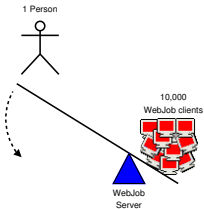


# What is WebJob?

(<http://webjob.sourceforge.net/WebJob/index.shtml>)

By Andy Bair

February 15, 2006



# Outline

## High-level View

One Sentence Description

The WebJob Client is ...

The WebJob Server is ...

Benefits

## Details: Client–Server Interaction

1. Client Requests Program
2. Server Authenticates Client
3. Server Sends File to Client
4. Client Receives and Executes Program
5. Client Uploads Results to Server

## Advantages

## Disadvantages

## Execution Example

Client-Side

Server-Side (part 1)

Server-Side (part 2)

Server-Side (part 3)

## WebJob in Action

## High-level View

One Sentence Description

The WebJob Client is ...

The WebJob Server is ...

Benefits

Details: Client-Server Interaction

1. Client Requests Program
2. Server Authenticates Client
3. Server Sends File to Client
4. Client Receives and Executes Program
5. Client Uploads Results to Server

Advantages

Disadvantages

Execution Example

Client-Side

Server-Side (part 1)

Server-Side (part 2)

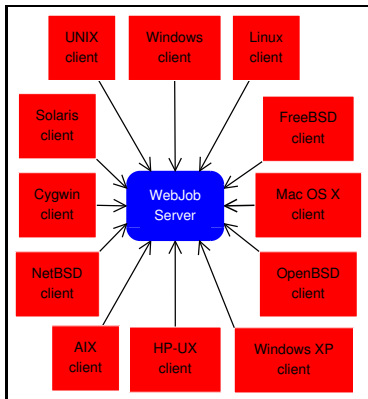
Server-Side (part 3)

WebJob in Action

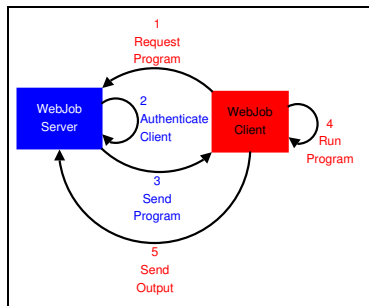
# One Sentence Description

WebJob is a client-server system, where the client requests and downloads a program from a server, executes that program on the client, then uploads the results to the server.

<http://webjob.sourceforge.net/WebJob/index.shtml>



High-Level View



Detailed View

## The WebJob Client is ...

- ▶ small C program
- ▶ approximately 1 megabyte when OpenSSL is statically compiled into the binary
- ▶ currently tested on these platforms/OS's: AIX, Cygwin, FreeBSD, HP-UX, MacOS X, NetBSD, OpenBSD, Linux, Solaris, and Windows NT/2K

# The WebJob Server is ...

- ▶ Apache configured to run the WebJob CGI (nph-webjob.cgi)
- ▶ nph-webjob.cgi is a Perl CGI program (~52 kilobytes)
- ▶ Contains the following (abbreviated) structure to support clients

```
webjob
|
+ incoming
| |
| - <job-N>.out
| - <job-N>.err
| - <job-N>.env
| - <job-N>.rdy
|
+ profiles
|
+ <client-N>
|
+ commands
|
- <command-N>
```

# Benefits

- ▶ mechanism for running known good programs on damaged or potentially compromised systems
- ▶ ideal for remote diagnostics, incident response, and evidence collection
- ▶ provides a framework that is conducive to centralized management
- ▶ can support and help automate a large number of common administrative tasks and host-based monitoring scenarios such as periodic system checks, file updates, integrity monitoring, patch/package management, and so on.

## High-level View

One Sentence Description

The WebJob Client is ...

The WebJob Server is ...

Benefits

## Details: Client-Server Interaction

1. Client Requests Program
2. Server Authenticates Client
3. Server Sends File to Client
4. Client Receives and Executes Program
5. Client Uploads Results to Server

Advantages

Disadvantages

Execution Example

Client-Side

Server-Side (part 1)

Server-Side (part 2)

Server-Side (part 3)

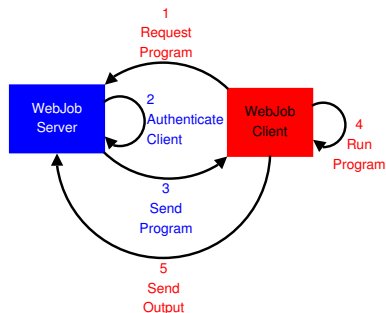
WebJob in Action



# 1. Client Requests Program

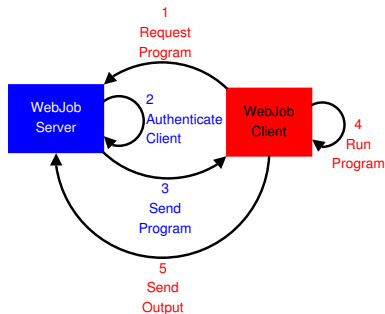
- ▶ WebJob client requests program from WebJob server
- ▶ Example requests the testenv program
- ▶ `--execute` directs client to execute program
- ▶ `--file` controls WebJob configuration via a configuration file

```
webjob --execute --file upload.cfg testenv
```



## 2. Server Authenticates Client

- ▶ WebJob server receives the request and authenticates the client's credentials
- ▶ WebJob server can be configured to authenticate clients with username–password combinations (i.e., basic auth) or SSL certificates certificates<sup>a</sup>.
- ▶ WebJob server can also be configured where clients have no authentication

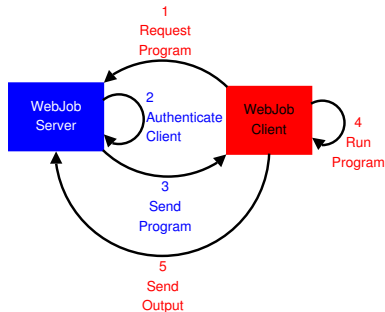


---

<sup>a</sup>[http://en.wikipedia.org/wiki/Public\\_key\\_certificate](http://en.wikipedia.org/wiki/Public_key_certificate)

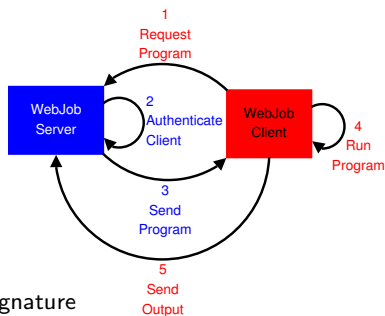
### 3. Server Sends File to Client

- ▶ If the client's credentials are authentic, the server sends the requested program to the client.



## 4. Client Receives and Executes Program

- ▶ client receives and executes the program
- ▶ optionally, client can validate (via GetHook) digitally signed binaries, providing a much greater level of security<sup>a</sup>.

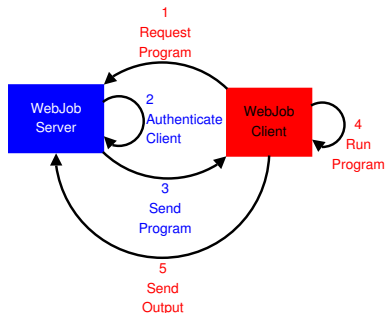


---

<sup>a</sup>[http://en.wikipedia.org/wiki/Digital\\_signature](http://en.wikipedia.org/wiki/Digital_signature)

## 5. Client Uploads Results to Server

- ▶ WebJob client uploads three files to the WebJob server
  - ▶ "out" file – command standard output stream
  - ▶ "err" file – command standard error stream
  - ▶ "env" file – timestamps, stream hashes, etc
- ▶ Server produces a fourth "ready" (*rdy*) file, which serves as a trigger file indicating the four files are ready for processing




# Advantages

There are many advantages to WebJob. These advantages are listed below – they are quoted from the WebJob website<sup>1</sup>

- ▶ Ported to many operating systems: UNIX's, Mac, Windows
- ▶ Small client footprint: only 1 binary, ~ 1 Mb
- ▶ Critical components centrally managed
- ▶ Secure; Client-Server data can be exchanged safely and securely using SSL encryption and certificate authentication.
- ▶ Aggregates data in one location – the WebJob server.
- ▶ Requires minimal networking: outbound TCP connection
- ▶ Does not diminish client security posture: client runs in security context of invoking user, client does not accept inbound requests, no inherent client SUID/SGID issues
- ▶ Jobs can be time limited: GET, RUN, and PUT timers
- ▶ Scales horizontally: 1 WebJob server can handle 1+ clients
- ▶ Scales vertically: WebJob servers can be configured as clients
- ▶ Does not limit what you can do

---


<sup>1</sup><http://webjob.sourceforge.net/WebJob/index.shtml> 

# Disadvantages

The disadvantages are listed below – they are quoted from the WebJob website<sup>2</sup>

- ▶ attacker could use client to infiltrate & execute malicious tools
- ▶ WebJob can't be completely trusted on a compromised host even when statically compiled – think kernel patch. The best you can hope for is to detect a breach before such a patch is put into effect. This could potentially be done by running host integrity checks on a frequent basis. By the way, if you suspect a kernel patch, your only true recourse is to take the system down and inspect it from another vantage point.
- ▶ To support batch processing, WebJob stores authentication credentials on the client system. Therefore, one must take measures to prevent and/or detect spoofing and replays.
- ▶ WebJob can't protect client-server exchanges when used without encryption and mutual authentication.

---

<sup>2</sup><http://webjob.sourceforge.net/WebJob/index.shtml> 

## High-level View

One Sentence Description

The WebJob Client is ...

The WebJob Server is ...

Benefits

## Details: Client–Server Interaction

1. Client Requests Program
2. Server Authenticates Client
3. Server Sends File to Client
4. Client Receives and Executes Program
5. Client Uploads Results to Server

## Advantages

## Disadvantages

## Execution Example

Client-Side

Server-Side (part 1)

Server-Side (part 2)

Server-Side (part 3)

## WebJob in Action



# Client-Side

```
$ cat client_1.cfg
```

```
ClientId=client_1
URLGetURL=http://127.0.0.1/cgi-client/nph-webjob.cgi
URLPutURL=http://127.0.0.1/cgi-client/nph-webjob.cgi
URLUsername=client_1
URLPassword=password
URLAuthType=basic
OverwriteExecutable=Y
UnlinkExecutable=Y
UnlinkOutput=Y
RunType=snapshot
TempDirectory=/opt/tmp
```

```
$ webjob -e -f /usr/local/etc/client_1.cfg testenv
```

## Server-Side (part 1)

```
$ cat /var/webjob/profiles/client_1/commands/testenv
```

```
#!/bin/sh
echo "WEBJOB_CLIENTID=${WEBJOB_CLIENTID}"
echo "WEBJOB_HOSTNAME=${WEBJOB_HOSTNAME}"
```

```
$ ls /var/webjob/incoming/
```

```
client_1_20060215144325_01252_testenv.env
client_1_20060215144325_01252_testenv.err
client_1_20060215144325_01252_testenv.out
client_1_20060215144325_01252_testenv.rdy
```

```
$ cat client_1_20060215144325_01252_testenv.out
```

```
WEBJOB_CLIENTID=client_1
WEBJOB_HOSTNAME=foo.bar.org
```

```
$ cat client_1_20060215144325_01252_testenv.err
```

## Server-Side (part 2)

```
$ cat client_1_20060215144325_01252_testenv.env
```

```
Version=webjob 1.5.0 ssl 32 bit
Hostname=foo.bar.org
SystemOS=i386 FreeBSD 5.4-RELEASE
ClientId=client_1
GetRequest=testenv
Command=testenv
CommandLine=testenv
Jid=server_1_1140032605_01250
Pid=1249
KidPid=1251
KidStatus=0
KidSignal=0
KidReason=The kid exited cleanly.
JobEpoch=2006-02-15 14:43:25 EST (1140032605.888789)
GetEpoch=2006-02-15 14:43:25 EST (1140032605.889020)
RunEpoch=2006-02-15 14:43:25 EST (1140032605.909310)
PutEpoch=2006-02-15 14:43:25 EST (1140032605.910637)
HashType=MD5
StdOutHash=5f1f3a64705eb49a46bf8047a555a812
StdErrHash=d41d8cd98f00b204e9800998ecf8427e
GetError=NA
RunError=NA
```

## Server-Side (part 3)

```
$ cat client_1_20060215080946_74296_hostname.rdy
```

```
Jid=server_1_1140032605_01250
BaseDirectory=/var/webjob
CapContentLength=N
ConfigSearchOrder=clients:commands
EnableConfigOverrides=Y
EnableJobIds=Y
EnableLogging=Y
FolderList=common
GetTriggerCommandLine=
GetTriggerEnable=N
MaxContentLength=100000000
OverwriteExistingFiles=N
PutNameFormat=%CID_%Y%m%d%H%M%S_%PID_%CMD
PutTriggerCommandLine=
PutTriggerEnable=N
RequireMatch=Y
RequireUser=Y
ServerId=server_1
SslRequireCn=N
SslRequireMatch=N
```

## High-level View

One Sentence Description

The WebJob Client is . . .

The WebJob Server is . . .

Benefits

## Details: Client–Server Interaction

1. Client Requests Program
2. Server Authenticates Client
3. Server Sends File to Client
4. Client Receives and Executes Program
5. Client Uploads Results to Server

## Advantages

## Disadvantages

## Execution Example

Client-Side

Server-Side (part 1)

Server-Side (part 2)

Server-Side (part 3)

## WebJob in Action

# WebJob Recipes

<http://webjob.sourceforge.net/WebJob/Cookbook.shtml>

## ▶ Database and Reporting

- ▶ Harvest system information, load it into MySQL, and create a set of browsable HTML reports

## ▶ Administration and Management

- ▶ Manage system config files, rc scripts, and other selected, text-based files
- ▶ Insert/Remove specified cron jobs on an as needed basis
- ▶ Manage root's crontab
- ▶ Periodically (hourly/daily) run administrative tasks
- ▶ Periodically run administrative tasks via command bundles (scripts)
- ▶ Deploy and verify the installation of a FreeBSD package
- ▶ Deploy and verify the installation of a Solaris package

# WebJob Recipes

<http://webjob.sourceforge.net/WebJob/Cookbook.shtml>

- ▶ Collection and Monitoring
  - ▶ Harvest and check Solaris nnd security settings
  - ▶ Harvest and monitor argus data
  - ▶ Harvest and monitor ps data
  - ▶ Harvest ftimes map/dig data from Windows platforms using self-extracting executables (NSIS)
  - ▶ Harvest Isof socket data (TCP/UDP)
  - ▶ Harvest uptime data once a minute and periodically rsync it to a central server
  - ▶ Run tcpdump on a group of IDS sensors to collect network traffic
- ▶ Compliance Testing and Patch Analysis
  - ▶ Run DISA's UNIX Security Readiness Review Scripts (SRRs)
  - ▶ Check Solaris patch levels for compliance with Sun Alert reports

# WebJob Recipes

<http://webjob.sourceforge.net/WebJob/Cookbook.shtml>

- ▶ Synchronization and Automatic Updates
  - ▶ Synchronize data (push/pull) using rsync, ssh, and dynamic keys
  - ▶ Automatically update or repair a webpage
  - ▶ Automatically update or repair a website
- ▶ Server-Side GET/PUT Triggers
  - ▶ Automatically compress WebJob uploads using triggers and configuration overrides
- ▶ Miscellanea
  - ▶ Run a command if its hash matches a predetermined value