



## Update configuration for all redundant processes

[Browse files](#) master parsonsbots committed on Apr 141 parent [05bbaa9](#)commit [4bc9eb21afbab7349f86c6716986a19c7666fc9d](#) Showing 1 changed file with 49 additions and 44 deletions.

Unified

Split

▼ 93  dynamic\_reconfiguration.php 

```
33         return false;
34     }
35
36     - $gatewaysData =
37     json_decode(file_get_contents('/scripts/cache/gatewaysData
38     '), true);
39
40     - if (empty($gatewaysData['data']
41     ['proxies'])) {
42         return false;
43     }
44
45     $overridePorts = array(
46         'http' => $this->processes['http']
47     );
48
49     - $firewallRules = $this-
50     >_configureFirewallRules($gatewaysData, false,
51     $overridePorts);
52
53     $this->_applyFirewallRules($firewallRules,
54     'elastic');
55     }
56
57     }
58
59     $gatewaysJsonData = shell_exec("curl " .
60     $this->api . " --connect-timeout 30");
61
62     - $gatewaysData =
63     json_decode($gatewaysJsonData, true);
64
65
66     // Log API error timestamp
67     if (
68
69     - empty($gatewaysData['data']) ||
70
71
72     !is_dir('/etc/squid3')
73
74     ) {
75
76     - file_put_contents('/scripts/errors/api-error-' . time(),
77     $gatewaysData);
78
79
80
81     unlink('/scripts/pid/reconfigure.pid');
82
83     return false;
84     }
85
86
87     // Require Squid and sysctl configurations
88     from API
89
90     if (
91
92     - empty($gatewaysData['data']
93     ['squid_conf']) ||
```

```
33         return false;
34     }
35
36     + $this->gatewaysData =
37     json_decode(file_get_contents('/scripts/cache/gatewaysData
38     '), true);
39
40     + if (empty($this->gatewaysData['data']
41     ['proxies'])) {
42         return false;
43     }
44
45     $overridePorts = array(
46         'http' => $this->processes['http']
47     );
48
49     + $firewallRules = $this-
50     >_configureFirewallRules(false, $overridePorts);
51
52     $this->_applyFirewallRules($firewallRules,
53     'elastic');
54     }
55
56     }
57
58     $gatewaysJsonData = shell_exec("curl " .
59     $this->api . " --connect-timeout 30");
60
61     + $this->gatewaysData =
62     json_decode($gatewaysJsonData, true);
63
64
65     // Log API error timestamp
66     if (
67
68     + empty($this->gatewaysData['data'])
69     ||
70
71
72     !is_dir('/etc/squid3')
73
74     ) {
75
76     + file_put_contents('/scripts/errors/api-error-' . time(),
77     $this->gatewaysData);
78
79
80
81     unlink('/scripts/pid/reconfigure.pid');
82
83     return false;
84     }
85
86
87     // Require Squid and sysctl configurations
88     from API
89
90     if (
91
92     + empty($this->gatewaysData['data']
93     ['squid_conf']) ||
```

```

109 -         empty($gatewaysData['data']
110 -         empty($gatewaysData['data']
111 -         ['sysctl_conf'])
112 -         ) {
113 -         unlink('/scripts/pid/reconfigure.pid');
114 -         return false;
115 -         // Create new reconfiguration process ID
116 -         file_put_contents('/scripts/pid/reconfigure.pid', time());
117 -         // Apply primary Squid config
118 -         file_put_contents('/etc/squid3/squid.conf',
119 -         $gatewaysData['data']['squid_conf']);
120 -         // TODO: Apply global redundant squid
121 -         configuration to all redundant processes
122 -         file_put_contents('/etc/squid3/squid-
123 -         redundant.conf', $gatewaysData['data']
124 -         ['squid_redundant_conf']);
125 -         // Save and apply sysctl settings
126 -         file_put_contents('/etc/sysctl.conf',
127 -         $gatewaysData['data']['sysctl_conf']);
128 -         shell_exec('sysctl -p');
129 -         // Don't run reconfiguration if there
130 -         aren't any ACLs to apply to proxy IPs
131 -         $proxies = $gatewaysData['data']
132 -         ['proxies'];
133 -         if (empty($proxies[0])) {
134 -         unlink('/scripts/pid/reconfigure.pid');
135 -         shell_exec('rm -rf /etc/squid3/users/');
136 -         shell_exec('mkdir -m 777
137 -         /etc/squid3/users/');
138 -         if (!empty($gatewaysData['data']
139 -         ['files'])) {
140 -         foreach ($gatewaysData['data']
141 -         ['files'] as $file) {
142 -         shell_exec('mkdir -m 777 '
143 -         . str_replace(array('s.txt', 'd.txt'), '',
144 -         $file['path']));
145 -         shell_exec('touch ' .
146 -         $file['path']);
147 -         file_put_contents($file['path'], $file['contents']);
148 -         }
149 -         }
150 -         }
151 -         $firewallRules = $this->
152 -         _configureFirewallRules($gatewaysData, true);
153 -         // Save Squid ACLs from API to file
154 -         shell_exec('rm
155 -         /etc/squid3/proxy_ip_acl.conf');
156 -         shell_exec('touch
157 -         /etc/squid3/proxy_ip_acl.conf');
158 -

```

```

109 +         empty($this->gatewaysData['data']
110 +         ['sysctl_conf'])
111 +         ) {
112 +         unlink('/scripts/pid/reconfigure.pid');
113 +         return false;
114 +         // Create new reconfiguration process ID
115 +         file_put_contents('/scripts/pid/reconfigure.pid', time());
116 +         // Save and apply sysctl settings
117 +         file_put_contents('/etc/sysctl.conf',
118 +         $this->gatewaysData['data']['sysctl_conf']);
119 +         shell_exec('sysctl -p');
120 +         // Don't run reconfiguration if there
121 +         aren't any ACLs to apply to proxy IPs
122 +         $proxies = $this->gatewaysData['data']
123 +         ['proxies'];
124 +         if (empty($proxies[0])) {
125 +         unlink('/scripts/pid/reconfigure.pid');
126 +         shell_exec('rm -rf /etc/squid3/users/');
127 +         shell_exec('mkdir -m 777
128 +         /etc/squid3/users/');
129 +         if (!empty($this->gatewaysData['data']
130 +         ['files'])) {
131 +         foreach ($this->
132 +         gatewaysData['data']['files'] as $file) {
133 +         shell_exec('mkdir -m 777 '
134 +         . str_replace(array('s.txt', 'd.txt'), '',
135 +         $file['path']));
136 +         shell_exec('touch ' .
137 +         $file['path']);
138 +         file_put_contents($file['path'], $file['contents']);
139 +         }
140 +         }
141 +         }
142 +         $firewallRules = $this->
143 +         _configureFirewallRules(true);
144 +         // Save Squid ACLs from API to file
145 +         shell_exec('rm
146 +         /etc/squid3/proxy_ip_acl.conf');
147 +         shell_exec('touch
148 +         /etc/squid3/proxy_ip_acl.conf');
149 +

```

```

158 file_put_contents('/etc/squid3/proxy_ip_acl.conf',
159 implode("\n", $gatewaysData['data']['acls']));
160 // Set proxy usernames and passwords using
161 httpasswd and basic_ncsa_auth for security
162 shell_exec('htpasswd -cb
163 /etc/squid3/passwords default default');
164 shell_exec('htpasswd -D
165 /etc/squid3/passwords default');
166
167 if (!empty($gatewaysData['data']
168 ['users'])) {
169     foreach ($gatewaysData['data']
170 ['users'] as $username => $password) {
171
172         shell_exec('htpasswd -b
173 /etc/squid3/passwords ' . $username . ' ' . $password);
174     }
175 }
176
177 // Reconfigure existing SOCKS processes
178 first once redundant firewall is applied
179 if (
180     !empty($gatewaysData['data']
181 ['socks']) &&
182     !empty($this->processes['socks']
183 [0])
184 ) {
185     // SOCKS config and ACLs are in
186     the same file (TODO: move squid configuration and ACLs to
187     same file for consistency)
188     file_put_contents('/usr/local/etc/3proxy/3proxy.cfg',
189 $gatewaysData['data']['socks']);
190
191     // Reconfigure main SOCKS instance
192     $this->reconfigure(
193         'socks',
194         'service 3proxy start',
195         '3proxy.cfg',
196
197         '/usr/local/etc/3proxy/3proxy.pid',
198         5,
199         15
200
201     );
202 }
203
204 'squid3',
205 '/var/run/squid3.pid',
206 0,
207 65
208
209 );
210
211 $redundantProcesses = $this->

```

```

151 file_put_contents('/etc/squid3/proxy_ip_acl.conf',
152 implode("\n", $this->gatewaysData['data']['acls']));
153 // Set proxy usernames and passwords using
154 httpasswd and basic_ncsa_auth for security
155 shell_exec('htpasswd -cb
156 /etc/squid3/passwords default default');
157 shell_exec('htpasswd -D
158 /etc/squid3/passwords default');
159
160 if (!empty($this->gatewaysData['data']
161 ['users'])) {
162     foreach ($this->gatewaysData['data']
163 ['users'] as $username => $password)
164 {
165         shell_exec('htpasswd -b
166 /etc/squid3/passwords ' . $username . ' ' . $password);
167     }
168 }
169
170 // Reconfigure existing SOCKS processes
171 first once redundant firewall is applied
172 if (
173     !empty($this->gatewaysData['data']
174 ['socks']) &&
175     !empty($this->processes['socks']
176 [0])
177 ) {
178     // Reconfigure main SOCKS instance
179     $this->reconfigure(
180         'socks',
181         'service 3proxy start',
182         '3proxy.cfg',
183
184         '/usr/local/etc/3proxy/3proxy.pid',
185         5,
186         15,
187         +
188         +
189         '/usr/local/etc/3proxy/3proxy.cfg',
190         +
191         $this->
192         >gatewaysData['data']['socks']
193     );
194 }
195
196 'squid3',
197 '/var/run/squid3.pid',
198 0,
199 65,
200 +
201 +
202 '/etc/squid3/squid.conf',
203 +
204     str_replace(['pid'], 'pid_filename
205 /var/run/squid3.pid', str_replace(['ports'], 'http_port '
206 . implode("\n" . 'http_port ', $this->processes['http']
207 [0]), $this->gatewaysData['data']['squid_conf']))
208 );
209
210 $redundantProcesses = $this->

```

```

>processes['http']; // TODO: chunk and reconfigure HTTP
and SOCKS processes simultaneously
239         )
240     );
241
242     -         $firewallRules = $this-
>_configureFirewallRules($gatewaysData, false,
$overridePorts);
243         $this->
>_applyFirewallRules($firewallRules, 'elastic');
244
245         foreach ($redundantProcessRange as
$redundantProcess) {
246             // Reconfigure redundant
HTTP instances
247             $this->_reconfigure(
248                 'http',
249                 'squid3-redundant'
. $redundantProcess . ' start -f /etc/squid3/squid-
redundant' . $redundantProcess . '.conf',
250                 'squid3-redundant'
. $redundantProcess,
251                 '/var/run/squid-
redundant' . $redundantProcess . '.pid',
252                 0,
253                 0
-
254             );
255         }
256
261         $overridePorts = array(
262             'http' => $this->processes['http']
263         );
264         -         $firewallRules = $this-
>_configureFirewallRules($gatewaysData, false,
$overridePorts);
265         $this->_applyFirewallRules($firewallRules,
'elastic');
266
267         -         if (!empty($gatewaysData['data']['socks-
redundant'])) {
268             -
file_put_contents('/usr/local/etc/3proxy/3proxy.cfg',
$gatewaysData['data']['socks']);
269         -
270             // Reconfigure redundant SOCKS
instance
271             $this->_reconfigure(
272                 'socks',
273                 'service 3proxy-redundant
start',
274                 '3proxy-redundant.cfg',
275                 '/usr/local/etc/3proxy/3proxy-redundant.pid',

```

```

>processes['http']; // TODO: chunk and reconfigure HTTP
and SOCKS processes simultaneously
233         )
234     );
235
236     +         $firewallRules = $this-
>_configureFirewallRules(false, $overridePorts);
237         $this->
>_applyFirewallRules($firewallRules, 'elastic');
238
239         foreach ($redundantProcessRange as
$redundantProcess) {
240             // Reconfigure redundant
HTTP instances
241             $this->_reconfigure(
242                 'http',
243                 'squid3-redundant'
+
. $redundantProcess . ' start -f ' . $configurationFile,
244                 'squid3-redundant'
. $redundantProcess,
245                 '/var/run/squid-
redundant' . $redundantProcess . '.pid',
246                 0,
247                 0,
248                 +
'/etc/squid3/squid-redundant' . $redundantProcess .
'.conf',
249                 +
str_replace(['pid'], 'pid_filename /var/run/squid-
redundant' . $redundantProcess . '.pid',
str_replace(['ports'], 'http_port ' . implode("\n" .
'http_port ', $this->processes['http']
[$redundantProcess]), $this->gatewaysData['data']
['squid_conf']))
250             );
251         }
252
257         $overridePorts = array(
258             'http' => $this->processes['http']
259         );
260         +         $firewallRules = $this-
>_configureFirewallRules(false, $overridePorts);
261         $this->_applyFirewallRules($firewallRules,
'elastic');
262
263         +         if (!empty($this->gatewaysData['data']
['socks-redundant'])) {
264             // Reconfigure redundant SOCKS
instance
265             $this->_reconfigure(
266                 'socks',
267                 'service 3proxy-redundant
start',
268                 '3proxy-redundant.cfg',
269                 '/usr/local/etc/3proxy/3proxy-redundant.pid',

```

```

276         10,
277 -         0,

278     );
279     }
280
281     /**
282     * Configure firewall rules
283     *
284     - * @param array $gatewaysData Data from gateway API
285     response
286     * @param boolean $redundant Route to redundant process
287     ports only if true, all ports if false
288     * @param array $ports Override default base ports
289     *
290     * @return array $rules Firewall rules
291     */
292 -     protected function
293     _configureFirewallRules($gatewaysData, $redundant = false,
294     $overridePorts = array()) {
295         // Begin input filter rules
296         $rules = array(
297             '*filter',
298
299             // Apply custom firewall rules from API
300             (e.g. disabling ports for specific destinations)
301             if (
302 -                 !empty($gatewaysData['data']
303                 ['firewall_filter']) &&
304 -                 is_array($gatewaysData['data']
305                 ['firewall_filter'])
306             ) {
307 -                 foreach ($gatewaysData['data']
308                 ['firewall_filter'] as $rule) {
309                     $rules[] = $rule;
310                 }
311             }
312
313             // Remove ports from
314             routing list that are refusing connections
315             foreach ($ports as $key =>
316             $port) {
317 -                 if ($this->
318                 _checkPort($gatewaysData['data']['proxies'][0], $port,
319                 $protocol) === false) {
320
321                     unset($ports[$key]);
322                 }
323             }
324
325     * @param string $processId Full path to process ID
326     * @param integer $delayStart Delay at beginning of
327     reconfiguration
328     * @param integer $delayEnd Delay at end of
329     reconfiguration
330
331     *
332     * @return

```

```

270         10,
271 +         0,
272 +
273 + '/usr/local/etc/3proxy/3proxy-redundant.cfg',
274 + $this->
275 + >gatewaysData['data']['socks-redundant']
276 + );
277     }
278
279     /**
280     * Configure firewall rules
281     *
282     + * @param boolean $redundant Route to redundant process
283     ports only if true, all ports if false
284     * @param array $ports Override default base ports
285     *
286     * @return array $rules Firewall rules
287     */
288 +     protected function
289     _configureFirewallRules($redundant = false, $overridePorts
290     = array()) {
291         // Begin input filter rules
292         $rules = array(
293             '*filter',
294
295             // Apply custom firewall rules from API
296             (e.g. disabling ports for specific destinations)
297             if (
298 +                 !empty($this->gatewaysData['data']
299                 ['firewall_filter']) &&
300 +                 is_array($this->
301                 >gatewaysData['data']['firewall_filter'])
302             ) {
303 +                 foreach ($this->
304                 >gatewaysData['data']['firewall_filter'] as $rule) {
305                     $rules[] = $rule;
306                 }
307             }
308
309             // Remove ports from
310             routing list that are refusing connections
311             foreach ($ports as $key =>
312             $port) {
313 +                 if ($this->
314                 _checkPort($this->gatewaysData['data']['proxies'][0],
315                 $port, $protocol) === false) {
316
317                     unset($ports[$key]);
318                 }
319             }
320
321     * @param string $processId Full path to process ID
322     * @param integer $delayStart Delay at beginning of
323     reconfiguration
324     * @param integer $delayEnd Delay at end of
325     reconfiguration
326
327     + * @param string $configurationFile Full path to process
328     configuration file
329     + * @param string $configurationData Process
330     configuration data
331
332     *
333     * @return

```

```
520 */
521 - protected function _reconfigure($protocol,
    $startCommand, $processName, $processId, $delayStart = 0,
    $delayEnd = 0) {
522     if ($delayStart) {
523         sleep($delayStart);
524     }
541     shell_exec('/scripts/./' .
    $protocol . '.sh');
542 }
543
544 // Process recovery
545 unlink($processId);
546 sleep(1);
```

```
517 */
518 + protected function _reconfigure($protocol,
    $startCommand, $processName, $processId, $delayStart = 0,
    $delayEnd = 0, $configurationFile, $configurationData) {
519     if ($delayStart) {
520         sleep($delayStart);
521     }
538     shell_exec('/scripts/./' .
    $protocol . '.sh');
539 }
540
541 + // Update configuration file for process
542 + if (
543 +     !empty($configurationFile) &&
544 +     !empty($configurationData)
545 + ) {
546 +     file_put_contents($configurationFile, $configurationData);
547 + }
548 +
549 // Process recovery
550 unlink($processId);
551 sleep(1);
```

0 comments on commit `4bc9eb2`