

Deploying WebPageTest

Justin Phelps | @Linuturk

SALTCONF 15

Today we are going to talk about:
What WebPageTest is and it's architecture
Some neat things about WebPageTest and what it can do
The challenges I faced with both WebPageTest and SaltStack.
But first, a little about me...

About Me



Linuturk on Twitter & Github onitato.com

Plug the Salt Master template.

Justin Phelps, SSCE

DevOps Engineer at Rackspace

Configuration management and CI/CD are my passions these days. My team maintains the Heat orchestration templates at Rackspace.

So what is WebPageTest?

Follow Along

States: github.com/linuturk/webpagetest

Slides: onitato.com

About WebPageTest



WebPageTest is an open source tool used to test the performance of websites on various browsers and devices.

Originally written at AOL and open sourced in 2008.

Source code is hosted in Github. Open Source!

The WPO Foundation runs the public instance.

Browsers / Devices



The browsers shown here.
Android Phones and Tablets
let's see an example test result.

Web Page Performance Test for www.onitato.com

From: Chicago, IL - Chrome - Cable
1/13/2015, 5:04:17 PM

F	A	A	N/A	F	✓
First Byte Time	Keep-alive Enabled	Compress Transfer	Compress Images	Cache static content	Effective use of CDN

Tester: CHICAGO-198.38.92.101
Re-run the test

[Raw page data - Raw object data](#)
[Export HTTP Archive \(.har\)](#)
[See in ShowSlow](#)
[View Test Log](#)

	Document Complete			Fully Loaded		
	Time	Requests	Bytes In	Time	Requests	Bytes In
First View	3.747s	20	68 KB	4.025s	20	152 KB
Repeat View	0.721s	1	0 KB	0.721s	1	0 KB

Waterfall		Screen Shot	Video
First View (3.747s)			Filmstrip View Watch Video
Repeat View (0.721s)			Filmstrip View Watch Video

Content Breakdown		Requests	Bytes
		<ul style="list-style-type: none"> html 42.1% js 21.1% css 26.3% image 2.1% font 0.6% 	<ul style="list-style-type: none"> html 32.9% js 26.7% css 23.1% image 23.1% font 0.6%

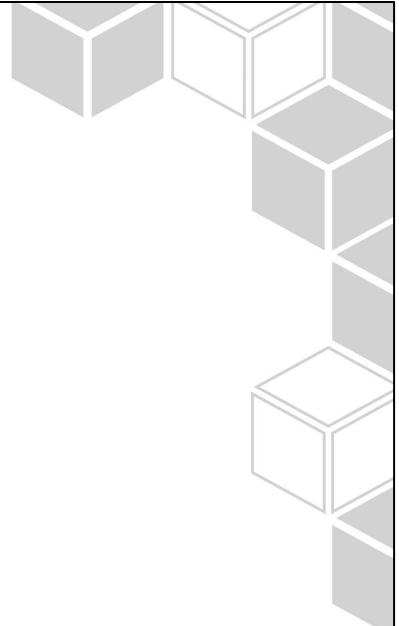
Video of the site loading

Grading scale for major categories

Waterfall graph of the page load

Full API access to results

http://www.webpagetest.org/result/150113_E7_18YJ/



API Demo

List Locations:

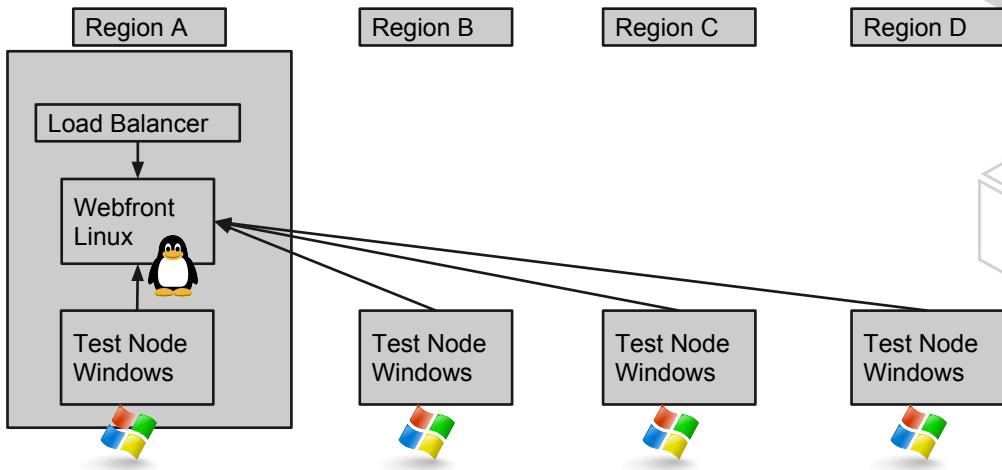
<http://webpagetest.rax.io/getLocations.php?f=json>

Submit test:

http://webpagetest.rax.io/runtest.php?url=onitato.com&f=json&location=Chicago_wptdriver:Chrome.LAN

Results: userUrl

Private Instance Architecture



Architecture of the private instance we have deployed internally.
Design limits you to a single Linux front end node.
Windows nodes poll the PHP app on the front node for work.
Test history and results are stored on the front node. There's potential for a backup front node, but test history would have to be sync'd.

The Easy Parts



Front end was a fairly basic PHP application on Linux. Behind nginx. Basic stuff for SaltStack.

Once installed and configured correctly, the WebPageTest agents themselves handle automatic updating of themselves and the browsers they have configured.

Salt-cloud is next

Salt-Cloud

Providers

Profiles

Maps

Salt Cloud - Provider

```
1 my-rackspace-config:
2     # Set the location of the salt-master
3     #
4     minion:
5         master: <IP or Hostname>
6
7     # Configure Rackspace using the OpenStack plugin
8     #
9     identity_url: 'https://identity.api.rackspacecloud.com/v2.0/tokens'
10    compute_name: cloudServersOpenStack
11    protocol: ipv4
12
13    # Set the compute region:
14    #
15    compute_region: IAD
16
17    # Configure Rackspace authentication credentials
18    #
19    user: <username>
20    tenant: <tenant>
21    apikey: <apikey>
22
23    provider: openstack
```

Don't forget to set the IP or DNS name for your master.
Rackspace Cloud in IAD.

Salt Cloud - Profile

```
1 wpt-linux:
2     provider: my-rackspace-config
3     size: 2 GB Performance
4     image: Ubuntu 14.04 LTS (Trusty Tahr) (PVHVM)
5 wpt-windows:
6     provider: my-rackspace-config
7     size: 4 GB Performance
8     image: 59901295-8ae6-4ab4-a697-01261799c8ac
9     win_installer: /root/Salt-Minion-2014.1.7-AMD64-Setup.exe
10    files:
11        'C:\cloud-automation\bootstrap.cmd':
12            /root/bootstrap.cmd
13    grains:
14        wpt_location: Location
15        wpt_label: Label
16        wpt_group: Group
17        wpt_browsers: "Chrome,Firefox,IE"
```

Note the `win_installer` and `files` options.

Grains are specific to web page test. You'll want a separate profile for each region you run.

Let's look at that `bootstrap.cmd` file.

Salt Cloud - Bootstrap

```
1 netsh advfirewall firewall add rule name="SaltBootstrap" dir=in protocol=TCP localport=445 action=allow
```

```
1 $CurrentVal = Get-NetFirewallRule -DisplayName SaltBootstrap
2
3 if ($CurrentVal.Enabled -eq "True") {
4     Disable-NetFirewallRule -DisplayName SaltBootstrap
5     Write-Output "changed=yes comment='Port 445 Disabled.'"
6 } Else {
7     Write-Output "changed=no comment='Port 445 Already Disabled.'"
8 }
```

Server Message Block (SMB)

Used server personality on the cloud to autorun this command when the server was created. You could also bake this into an image.

salt-cloud can't communicate with the server without this port.

Close it as part of your automation!

Salt Cloud - Map File

```
1 wpt-linux:
2   - web01.webpagetest.example.com
3   - web02.webpagetest.example.com
4 wpt-windows:
5   - win01.webpagetest.example.com
6   - win02.webpagetest.example.com
```

Map files are super useful for standing up infrastructure.
Let's look at the Pillar data for this deployment next.

Pillar

```
1 webpagetest:
2     sitename: "webpagetest.example.com"
3     zipurl: "https://github.com/WPO-Foundation/webpagetest/releases/do
4     zipsha: "sha256=19ee9df78205f99153c6fa80b6e6e98394cc680c4a9e13b31b
5     http: "apache2" # nginx or apache2
6     php: "php5" # php5 or php5-fpm
7     win:
8         user: "webpagetest"
9         pass: "changeme"
10        install_dir: 'C:\webpagetest'
11        temp_dir: 'C:\wpttemp'
12        zip_file: "webpagetest_2.15.zip"
13    packages:
14        - imagemagick
15        - libimage-exiftool-perl
16        - libjpeg-turbo-progs
17        - php5-curl
18        - php5-gd
19        - php5-sqlite
20        - php5-apcu
21        - python-software-properties
22        - unzip
```

example pillar data

sets values to be used later in various states, scripts, and files

Sitename is the dns name of the front end servers

zipurl zipsha ensure a good installer

can switch between nginx/apache php5/php5-fpm

win: includes settings for the windows portion, including the auto logon username and password.

packages: various dependencies for the webpagetest php front end.

Pillar

```
23 phpvars:  
24     upload_max_filesize: "12M"  
25     post_max_size: "12M"  
26     memory_limit: "512M"  
27     fpm: "unix:/var/run/php5-fpm.sock"  
28 settings:  
29     product: "WebPageTest"  
30     contact: "contact@domain.com"  
31     maxruns: 9  
32     allowPrivate: 0  
33     key: "changeme" #Do not use an & in your key  
34 locations:  
35     - name: TestLocation  
36         label: TestLabel  
37         group: TestGroup  
38         browsers: "Chrome,Firefox,IE"  
39         drivers:  
40             - wptdriver  
41             - urlblast  
42         default: True
```

Bottom half of the example pillar data
tuning php vars for the front end php app
settings for webpagetest itself.
Locations (list of hashes that define all available locations for a test node)

Powershell States

```
1 close-port-445:  
2   cmd.script:  
3     - source: salt://webpagetest/powershell/Set-ClosePort445.ps1  
4     - shell: powershell  
5     - stateful: True
```

```
1 $CurrentVal = Get-NetFirewallRule -DisplayName SaltBootstrap  
2  
3 if ($CurrentVal.Enabled -eq "True") {  
4   Disable-NetFirewallRule -DisplayName SaltBootstrap  
5   Write-Output "changed=yes comment='Port 445 Disabled.'"  
6 } Else {  
7   Write-Output "changed=no comment='Port 445 Already Disabled.'"  
8 }
```

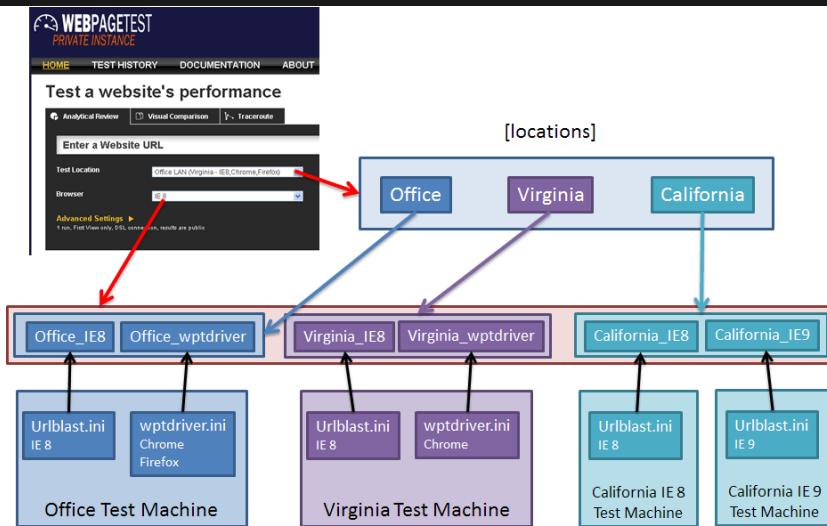
same as other cmd.scripts but set shell to ‘powershell’
take note of the use of stateful and the output written in the powershell - Best Practice!

WebPageTest Challenges



Configuring Locations
Windows Server + DummyNET Driver
Windows Automation
Mobile devices
The ampersand bug

Configuring Locations



This is from the WebPageTest documentation, and is there to explain how Locations should be configured.

Confused? So was I.

The basic idea is that all locations are defined on the Linux front end.

Then, you match your location settings on each Windows test node to the appropriate configuration on the front end.

I abstracted this into grains and pillar.

Configuring Locations

Pillar	Grains
webpagetest.locations.name	wpt_location
webpagetest.locations.label	wpt_label
webpagetest.locations.group	wpt_group
webpagetest.locations.browsers	wpt_browsers

```
34 locations:
35   - name: TestLocation
36     label: TestLabel
37     group: TestGroup
38     browsers: "Chrome,Firefox,IE"
39     drivers:
40       - wptdriver
41       - urlblast
42     default: True
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
59 wpt-windows:
60   provider: my-rackspace-config
61   size: 4 GB Performance
62   image: 59901295-8ae6-4ab4-a697-01261799c
63   win_installer: /root/Salt-Minion-2014.1.
64   files:
65     'C:\cloud-automation\bootstrap.cmd':
66       /root/bootstrap.cmd
67   grains:
68     wpt_location: Location
69     wpt_label: Label
70     wpt_group: Group
71     wpt_browsers: "Chrome,Firefox,IE"
```

[Define locations in Pillar](#) and iterate over this list to write the front end location configuration.

[Configure grains in salt-cloud](#) and write their configs using grain data.

As long as those entries in Pillar match the same Grain data, your locations should work.

Xen + DummyNET Driver



UPPSALA
UNIVERSITET

Can't perform traffic shaping without this DummyNET Driver.

The Xen hypervisor isn't compatible without special changes that aren't allowed on the Rackspace Public Cloud. (AWS as well.)

mindinst.exe allowed me to automate the installation of the DummyNET network binding. There isn't a way to do this with pure PowerShell.

Uppsala University is a research university in Uppsala, Sweden, and is the oldest university in Sweden, founded in 1477.

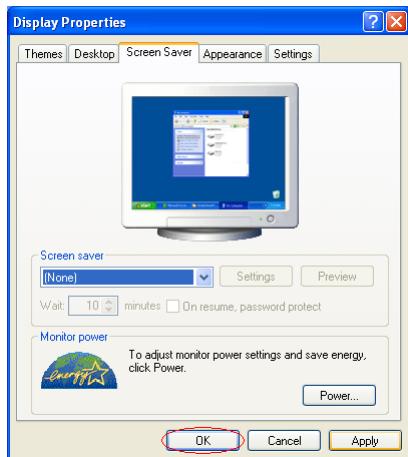
Auto Logon



```
1 $LogonPath = 'HKLM:\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon'
2 $LastUser = 'HKLM:\Software\Microsoft\Windows\CurrentVersion\Authentication\LogonUI'
3 $Domain = "{$ grains.host }"
4 $Username = "{$ pillar['webpagetest']['win']['user'] }"
5 $Password = "{$ pillar['webpagetest']['win']['pass'] }"
6
7 $CurrentVal = Get-ItemProperty -Path $LogonPath -Name AutoAdminLogon
8
9 If ($CurrentVal.AutoAdminLogon -eq "1") {
10   $CurrentUser = Get-ItemProperty -Path $LogonPath -Name DefaultUserName
11   $CurrentPass = Get-ItemProperty -Path $LogonPath -Name DefaultPassword
12
13   If ($CurrentUser.DefaultUserName -ne $Username -Or $CurrentPass.DefaultPassword -ne $Password) {
14     Set-ItemProperty -Path $LogonPath -Name DefaultUserName -Value "$Username"
15     Set-ItemProperty -Path $LogonPath -Name DefaultPassword -Value "$Password"
16     Write-Output "changed=yes comment='Credentials Updated.'"
17   } Else {
18     Write-Output "changed=no comment='AutoLogon already enabled.'"
19   }
20 } Else {
21   Set-ItemProperty -Path $LogonPath -Name AutoAdminLogon -Value "1"
22   Set-ItemProperty -Path $LogonPath -Name DefaultDomainName -Value "$Domain"
23   New-ItemProperty -Path $LogonPath -Name DefaultUserName -Value "$Username"
24   New-ItemProperty -Path $LogonPath -Name DefaultPassword -Value "$Password"
25   New-ItemProperty -Path $LogonPath -Name DontDisplayLastUserName -Value 1
26   Set-ItemProperty -Path $LastUser -Name LastLoggedOnUser -Value "$Username"
27   Set-ItemProperty -Path $LogonPath -Name LastUsedUsername -Value "$Username"
28   Write-Output "changed=yes comment='AutoLogon enabled.'"
29 }
```

Auto logon and screensavers is for the video capture that happens while a page is being loaded.

Disable Screensavers



```
1 $Path = 'HKCU:\Control Panel\Desktop'
2
3 Try {
4     $CurrentVal = Get-ItemProperty -Path $Path -Name ScreenSaveActive
5     Write-Output $CurrentVal
6 } Catch {
7     $CurrentVal = False
8 } Finally {
9     if ($CurrentVal.ScreenSaveActive -ne 0) {
10         Set-ItemProperty -Path $Path -Name ScreenSaveActive -Value 0
11         Write-Output "changed=yes comment='Screensaver Disabled.'"
12     } Else {
13         Write-Output "changed=no comment='Screensaver Already Disabled.'"
14     }
15 }
```

Auto logon and screensavers is for the video capture that happens while a page is being loaded.

Disable Shutdown Tracker



```
1 $Path = 'HKLM:\Software\Microsoft\Windows\CurrentVersion\Reliability'
2
3 Try {
4     $CurrentVal = Get-ItemProperty -Path $Path -Name ShutdownReasonUI
5     Write-Output $CurrentVal
6 } Catch {
7     $CurrentVal = False
8 } Finally {
9     if ($CurrentValShutdownReasonUI -ne 0) {
10         New-ItemProperty -Path $Path -Name ShutdownReasonUI -Value 0
11         Write-Output "changed=yes comment='Shutdown Tracker Disabled.'"
12     } Else {
13         Write-Output "changed=no comment='Shutdown Tracker Already Disabled.'"
14     }
15 }
```

Auto reboots are performed by the webpagetest agent when necessary for updates.

IE Browser Plugin



There isn't a way through the registry or PowerShell to authorize a browser plugin. You can only install them. They won't actually work until they are authorized when the browser first launches.

Mobile Devices



Android devices require USB debugging and a physical connection via USB to the testing node.

Our deployment was done on the Rackspace Cloud, so this wasn't possible.

The Ampersand (&) Bug

&

This character gave me hell for about an entire day. Care to guess how?

The Ampersand (&) Bug

```
/work/getwork.php?  
shards=1&location=Virginia_wptdriver&key=secret1  
23&software=wpt&version=2.15.0.183  
&ver=183&pc=COMPUTERNAME&dns=69.20.0.164-  
69.20.0.196&freedisk=17.036&GPU=0
```

What if I show you this line from the logs ...

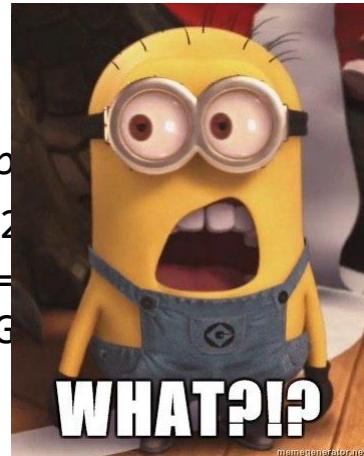
The Ampersand (&) Bug

```
/work/getwork.php?  
shards=1&location=Virginia_wptdriver&key=secr  
et123&software=wpt&version=2.15.0.183  
&ver=183&pc=COMPUTERNAME&dns=69.20.0.164-  
69.20.0.196&freedisk=17.036&GPU=0
```

WAT?

The Ampersand (&) Bug

```
/work/getwork.php?  
shards=1&location=Virginia_wp  
et123&software=wpt&version=2  
&ver=183&pc=COMPUTERNAME&dns=  
69.20.0.196&freedisk=17.036&G
```



I generate random secure passwords for my work. The one I generated to act as the secret key had an ampersand. Since work requests are passed using a PHP GET request, everything is passed in the URL of the request. The ampersand effectively cut my key in half, making the request fail.

SaltStack Challenges



salt-cloud windows bootstrapping
available windows modules
learning powershell

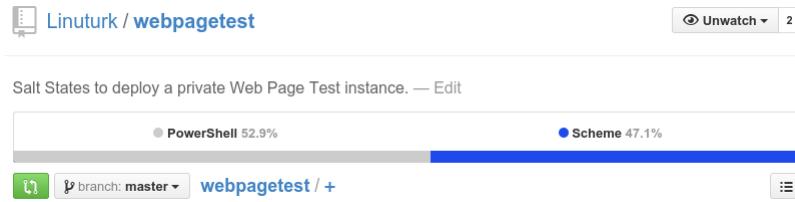
Windows Salt-Cloud

Bootstrapping Windows with salt-cloud today is a weird experience:

- [Use Password from OpenStack API - PR14748](#)
- [Fingerprint Mismatch, Map File Broke - #14593](#)
- [Discussion on new bootstrap method - #15984](#)

The PR was authored by me and fixed an immediate problem with passwords. Recently backported to 2014.7 and has been merged! YAY!
You still can't use a map file to provision multiple windows servers at the same time.
The Salt Master fingerprint gets corrupted on the minion, and it never connects properly.
There's a larger discussion going on about replacing winexe with another Windows remote execution tool.

Windows State Modules



Over half the code written for this project was PowerShell code versus the YAML (mis-identified as Scheme by Github).

A lot of this code could be replaced with a Windows Registry State Module. Hack Day project?

Learning PowerShell



```
PS C:\> Get-ChildItem 'MediaCenter\jMusic' -rec |  
    >>     where { -not $_.PSIsContainer -and $_.Extension -match '\wma|mp3' } |  
    >>     Measure-Object -property length -sum -min -max -ave  
  
Count : 1307  
Average : 100112% 09563887  
Sum : 9177097857  
Maximum : 22985267  
Minimum : 10000000  
Property : Length  
  
PS C:\> Get-UniObject CIM_BIOSElement | select biosv*, man*, ser* | Format-List  
  
BIOSVersion : (TOSCPL 6840000, Ver 1.00PARTIBL)  
Manufacturer : TOSHIBA  
SerialNumber : M0211IGH  
  
PS C:\> <(wmisearcher ID:  
    >> SELECT * FROM CIM_Job  
    >> WHERE Priority =  
    >> '0').get() | Format-Custom  
    >>  
class ManagementObject#root\cimv2\Min32_PrintJob  
<  
    Document = Monad Manifesto - Public  
    JobId = 6  
    JobState =  
    Owner = User  
    Priority = 02  
    Size = 027988  
    Name = Epson Stylus COLOR 740 ESC/P 2, 6  
  
PS C:\> $rssUrl = 'http://blogs.msdn.com/powershell/rss.aspx'  
PS C:\> $xml = [xml]::new()  
PS C:\> $xml.Load($rssUrl); $xml.SelectNodes('channel/item')  
title  
PS C:\> $xml.title[0].Content = "Getting in PowerShell 02"  
PowerShell 1 Presented At Microsoft  
MIS Talk: System Center Foundation Technologies  
  
PS C:\> $host.version.ToString().Insert(0, 'Windows PowerShell: ')  
Windows PowerShell: 1.0.0.0  
PS C:\>
```

Actually not that bad. Was a positive experience overall.
Lucky that I sit right next to 3 PowerShell developers at my office.

Try WebPageTest

Give WebPageTest a try using:



SALTSTACK

the public instance at webpagetest.org
Salt to spin up your own private architecture
Rackspace Orchestration Template (Heat)

We also have a Salt Master template!

Questions?

Submit Feedback!

Slides at: onitato.com

States and PowerShell at:

www.github.com/linuturk/webpagetest

