9 must be **shorter** than 8 characters. Additionally, "odd" characters may not work, as we supply the password as part of a URL.
5. Change the popup menu from **Identity** to **Sharing**.
6. The **Allow user to connect to this computer** check box should be *deselected*.
7. The **Allow user to link to programs on this computer** check box should be *selected*.

Make note of the IP address (or DNS name) of this machine, the username you created, and the associated password. You'll need these items later to connect to the machine and execute the shutdown scripts.

**Note:** if you will use a single source to shut down multiple targets, you may wish to use the same username and password for each target. There are security implications to having the same username and password for each machine, but it also makes sending commands to a large group of machines much easier.

## 1.4 Configuring the Source

The **source** machine (running Mac OS X) requires no system configuration to work with the targets. The source simply needs to fire the remote AppleEvent and send it to the **target** machine over the network.

The simplest way to fire the event is by using a command-line AppleScript. Open a terminal on the source machine and execute the following (the command should be input as a single line):

```
osascript -l AppleScript -e "tell Application \"Finder\" of machine
\"eppc://username:password@hostname\" to open file \"CommTest\"
of folder \"RemoteShutdown\" of startup disk"
```

(A copy of this command lives in the **scripts/source** directory for this project.)

You must replace **username**, **password**, and **hostname** with the specific values for your target machine. The username and password are set in the **File Sharing** control panel on the target, and the IP address is set in the **TCP/IP** control panel.

Upon running the command, the target machine should launch the **CommTest** script and preset a dialog box. If it doesn't, the source machine should return an error message to help diagnose the problem.

Once you have the sample script working, you may try the **ForcedShutDown** script. If you are installing on a production machine, be careful not to shut it down accidentally!

## 1.5 Configuring NUT as a Trigger

We designed the remote shutdown script as part of our automated shutdown process during a power failure. This section describes how to integrate the scripts with NUT, our UPS management software. Note that the principles discussed here can be adapted to other uses with minimal effort.

### 1.5.1 Setting Up NUT

To use NUT as a trigger, NUT must be configured to trigger an **upssched** script when power-related events occur.

Configuring NUT is beyond the scope of this document; please refer to [our documentation for setting up NUT](#) for more information.

Before proceeding, you should have NUT configured to call **upssched** for ONBATT,

ONLINE, and FSD events. In turn, `upssched` should invoke a script on the system that correctly parses these events. We'll be modifying this script to add support for calling the remote shutdown scripts.

## 1.5.2 Sample Configuration Files

As an example, the config files from our stock NUT config are shown below. We'll be working from these configurations in the sections below:

### `upsmon.conf`

The `upsmon.conf` file must include support for calling `upssched` when power events occur. This means including the following lines:

---

```
NOTIFYCMD /usr/local/nut_upsmon/sbin/upssched

NOTIFYFLAG ONBATT WALL+SYSLOG+EXEC
NOTIFYFLAG ONLINE WALL+SYSLOG+EXEC
NOTIFYFLAG FSD WALL+SYSLOG+EXEC
```

---

You may need to modify paths or modify lines to your liking. The lines above are *an example of the minimum required configuration*.

### `upssched.conf`

The `upssched.conf` file tells NUT what script to call during power-related events, and which events to listen for. You can configure an event to trigger immediately (`EXECUTE`), or to wait a certain amount of time (`START-TIMER`).

In the sample below, we start a timer during on-battery events, but execute immediately when we receive a forced shutdown request (*e.g.*, when the battery is almost exhausted).

---

```
CMDSRIPT /usr/local/nut_upsmon/bin/early-shutdown

AT ONBATT * START-TIMER early-shutdown 120
AT ONLINE * CANCEL-TIMER early-shutdown
AT FSD * EXECUTE forced-shutdown
```

---

## early-shutdown

The script `early-shutdown` must handles the commands from `upssched` (e.g., `early-shutdown` and `forced-shutdown`) and makes the calls to any additional scripts. This is where we add our AppleScript code to shut down other machines.

Below is the stock `early-shutdown` script from our NUT package. At the moment, it simply shuts down the local machine. We'll be building on this script to add the code to shut down other machines.

---

```
case "${1}" in
    early-shutdown)
        logger -t early-shutdown "Early Shutdown time has arrived!"
        /usr/local/nut_upsmon/sbin/upsmon -c fsd
        ;;
    forced-shutdown)
        logger -t early-shutdown "UPS Master sent Forced Shutdown Request"
        /usr/local/nut_upsmon/sbin/upsmon -c fsd
        ;;
    *)
        logger -t early-shutdown "Unknown command: ${1}"
        ;;
esac
```

---

### 1.5.3 Creating a Remote Shutdown Script

We'll assume that we need to shut down multiple target machines from a single source. To do so, we'll write a script that keeps a list of machines that must be shut down, and then sends each machine the proper commands.

We have a sample script, called `shutdown_os9`, which is included in the `scripts/source` directory of this project. You should customize it for your needs.

Because this script contains usernames and passwords, its permissions should be set to only allow access to the NUT user (`ups_mon`, in this case):

```
chown nut_upsmon shutdown_os9
chmod 700 shutdown_os9
```

## 1.5.4 Calling the Remote Shutdown Script

To actually invoke the shutdown script, we must add it to the script that `upssched` calls. In our NUT package, that script is called `early-shutdown`. We simply add the call to the script in the right places, as shown below. We have a sample script, called `early-shutdown-with-remote`, which is included in the `scripts/nut` directory of this project.

Note that NUT must be correctly configured to call this script via `upssched`. See [our NUT documentation](#) for more information on configuring NUT.