無料電子ブック

学習

varnish

Free unaffiliated eBook created from
Stack Overflow contributors.

#varnish

1: ワニスをめる

このセクションでは、ワニスのと、それをするについてします。

またワニスのきなについてもし、するトピックにリンクするがあります。ワニスのドキュメンテーションはしいので、これらのトピックのバージョンをするがあります。

バージョン

バージョン	
5.1.2	2017-04-07
5.1.1	2017-03-16
5.0	2016-09-15
4.1.5	2016-02-09
4.0.4	2016-11-30
3.0.7	2015-03-23

Examples

インストールまたはセットアップ

は、さまざまなLinuxディストリビューションにバージョンのVarnishをするです。

CentOS 7

```
curl -s https://packagecloud.io/install/repositories/varnishcache/varnish5/script.rpm.sh |
sudo bash
```

Ubuntu

```
apt-get install apt-transport-https
curl https://repo.varnish-cache.org/GPG-key.txt | apt-key add -
echo "deb https://repo.varnish-cache.org/ubuntu/ trusty varnish-4.1" \
  >> /etc/apt/sources.list.d/varnish-cache.list
apt-get update
apt-get install varnish
```

Debian

```
apt-get install apt-transport-https
curl https://repo.varnish-cache.org/GPG-key.txt | apt-key add -
echo "deb https://repo.varnish-cache.org/debian/ jessie varnish-4.1"\
  >> /etc/apt/sources.list.d/varnish-cache.list
apt-get update
apt-get install varnish
```

ワニスVCL

Varnishは、Varnish Configuration LanguageVCLをしてHTTPをおよびします。のVCLのスニペットは、/ imagesサブディレクトリへのからCookieをします。

```
sub vcl_recv {
    if (req.url ~ "^/images") {
        unset req.http.cookie;
    }
}
```

オンラインでワニスをめるをむ https://riptutorial.com/ja/varnish/topic/4705/ワニスをめる

2: vmodの

き

あなたのプラットフォームのバイナリがにな、ずしもvmodをコンパイルするはありません。CentOS 6と7ので、IngvarのCOPRビルドをして、Varnish Softwareのモジュールのコレクションをインストールすることができます https ://copr.fedorainfracloud.org/coprs/ingvar/varnish51/

Examples

vmodのコンパイルとインストール

vmodのインストールには、ファイルをむインストールされたバージョンのVarnish Cacheがです。は、ワニスのドキュメントにされています。

ソースコードはオートツールでされています

```
sudo apt-get install libvarnishapi-dev || sudo yum install varnish-libs-devel
./bootstrap   # If running from git.
./configure
make
make check   # optional
sudo make install
```

オンラインでvmodのをむ https://riptutorial.com/ja/varnish/topic/9669/vmodの

3: モニタニス

き

`varnishstat`をして、のVarnishインスタンスのメトリックをします。はインスト―ルによってなります。 `varnishstat -1`をすると、すべてのメトリックがな`grep` -ableでされます。

そののユ―ティリティは、ワニスののステ―タスとログをるためにできます `varnishtop` 、 `varnishlog`など

Examples

クライアントメトリック - トラフィック

クライアントメトリックは、クライアントとワニスキャッシュのトラフィックをカバ―します。

- sess_conn - の。
- client_req - クライアントの。
- sess_dropped - なキュ―のためにがされました。

`sess_conn`と`client_req`をしてトラフィックをする - またはしているか、スパイクしているかなど。のはをしているがあります。

`sess_dropped`をして、キャッシュがセッションをしているかどうかをします。もしそうなら、`thread_pool_max`をやすがあります。

```
varnishstat -1 | grep "sess_conn\|client_req \|sess_dropped"
MAIN.sess_conn              62449574         3.38 Sessions accepted
MAIN.client_req            184697229         9.99 Good client requests received
MAIN.sess_dropped                  0         0.00 Sessions dropped for thread
```

キャッシュのパフォ―マンス

おそらくもなパフォ―マンスメトリックはヒットです。

ワニスはのようなをル―ティングします。

- キャッシュなリクエストであるハッシュ。これは、キャッシュのにじて`hit`たり、 `miss hit`たりするがあります。
- キャッシュなリクエストではないHitpass。

`miss`と`hitpass`をむハッシュは、サ―バ―のバックエンドからフェッチされてされます。 `hit`したハッシュはキャッシュからされます。

するメトリック

- cache_hit - キャッシュでヒットしたハッシュの。
- cache_miss - キャッシュのミスをうハッシュの。
- cache_hitpass - のヒットパスの。

```
varnishstat -1 | grep "cache_hit \|cache_miss \|cache_hitpass"
MAIN.cache_hit              99032838         5.36 Cache hits
MAIN.cache_hitpass                 0         0.00 Cache hits for pass
MAIN.cache_miss             42484195         2.30 Cache misses
```

のヒットをのようにします。

```
cache_hit / (cache_hit + cache_miss)
```

このでは、ヒットは0.7または70です。あなたはこれをなりくしたいとっています。 70はまともなです。メモリをやしてvclをカスタマイズすることで、ヒットをさせることができます。また、ヒットのきなをします。

キャッシュされたオブジェクトの

キャッシュされたオブジェクトをして、それらがれになると「である」かどうかをします。

- n_expired - れのオブジェクトの。
- n_lru_nuked - にされたオブジェクトです。きがないためにキャッシュからりかれたされたオブジェクトの。

```
varnishstat -1 | grep "n_expired\|n_lru_nuked"
MAIN.n_expired              42220159         .   Number of expired objects
MAIN.n_lru_nuked              264005         .   Number of LRU nuked objects
```

ここでたいのは、スペ―スがりないためにキャッシュがかつにオブジェクトをしすがえているだけではなく、 `n_lru_nuked`です。キャッシュサイズをきくするがあります。

`n_expired`はあなたのアプリケ―ションに`n_expired`です。よりいきるはこのをさせるが、ではしばしばオブジェクトをしない。また、キャッシュにはさらにきなサイズがながあります。

スレッドの

あなたは、ワニスキャッシュをるためにいくつかのスレッドメトリックをするがあります。 OSのリソ―スがしているか、うまくしていますか。

- threads - すべてのプ―ルのスレッドの。
- threads_created - されたスレッドの。
- threads_failed - Varnishがスレッドのにした。
- threads_limited - ワニスがになってからスレッドをしないようにされた。

- thread_queue_len - のキュ―のさ。スレッドをっているの。
- sess_queued - がキュ―にれられなければならないスレッドがない。

```
varnishstat -1 | grep "threads\|thread_queue_len\|sess_queued"
MAIN.threads                    100          .   Total number of threads
MAIN.threads_limited              1       0.00 Threads hit max
MAIN.threads_created           3715       0.00 Threads created
MAIN.threads_destroyed         3615       0.00 Threads destroyed
MAIN.threads_failed               0       0.00 Thread creation failed
MAIN.thread_queue_len             0          .   Length of session queue
MAIN.sess_queued               2505       0.00 Sessions queued for thread
```

`thread_queue_len`が0でないは、Varnishにリソ―スがなく、をキュ―にれめたことをします。これにより、これらののパフォ―マンスがします。をべるがあります。

`threads_failed`もて`threads_failed`。これがえると、サ―バ―にらかのリソ―スがしていることをします。`threads_limited`をやすと、サ―バ―`thread_pool_max`をやすがじるがあります。

バックエンドメトリックの

ワニスとバックエンドののコミュニケ―ションをするくのメトリクスがあります。

ここでもなはのとおりです。

- backend_busy - バックエンドによってされたhttp 5xxステ―タスの。VCLをすると、ワニスがのバックエンドをすようにできます。
- backend_fail - Varnishがバックエンドにできなかった。これにはいくつかのがありますTCPなし、のバイトまでのい、いバイトの。この、バックエンドはではありません。
- backend_unhealthy - Varnishがバックエンドに「ping」できなかったHTTP 200でしなかった

```
varnishstat -1 | grep "backend_"
MAIN.backend_conn           86913481         4.70 Backend conn. success
MAIN.backend_unhealthy             0         0.00 Backend conn. not attempted
MAIN.backend_busy                  0         0.00 Backend conn. too many
MAIN.backend_fail                  7         0.00 Backend conn. failures
MAIN.backend_reuse                 0         0.00 Backend conn. reuses
MAIN.backend_toolate               0         0.00 Backend conn. was closed
MAIN.backend_recycle               0         0.00 Backend conn. recycles
MAIN.backend_retry                 0         0.00 Backend conn. retry
MAIN.backend_req            86961073         4.70 Backend requests made
```

オンラインでモニタニスをむ https://riptutorial.com/ja/varnish/topic/9072/モニタニス

4: VCL

き

ビルトインVCLには、Varnishにまれ、 にされるプロシ―ジャがまれています。

のサイトにしたロジックをすることで、ユ―ザ―のVCLをすることができます。たとえば、POSTのキャッシュをスキップしたり、Cookieヘッダ―またはヘッダ―があるはキャッシュをスキップしたりします。

みみロジックのいくつかがない、ユ―ザ―は、みみVCLロジックがましくないプロシ―ジャからの`return()`コ―ルをできます。

Examples

ワニス3.0

```
/*-
 *
 * The default VCL code.
 *
 * NB! You do NOT need to copy & paste all of these functions into your
 * own vcl code, if you do not provide a definition of one of these
 * functions, the compiler will automatically fall back to the default
 * code from this file.
 *
 */

sub vcl_recv {
    if (req.restarts == 0) {
        if (req.http.x-forwarded-for) {
            set req.http.X-Forwarded-For =
                req.http.X-Forwarded-For + ", " + client.ip;
        } else {
            set req.http.X-Forwarded-For = client.ip;
        }
    }
    if (req.request != "GET" &&
      req.request != "HEAD" &&
      req.request != "PUT" &&
      req.request != "POST" &&
      req.request != "TRACE" &&
      req.request != "OPTIONS" &&
      req.request != "DELETE") {
        /* Non-RFC2616 or CONNECT which is weird. */
        return (pipe);
    }
    if (req.request != "GET" && req.request != "HEAD") {
        /* We only deal with GET and HEAD by default */
        return (pass);
    }
    if (req.http.Authorization || req.http.Cookie) {
```

```
        /* Not cacheable by default */
        return (pass);
    }
    return (lookup);
}

sub vcl_pipe {
    # Note that only the first request to the backend will have
    # X-Forwarded-For set.  If you use X-Forwarded-For and want to
    # have it set for all requests, make sure to have:
    # set bereq.http.connection = "close";
    # here.  It is not set by default as it might break some broken web
    # applications, like IIS with NTLM authentication.
    return (pipe);
}

sub vcl_pass {
    return (pass);
}

sub vcl_hash {
    hash_data(req.url);
    if (req.http.host) {
        hash_data(req.http.host);
    } else {
        hash_data(server.ip);
    }
    return (hash);
}

sub vcl_hit {
    return (deliver);
}

sub vcl_miss {
    return (fetch);
}

sub vcl_fetch {
    if (beresp.ttl <= 0s ||
        beresp.http.Set-Cookie ||
        beresp.http.Vary == "*") {
                /*
                 * Mark as "Hit-For-Pass" for the next 2 minutes
                 */
                set beresp.ttl = 120 s;
                return (hit_for_pass);
    }
    return (deliver);
}

sub vcl_deliver {
    return (deliver);
}

sub vcl_error {
    set obj.http.Content-Type = "text/html; charset=utf-8";
    set obj.http.Retry-After = "5";
    synthetic {"
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
```

```
 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html>
  <head>
    <title>"} + obj.status + " " + obj.response + {"</title>
  </head>
  <body>
    <h1>Error "} + obj.status + " " + obj.response + {"</h1>
    <p>"} + obj.response + {"</p>
    <h3>Guru Meditation:</h3>
    <p>XID: "} + req.xid + {"</p>
    <hr>
    <p>Varnish cache server</p>
  </body>
</html>
"};
    return (deliver);
}

sub vcl_init {
        return (ok);
}

sub vcl_fini {
        return (ok);
}
```

ワニス4.0

```
/*
 * The built-in (previously called default) VCL code.
 *
 * NB! You do NOT need to copy & paste all of these functions into your
 * own vcl code, if you do not provide a definition of one of these
 * functions, the compiler will automatically fall back to the default
 * code from this file.
 *
 * This code will be prefixed with a backend declaration built from the
 * -b argument.
 */

vcl 4.0;

######################################################################
# Client side

sub vcl_recv {
    if (req.method == "PRI") {
    /* We do not support SPDY or HTTP/2.0 */
    return (synth(405));
    }
    if (req.method != "GET" &&
      req.method != "HEAD" &&
      req.method != "PUT" &&
      req.method != "POST" &&
      req.method != "TRACE" &&
      req.method != "OPTIONS" &&
      req.method != "DELETE") {
        /* Non-RFC2616 or CONNECT which is weird. */
        return (pipe);
```

```
    }

    if (req.method != "GET" && req.method != "HEAD") {
        /* We only deal with GET and HEAD by default */
        return (pass);
    }
    if (req.http.Authorization || req.http.Cookie) {
        /* Not cacheable by default */
        return (pass);
    }
    return (hash);
}

sub vcl_pipe {
    # By default Connection: close is set on all piped requests, to stop
    # connection reuse from sending future requests directly to the
    # (potentially) wrong backend. If you do want this to happen, you can undo
    # it here.
    # unset bereq.http.connection;
    return (pipe);
}

sub vcl_pass {
    return (fetch);
}

sub vcl_hash {
    hash_data(req.url);
    if (req.http.host) {
        hash_data(req.http.host);
    } else {
        hash_data(server.ip);
    }
    return (lookup);
}

sub vcl_purge {
    return (synth(200, "Purged"));
}

sub vcl_hit {
    if (obj.ttl >= 0s) {
        // A pure unadultered hit, deliver it
        return (deliver);
    }
    if (obj.ttl + obj.grace > 0s) {
        // Object is in grace, deliver it
        // Automatically triggers a background fetch
        return (deliver);
    }
    // fetch & deliver once we get the result
    return (fetch);
}

sub vcl_miss {
    return (fetch);
}

sub vcl_deliver {
    return (deliver);
}
```

```
/*
 * We can come here "invisibly" with the following errors: 413, 417 & 503
 */
sub vcl_synth {
    set resp.http.Content-Type = "text/html; charset=utf-8";
    set resp.http.Retry-After = "5";
    synthetic( {"<!DOCTYPE html>
<html>
  <head>
    <title>"} + resp.status + " " + resp.reason + {"</title>
  </head>
  <body>
    <h1>Error "} + resp.status + " " + resp.reason + {"</h1>
    <p>"} + resp.reason + {"</p>
    <h3>Guru Meditation:</h3>
    <p>XID: "} + req.xid + {"</p>
    <hr>
    <p>Varnish cache server</p>
  </body>
</html>
"} );
    return (deliver);
}

#########################################################################
# Backend Fetch

sub vcl_backend_fetch {
    return (fetch);
}

sub vcl_backend_response {
    if (beresp.ttl <= 0s ||
      beresp.http.Set-Cookie ||
      beresp.http.Surrogate-control ~ "no-store" ||
      (!beresp.http.Surrogate-Control &&
        beresp.http.Cache-Control ~ "no-cache|no-store|private") ||
      beresp.http.Vary == "*") {
        /*
        * Mark as "Hit-For-Pass" for the next 2 minutes
        */
        set beresp.ttl = 120s;
        set beresp.uncacheable = true;
    }
    return (deliver);
}

sub vcl_backend_error {
    set beresp.http.Content-Type = "text/html; charset=utf-8";
    set beresp.http.Retry-After = "5";
    synthetic( {"<!DOCTYPE html>
<html>
  <head>
    <title>"} + beresp.status + " " + beresp.reason + {"</title>
  </head>
  <body>
    <h1>Error "} + beresp.status + " " + beresp.reason + {"</h1>
    <p>"} + beresp.reason + {"</p>
    <h3>Guru Meditation:</h3>
    <p>XID: "} + bereq.xid + {"</p>
```

```
    <hr>
    <p>Varnish cache server</p>
  </body>
</html>
"} );
    return (deliver);
}

#############################################################################
# Housekeeping

sub vcl_init {
    return (ok);
}

sub vcl_fini {
    return (ok);
}
```

オンラインでVCLをむ https://riptutorial.com/ja/varnish/topic/5001/vcl

クレジット

S. No		Contributors
1	ワニスをめる	alejdg, Community, Daniel V.
2	vmodの	Daniel V.
3	モニタニス	Jensd
4	VCL	Daniel V., fgsch, Redithion