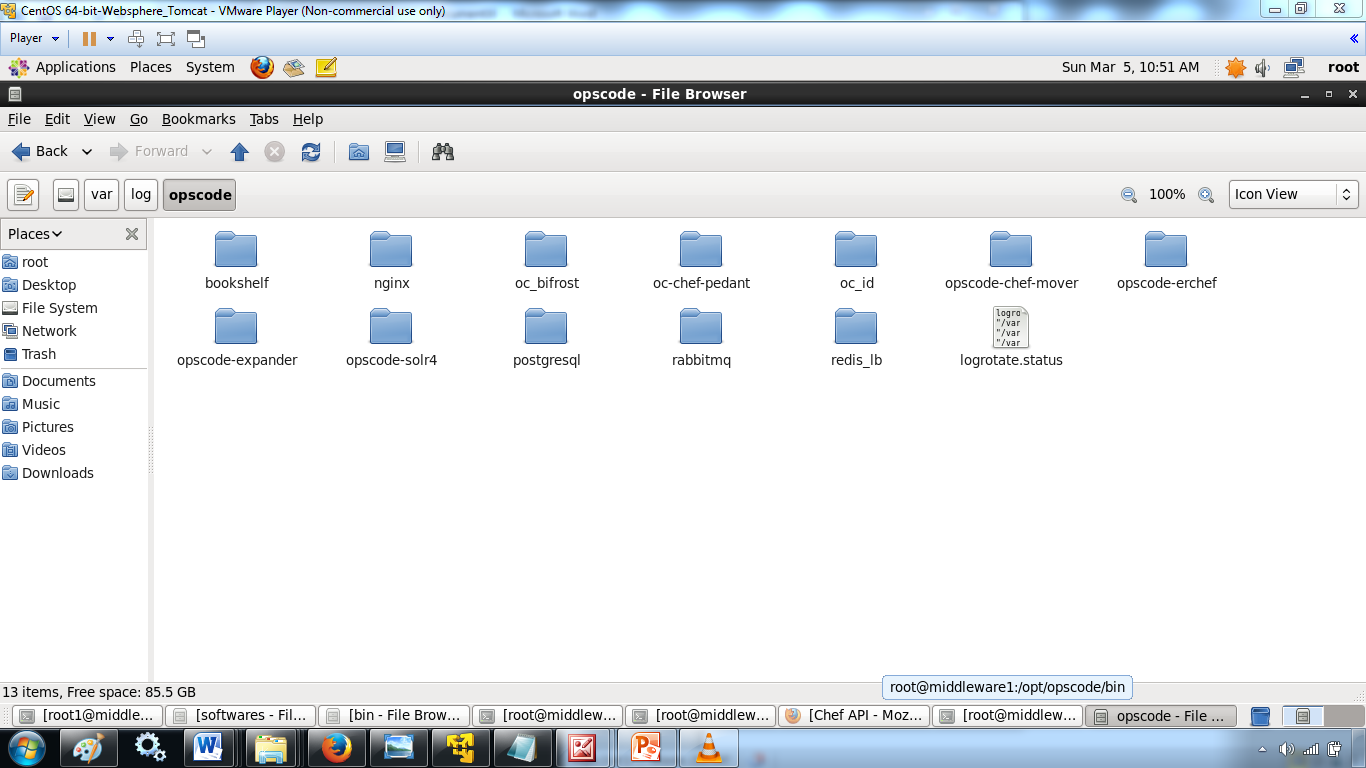
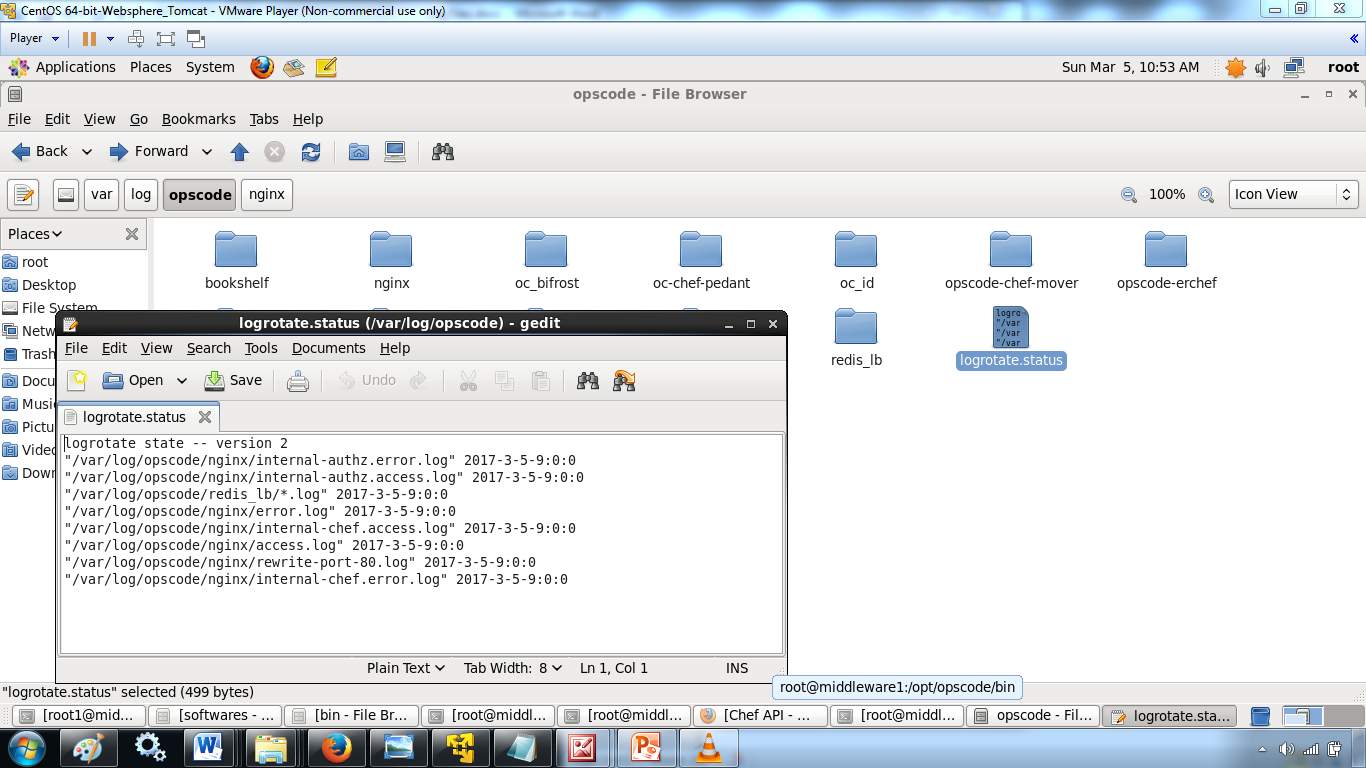
**Log Files**

All logs generated by the Chef server can be found in /var/log/opscode. Each service enabled on the system also has a sub-directory in which service-specific logs are located, typically found in /var/log/opscode/service\_name.

/var/log/opscode





**View Log Files**

The Chef server has built-in support for easily tailing the logs that are generated. To view all the logs being generated on the Chef server, enter the following command:

$ chef-server-ctl tail

To view logs for a specific service:

$ chef-server-ctl tail SERVICENAME

where SERVICENAME should be replaced with name of the service for which log files will be viewed.

Another way to view log files is to use the system utility tail:

$ tail -50f /var/log/chef-server/opscode-chef/current

**tail Log Files**

The tail subcommand is used to follow all of the Chef server logs for all services. This command can also be run for an individual service by specifying the name of the service in the command.

This subcommand has the following syntax:

$ chef-server-ctl tail SERVICE\_NAME

where SERVICE\_NAME represents the name of any service that is listed after running the service-list subcommand.

Another common approach to tailing the log files for a service is to use the system utility tail. For example:

$ tail -50f /var/log/opscode/opscode-chef/current

**Supervisor**

Supervisor logs are created and managed directly by the service supervisor, and are automatically rotated when a the current log file reaches 1,000,000 bytes. 10 log files are kept. The latest supervisor log is always located in /var/log/chef-server/service\_name/current and rotated logs have a filename starting with @ followed by a precise tai64n timestamp based on when the file was rotated.

Supervisor logs are available for the following services:

* bifrost
* bookshelf
* nginx
* opscode-erchef
* opscode-expander
* opscode-expander-reindexer
* opscode-solr4
* postgresql
* rabbitmq
* redis

**nginx, access**

Nginx is an important entry point for data on the Chef server, which means that debugging efforts frequently start with analyzing the **nginx** service’s access.log file. This log contains every HTTP request made to the front-end machine and can be very useful when investigating request rates and usage patterns. The following is an example log entry:

175.185.9.6 - - [12/Jul/2013:15:56:54 +0000] "GET

/organizations/exampleorg/data/firewall/nova\_api HTTP/1.1" 200

"0.850" 452 "-" "Chef Client/0.10.2 (ruby-1.8.7-p302; ohai-0.6.4;

x86\_64-linux; +http://opscode.com)" "127.0.0.1:9460" "200"

"0.849" "0.10.2" "version=1.0" "some\_node.example.com"

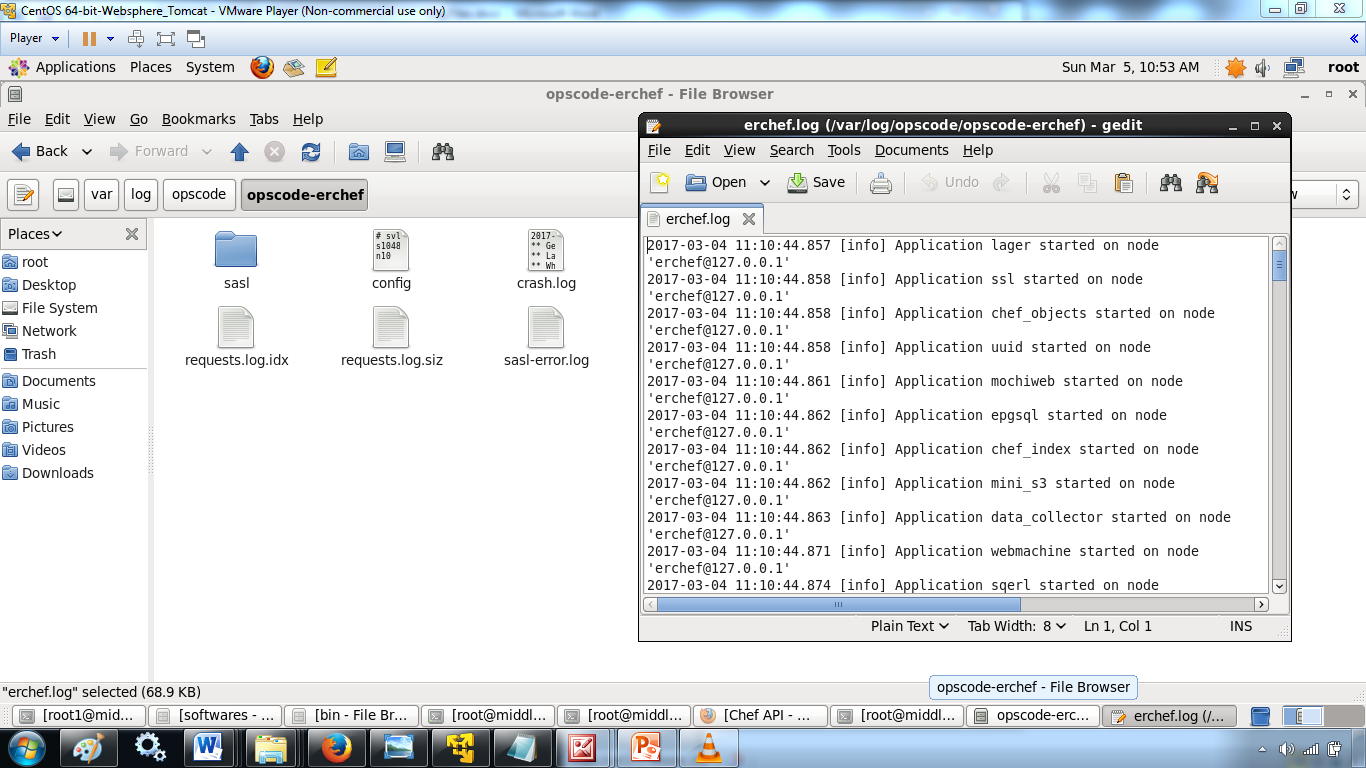
"2013-07-12T15:56:40Z" "2jmj7l5rSw0yVb/vlWAYkK/YBwk=" 985

where important fields in this log include:

* The HTTP status code (200)
* The IP address of the requesting client (175.185.9.6)
* The timestamp ([12/Jul/2013:15:56:54 +0000])
* The total request time ("0.850")
* The request method (GET)
* The request URL (/organizations/exampleorg/data/firewall/nova\_api)

**opscode-erchef, current**

The **opscode-erchef** service’s current.log file contains a history of stack traces from major application crashes.



**opscode-erchef, erchef**

The **opscode-erchef** service’s erchef.log file contains a history of API requests that have been processed by Erchef. These logs can be rotated quickly, therefore it is generally best to sort them by date, and then find the most recently updated log file:

$ ls -lrt /var/log/opscode/opscode-erchef/erchef.log.\*

The following is an example log entry:

2013-08-06T08:54:32Z erchef@127.0.0.1 INFO org\_name=srwjedoqqoypgmvafmoi; req\_id=g3IAA2QAEGVyY2hlZkAx

where important fields in this log include:

* The HTTP method (POST)
* The HTTP path (/organizations/srwjedoqqoypgmvafmoi/environments)
* The message ({created,<<"\_default">>})
* The organization name (org\_name=srwjedoqqoypgmvafmoi)
* The timestamp (2013-08-06T08:54:32Z)
* The name of the user and/or chef-client which made the request (pivotal)

In addition, the log file may contain additional entries that detail the amounts of time spent interacting with other services:

* rdbms\_time (the time spent talking to the **postgresql** service)
* req\_time (the request time)
* solr\_time (the time spent talking to the **opscode-solr** service)

**Application**

Application logs are created by the services directly, and may require log rotation policies to be applied based on organizational goals and the platform(s) on which the services are running.

**nginx**

The nginx service creates both supervisor and administrator logs. The administrator logs contain both access and error logs for each virtual host utilized by the Chef server. Each of the following logs require external log rotation.

| **Logs** | **Description** |
| --- | --- |
| /var/log/opscode/nginx/access.log | The Web UI and API HTTP access logs. |
| /var/log/opscode/nginx/error.log | The Web UI and API HTTP error logs. |
| /var/log/opscode/nginx/internal-account.access.log | The opscode-account internal load-balancer access logs. |
| /var/log/opscode/nginx/internal-account.error.log | The opscode-account internal load-balancer error logs. |
| /var/log/opscode/nginx/internal-authz.access.log | The opscode-authz internal load-balancer access logs. |
| /var/log/opscode/nginx/internal-authz.error.log | The opscode-authz internal load-balancer error logs. |
| /var/log/opscode/nginx/internal-chef.access.log | The opscode-chef and opscode-erchef internal load-balancer access logs. |
| /var/log/opscode/nginx/internal-chef.error.log | The opscode-chef and opscode-erchef internal load-balancer error logs. |
| /var/log/opscode/nginx/nagios.access.log | The nagios access logs. |
| /var/log/opscode/nginx/nagios.error.log | The nagios error logs. |
| /var/log/opscode/nginx/rewrite-port-80.log | The rewrite logs for traffic that uses HTTP instead of HTTPS. |

To follow the logs for the service:

$ chef-server-ctl tail nginx

**Read Log Files**

The **nginx** access log format is as follows:

log\_format opscode '$remote\_addr - $remote\_user [$time\_local] '

'"$request" $status "$request\_time" $body\_bytes\_sent '

'"$http\_referrer" "$http\_user\_agent" "$upstream\_addr" '

'"$upstream\_status" "$upstream\_response\_time" "$http\_x\_chef\_version" '

'"$http\_x\_ops\_sign" "$http\_x\_ops\_userid" "$http\_x\_ops\_timestamp" '

'"$http\_x\_ops\_content\_hash" $request\_length';

A sample log line:

192.168.4.1 - - [17/Feb/2012:16:02:42 -0800]

"GET /organizations/nginx/cookbooks HTTP/1.1" 200

"0.346" 12 "-"

"Chef Knife/0.10.4 (ruby-1.9.3-p0;

ohai-0.6.10;

x86\_64-darwin11.2.0;

+http://opscode.com

)"

"127.0.0.1:9460" "200" "0.339" "0.10.4"

"version=1.0" "adam" "2012-02-18T00:02:42Z"

"2jmj7l5rSw0yVb/vlWAYkK/YBwk=" 871

Field descriptions:

| **Field** | **Description** |
| --- | --- |
| $remote\_addr | The IP address of the client who made this request. |
| $remote\_user | The HTTP basic auth user name of this request. |
| $time\_local | The local time of the request. |
| $request | The HTTP request. |
| $status | The HTTP status code. |
| $request\_time | The time it took to service the request. |
| $body\_bytes\_sent | The number of bytes in the HTTP response body. |
| $http\_referrer | The HTTP referrer. |
| $http\_user\_agent | The user agent of the requesting client. |
| $upstream\_addr | The upstream reverse proxy used to service this request. |
| $upstream\_status | The upstream reverse proxy response status code. |
| $upstream\_response\_time | The upstream reverse proxy response time. |
| $http\_x\_chef\_version | The version of Chef used to make this request. |
| $http\_x\_ops\_sign | The version of the authentication protocol. |
| $http\_x\_ops\_userid | The client name that was used to sign this request. |
| $http\_x\_ops\_timestamp | The timestamp from when this request was signed. |
| $http\_x\_ops\_content\_hash | The hash of the contents of this request. |
| $request\_length | The length of this request. |