

Kubernetes Ingress-nginx高级用法 原创

南宫乘风 2021-07-29 11:20:16 博文文章分类: Kubernetes

©著作权

文章标签 ingress kubernetes ingress 文章分类 Nginx 服务器 阅读数 437

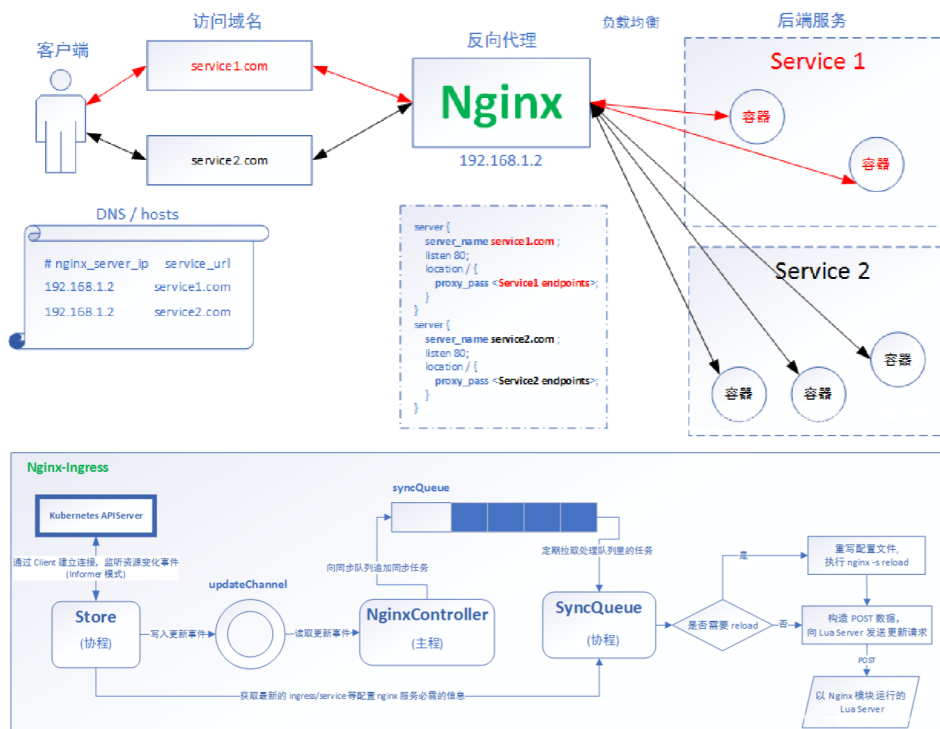
1、什么是ingress

ingress（在kubernetes v1.1时添加）暴露从集群外到集群内服务的HTTP或HTTPS路由。定义在ingress资源上的规则控制流量的路由。

1. internet
2. |
3. [Ingress]
4. --|-----|--
5. [Services]

一个ingress可以配置用于提供外部可访问的服务url、负载均衡流量、SSL终端和提供虚拟主机名配置。ingress controller负责实现（通常使用负载均衡器(loadbalancer)）入口（ingress）。但是它也可以配置你的边缘路由器或额外的前端来帮助处理流量。

ingress不暴露任何端口或协议。将HTTP和HTTPS之外的服务公开到因特网通常使用类型是NodePort或loadbalancer的service。



文章目录

- 6、Ingress-nginx的前后端分
- 7、Ingress-nginx的SSL配置
- 8、Ingress-nginx的黑白名单
 - (1)、白名单 添加白名单的
 - (2)、黑名单 黑名单就只能
- 9、Ingress-nginx的匹配请求:
- 10、Ingress-nginx的速率限
- 11、Ingress-nginx的基本认
- (1)、创建密码, 我这里用
- (2)、创建secret
- (3)、配置Ingress
- 12、Ingress-nginx实现灰度
- 13、Ingress-nginx自定义错

calico-node 报错calico/node

解决二进制K8S部署的metrics

kubeadm部署Kubernetes (k8

Kubernetes (k8s) 的调度器

近期文章

- 1.GO的WEB编程 (GIN实现邮
- 2.TCP扫描增强器实现65000端I
- 3.golang的ping检测主机存活
- 4.Centos7集群时间同步 (Chro
- 5.集群服务器的网络连接状态接



2021年

12月	11月	10月
3篇	5篇	2篇
08月	07月	
7篇	288篇	

热门文章

- Rsyslog同步集群服务器的网
- 集群服务器的网络连接状态接
- TCP扫描增强器实现65000端I
- golang的ping检测主机存活
- GO的WEB编程 (GIN实现邮

2、Ingress区别

差别: <https://github.com/nginxinc/kubernetes-ingress/blob/master/docs/nginx-ingress-controllers.md>

Ingress-nginx的官方文档: <https://kubernetes.github.io/ingress-nginx/user-guide/nginx-configuration/annotations/#rewrite>

Ingress-nginx Github: <https://github.com/kubernetes/ingress-nginx>

Nginx-ingress

Nginx-ingress: nginx官方维护的ingress

Nginx-ingress的官方文档: <https://docs.nginx.com/nginx-ingress-controller/configuration/ingress-resources/advanced-configuration-with-annotations/>

Nginx-ingress Github: <https://github.com/nginxinc/kubernetes-ingress/blob/master/docs/nginx-ingress-controllers.md>

类似的还有: Traefik、HAProxy、Istio

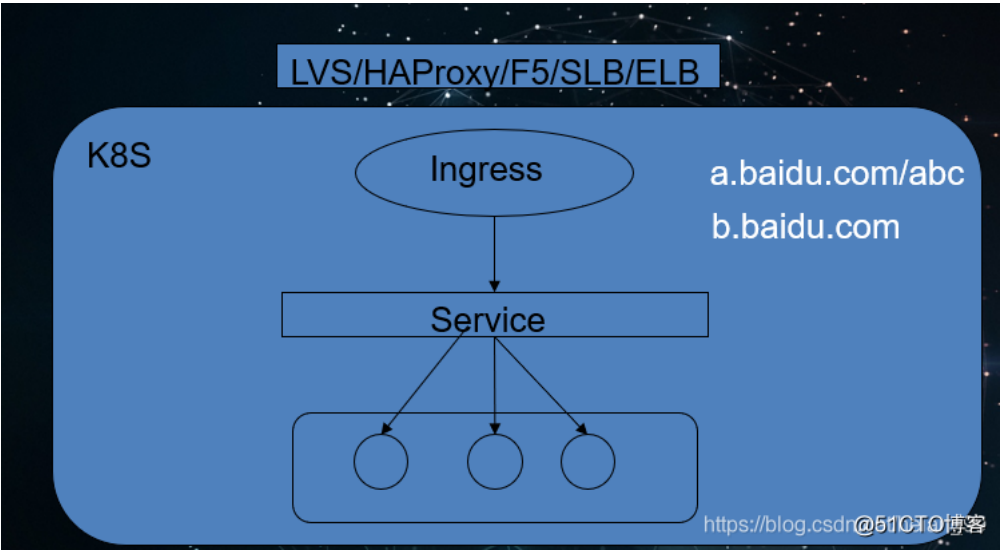
3、Ingress-nginx部署

- (1) yaml部署Ingress-nginx
- (2) helm部署Ingress-nginx

Ingress-nginx的高级介绍, 我这边以Kubernetes的那个插件为主。

4、Ingress-nginx创建流程

- (1) 首先创建pod
- (2) 创建service
- (3) 创建ingress-nginx



- 1. kubectl create ns heian
- 2. #创建命名空间后, 运行下面yaml, 就可以实现上面三个步骤的工作
- 3.
- 4. ---
- 5. apiVersion: apps/v1
- 6. kind: Deployment
- 7. metadata:
- 8. namespace: heian
- 9. name: ingress-heian
- 10. annotations:

【Nginx】Nginx概述

文章目录

- 6、Ingress-nginx的前后端分离
- 7、Ingress-nginx的SSL配置
- 8、Ingress-nginx的黑白名单
 - (1)、白名单 添加白名单的
 - (2)、黑名单 黑名单就只能
- 9、Ingress-nginx的匹配请求:
- 10、Ingress-nginx的速率限制
- 11、Ingress-nginx的基本认证
 - (1)、创建密码, 我这里用h
 - (2)、创建secret
 - (3)、配置Ingress
- 12、Ingress-nginx实现灰度全
- 13、Ingress-nginx自定义错误

相关标签

helm ingress-nginx helm ingress-nginx ingress-nginx ingress-nginx怎么映射服务 ingress-nginx的mandatory.yaml ingress-nginx部署 k8s ingress nginx用法 k8s ingress部署前端代码 k8sv1.18安装nginx



```
14.   k8s.kuboard.cn/ingress: 'true'
15.   labels:
16.     app: ingress-heian
17.   spec:
18.     selector:
19.       matchLabels:
20.         app: ingress-heian
21.     revisionHistoryLimit: 10
22.     template:
23.       metadata:
24.         labels:
25.           app: ingress-heian
26.       spec:
27.         affinity: {}
28.         securityContext:
29.           seLinuxOptions: {}
30.         imagePullSecrets: []
31.         restartPolicy: Always
32.         initContainers: []
33.         containers:
34.           - image: 'wangyanglinux/myapp:v1'
35.             imagePullPolicy: IfNotPresent
36.             name: ingress-heian
37.             volumeMounts:
38.               - name: tz-config
39.                 mountPath: /usr/share/zoneinfo/Asia/Shanghai
40.               - name: tz-config
41.                 mountPath: /etc/localtime
42.               - name: timezone
43.                 mountPath: /etc/timezone
44.             resources:
45.               limits:
46.                 cpu: 100m
47.                 memory: 100Mi
48.               requests:
49.                 cpu: 10m
50.                 memory: 10Mi
51.             env:
52.               - name: TZ
53.                 value: Asia/Shanghai
54.               - name: LANG
55.                 value: C.UTF-8
56.             lifecycle: {}
57.             ports:
58.               - name: web
59.                 containerPort: 80
60.                 protocol: TCP
61.             terminationMessagePath: /dev/termination-log
62.             terminationMessagePolicy: File
63.         volumes:
64.           - name: tz-config
65.             hostPath:
66.               path: /usr/share/zoneinfo/Asia/Shanghai
67.               type: ""
68.           - name: timezone
69.             hostPath:
70.               path: /etc/timezone
71.               type: ""
72.         dnsPolicy: ClusterFirst
73.         dnsConfig:
74.           options: []
75.         schedulerName: default-scheduler
76.         terminationGracePeriodSeconds: 30
77.         progressDeadlineSeconds: 600
78.         strategy:
79.           type: RollingUpdate
80.         rollingUpdate:
```

文章目录

- 6、Ingress-nginx的前后端分
- 7、Ingress-nginx的SSL配置
- 8、Ingress-nginx的黑白名单
 - (1)、白名单 添加白名单的
 - (2)、黑名单 黑名单就只能
- 9、Ingress-nginx的匹配请求:
- 10、Ingress-nginx的速率限
- 11、Ingress-nginx的基本认
 - (1)、创建密码, 我这里用h
 - (2)、创建secret
 - (3)、配置Ingress
- 12、Ingress-nginx实现灰度全
- 13、Ingress-nginx自定义错误

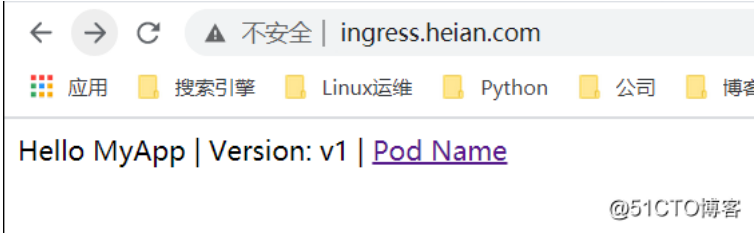


```
84. ---
85. ---
86. apiVersion: v1
87. kind: Service
88. metadata:
89.   namespace: heian
90.   name: ingress-heian
91.   annotations:
92.     k8s.kuboard.cn/workload: ingress-heian
93.   labels:
94.     app: ingress-heian
95. spec:
96.   selector:
97.     app: ingress-heian
98.   type: ClusterIP
99.   ports:
100.    - port: 80
101.      targetPort: 80
102.      protocol: TCP
103.      name: ingress-web-1
104.      nodePort: 0
105.   sessionAffinity: None
106. ---
107. ---
108. apiVersion: networking.k8s.io/v1beta1
109. kind: Ingress
110. metadata:
111.   namespace: heian
112.   name: ingress-heian
113.   annotations:
114.     k8s.kuboard.cn/workload: ingress-heian
115.   labels:
116.     app: ingress-heian
117. spec:
118.   rules:
119.    - host: ingress.heian.com
120.      http:
121.        paths:
122.         - path: /
123.           backend:
124.             serviceName: ingress-heian
125.             servicePort: 80
```

文章目录

- 6、Ingress-nginx的前后端分
- 7、Ingress-nginx的SSL配置
- 8、Ingress-nginx的黑白名单
 - (1)、白名单 添加白名单的
 - (2)、黑名单 黑名单就只能
- 9、Ingress-nginx的匹配请求:
- 10、Ingress-nginx的速率限
- 11、Ingress-nginx的基本认
 - (1)、创建密码, 我这里用h
 - (2)、创建secret
 - (3)、配置Ingress
- 12、Ingress-nginx实现灰度全
- 13、Ingress-nginx自定义错误

效果截图:



5、Ingress-nginx的域名重定向 (Redirect)

annotations声明

- 1. #重定向, 就是这一句, 现在访问这个域名, 会重定向到我的博客地址
- 2. annotations:
- 3. nginx.ingress.kubernetes.io/permanent-redirect: 'https://blog.csdn.net/heian_99'



```
3. kind: Ingress
4. metadata:
5. annotations:
6.   nginx.ingress.kubernetes.io/permanent-redirect: 'https://blog.csdn.net/heian_99'
7. name: ingress-heian
8. namespace: heian
9. spec:
10. rules:
11.   - host: ingress.heian.com
12.     http:
13.       paths:
14.         - backend:
15.             serviceName: ingress-heian
16.             servicePort: 80
17.         path: /
18.         pathType: ImplementationSpecific
```

文章目录

- 6、Ingress-nginx的前后端分
- 7、Ingress-nginx的SSL配置
- 8、Ingress-nginx的黑白名单
 - (1)、白名单 添加白名单的;
 - (2)、黑名单 黑名单就只能
- 9、Ingress-nginx的匹配请求:
- 10、Ingress-nginx的速率限
- 11、Ingress-nginx的基本认证
 - (1)、创建密码, 我这里用
 - (2)、创建secret
 - (3)、配置Ingress
- 12、Ingress-nginx实现灰度全
- 13、Ingress-nginx自定义错误

这个是ingress-nginx里面nginx的配置

```
## end server grafana.test.com

## start server ingress.heian.com
server {
    server_name ingress.heian.com ;

    listen 80 ;
    listen [::]:80 ;
    listen 443 ssl http2 ;
    listen [::]:443 ssl http2 ;

    set $proxy upstream_name "-";

    ssl_certificate_by_lua_block {
        certificate.call()
    }

    location / {

        set $namespace      "heian";
        set $ingress_name    "ingress-heian";
        set $service_name    "ingress-heian";
        set $service_port    "80";
        set $location_path   "/";

        rewrite by_lua_block {
            lua_ingress.rewrite({
                force_ssl_redirect = false,
                ssl_redirect = true,
                force_no_ssl_redirect = false,
                use_port_in_redirects = false,
            })
            balancer.rewrite()
            plugins.run()
        }

        # be careful with 'access by_lua_block' and 'satisfy any' directives as satisfy any
        # will always succeed when there's 'access by_lua_block' that does not have any lua code doing 'ngx.exit(ngx.DECLINED)'
        # other authentication method such as basic auth or external auth useless - all requests will be allowed.
        #access by_lua_block {
        #}

        header_filter_by_lua_block {
            lua_ingress.header()
            plugins.run()
        }

        body_filter_by_lua_block {
        }

        log_by_lua_block {
            balancer.log()

            monitor.call()

            plugins.run()
        }

        port_in_redirect off;

        set $balancer_ewma_score -1;
        set $proxy_upstream_name "heian-ingress-heian-80";
        set $proxy_host          $proxy_upstream_name;
        set $pass_access_scheme  $scheme;

        set $pass_server_port    $server_port;

        set $best_http_host      $http_host;
        set $pass_port           $pass_server_port;

        set $proxy_alternative_upstream_name "";

        client_max_body_size    1m;

        proxy_set_header Host      $best_http_host;

        # Pass the extracted client certificate to the backend

        # Allow websocket connections
        proxy_set_header          Upgrade          $http_upgrade;

        proxy_set_header          Connection       $connection_upgrade;

        proxy_set_header X-Request-ID $req_id;
        proxy_set_header X-Real-IP   $remote_addr;

        proxy_set_header X-Forwarded-For $remote_addr;

        proxy_set_header X-Forwarded-Host $best_http_host;
        proxy_set_header X-Forwarded-Port $pass_port;
        proxy_set_header X-Forwarded-Proto $pass_access_scheme;

        proxy_set_header X-Scheme      $pass_access_scheme;
    }
```



```
# Custom headers to proxied server

proxy_connect_timeout      5s;
proxy_send_timeout         60s;
proxy_read_timeout         60s;

proxy_buffering            off;
proxy_buffer_size          4k;
proxy_buffers               4 4k;

proxy_max_temp_file_size   1024m;

proxy_request_buffering    on;
proxy_http_version         1.1;

proxy_cookie_domain        off;
proxy_cookie_path          off;

# In case of errors try the next upstream server before returning an error
proxy_next_upstream        error timeout;
proxy_next_upstream_timeout 0;
proxy_next_upstream_tries  3;

return 301 https://blog.csdn.net/heian_99;
proxy_pass http://upstream_balancer;
proxy_redirect              off;

}

## end server ingress.heian.com

## start server krm.test.com
server {
    server_name krm.test.com ;

    listen 80 ;
}
```

https://blog.csdn.net/heian_99

文章目录

- 6、Ingress-nginx的前后端分离
- 7、Ingress-nginx的SSL配置
- 8、Ingress-nginx的黑白名单
 - (1)、白名单 添加白名单的;
 - (2)、黑名单 黑名单就只能;
- 9、Ingress-nginx的匹配请求:
- 10、Ingress-nginx的速率限制
- 11、Ingress-nginx的基本认证
 - (1)、创建密码, 我这里用h
 - (2)、创建secret
 - (3)、配置Ingress
- 12、Ingress-nginx实现灰度全
- 13、Ingress-nginx自定义错误

6、Ingress-nginx的前后端分离 (Rewrite)

1. annotations:
2. nginx.ingress.kubernetes.io/rewrite-target: /\$2

1. ---
2. apiVersion: networking.k8s.io/v1beta1
3. kind: Ingress
4. metadata:
5. annotations:
6. nginx.ingress.kubernetes.io/rewrite-target: /\$2
7. name: ingress-heian
8. namespace: heian
9. spec:
10. rules:
11. - host: ingress.heian.com
12. http:
13. paths:
14. - backend:
15. serviceName: ingress-heian
16. servicePort: 80
17. path: /something/(/\$(.*)
18. pathType: ImplementationSpecific

← → ↻ 🔒 不安全 | ingress.heian.com

应用 搜索引擎 Linux运维 Python 公司 博客 论坛 学习 云服务 视频 百度翻译 技术

404 Not Found

nginx
https://blog.csdn.net/heian_99



Hello MyApp | Version: v1 | [Pod Name](#)

@51CTO博客

文章目录

- 6、Ingress-nginx的前后端分
- 7、Ingress-nginx的SSL配置
- 8、Ingress-nginx的黑白名单
 - (1)、白名单 添加白名单的
 - (2)、黑名单 黑名单就只能
- 9、Ingress-nginx的匹配请求:
- 10、Ingress-nginx的速率限
- 11、Ingress-nginx的基本认
 - (1)、创建密码, 我这里用
 - (2)、创建secret
 - (3)、配置Ingress
- 12、Ingress-nginx实现灰度
- 13、Ingress-nginx自定义错

7、Ingress-nginx的SSL配置

官网: <https://kubernetes.github.io/ingress-nginx/user-guide/tls/>

创建自建证书 (主: 浏览器不认可的)

Ingress-nginx配置了SSL, 会自动跳转到https的网页的

```
1. openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout tls.key -out tls.cert -subj "/CN=ingress.heian.com/O=ingress-heian"
2.
3. kubectl create secret tls ca-ceart --key tls.key --cert tls.cert -n heian
```

```
1. ---
2. apiVersion: networking.k8s.io/v1beta1
3. kind: Ingress
4. metadata:
5.   annotations:
6.     nginx.ingress.kubernetes.io/rewrite-target: /$2
7.   name: ingress-heian
8.   namespace: heian
9. spec:
10.  rules:
11.  - host: ingress.heian.com
12.    http:
13.      paths:
14.      - backend:
15.          serviceName: ingress-heian
16.          servicePort: 80
17.        path: /something/(.*)
18.        pathType: ImplementationSpecific
19.  tls:
20.  - hosts:
21.    - ingress.heian.com
22.    secretName: ca-ceart
```

```
1.  tls:
2.  -  hosts:
3.    -  ingress.heian.com
4.    secretName: ca-ceart
```



Hello MyApp | Version: v1 | [Pod Name](#)

@51CTO博客

禁用https强制跳转

```
1. annotations:
2.   nginx.ingress.kubernetes.io/ssl-redirect: "false"
```



8、Ingress-nginx的黑白名单

Annotations：只对指定的ingress生效

ConfigMap：全局生效

黑名单可以使用ConfigMap去配置，白名单建议使用Annotations去配置。

(1)、白名单 添加白名单的方式可以直接写annotation，也可以配置在ConfigMap中。

写在annotation中：

```
1.  ---
2.  apiVersion: networking.k8s.io/v1beta1
3.  kind: Ingress
4.  metadata:
5.    annotations:
6.      nginx.ingress.kubernetes.io/whitelist-source-range: 192.168.0.100
7.  name: ingress-heian
8.  namespace: heian
9.  spec:
10.   rules:
11.     - host: ingress.heian.com
12.       http:
13.         paths:
14.           - backend:
15.               serviceName: ingress-heian
16.               servicePort: 80
17.             path: /
18.             pathType: ImplementationSpecific
```

```
[root@k8s-master01 ~]# hostname -i
192.168.0.100
[root@k8s-master01 ~]# curl ingress.heian.com -I
HTTP/1.1 200 OK
Date: Sat, 20 Mar 2021 03:18:43 GMT
Content-Type: text/html
Content-Length: 65
Connection: keep-alive
Last-Modified: Fri, 02 Mar 2018 03:39:12 GMT
ETag: "5a98c760-41"
Accept-Ranges: bytes

[root@k8s-master01 ~]#
```

<https://blog.csdn.net/51CTO博客>

```
[root@k8s-node02 ~]# hostname -i
192.168.0.109
[root@k8s-node02 ~]# curl ingress.heian.com -I
HTTP/1.1 403 Forbidden
Date: Sat, 20 Mar 2021 03:19:15 GMT
Content-Type: text/html
Content-Length: 146
Connection: keep-alive

[root@k8s-node02 ~]#
```

@51CTO博客

也可以写固定IP，也可以写网段。配置到ConfigMap中：

```
1.  apiVersion: v1
2.  kind: ConfigMap
3.  metadata:
4.    labels:
5.      helm.sh/chart: ingress-nginx-2.1.0
6.      app.kubernetes.io/name: ingress-nginx
7.      app.kubernetes.io/instance: ingress-nginx
8.      app.kubernetes.io/version: 0.32.0
9.      app.kubernetes.io/managed-by: Helm
10.     app.kubernetes.io/component: controller
```

文章目录

- 6、Ingress-nginx的前后端分
- 7、Ingress-nginx的SSL配置
- 8、Ingress-nginx的黑白名单
 - (1)、白名单 添加白名单的
 - (2)、黑名单 黑名单就只能
- 9、Ingress-nginx的匹配请求:
- 10、Ingress-nginx的速率限
- 11、Ingress-nginx的基本认
 - (1)、创建密码，我这里用
 - (2)、创建secret
 - (3)、配置Ingress
- 12、Ingress-nginx实现灰度
- 13、Ingress-nginx自定义错




```
14.   whitelist-source-range: 10.1.10.0/24
```

(2)、黑名单 黑名单就只能通过ConfigMap来配置

ConfigMap配置如下:

```
1.   apiVersion: v1
2.   kind: ConfigMap
3.   metadata:
4.     labels:
5.       helm.sh/chart: ingress-nginx-2.1.0
6.       app.kubernetes.io/name: ingress-nginx
7.       app.kubernetes.io/instance: ingress-nginx
8.       app.kubernetes.io/version: 0.32.0
9.       app.kubernetes.io/managed-by: Helm
10.      app.kubernetes.io/component: controller
11.   name: ingress-nginx-controller
12.   namespace: ingress-nginx
13.   data:
14.     whitelist-source-range: 10.1.10.0/24
15.     block-cidrs: 10.1.10.100
```

annotation配置

```
1.   ---
2.   apiVersion: networking.k8s.io/v1beta1
3.   kind: Ingress
4.   metadata:
5.     annotations:
6.       kubernetes.io/ingress.class: nginx
7.       nginx.ingress.kubernetes.io/server-snippet: |-
8.         deny 192.168.0.1;
9.         deny 192.168.0.100;
10.        allow all;
11.   creationTimestamp: null
12.   name: ingress-heian
13.   spec:
14.     rules:
15.       - host: ingress.heian.com
16.         http:
17.           paths:
18.             - backend:
19.                 serviceName: ingress-heian
20.                 servicePort: 80
21.               path: /
22.               pathType: ImplementationSpecific
23.   status:
24.     loadBalancer: {}
```

9、Ingress-nginx的匹配请求头

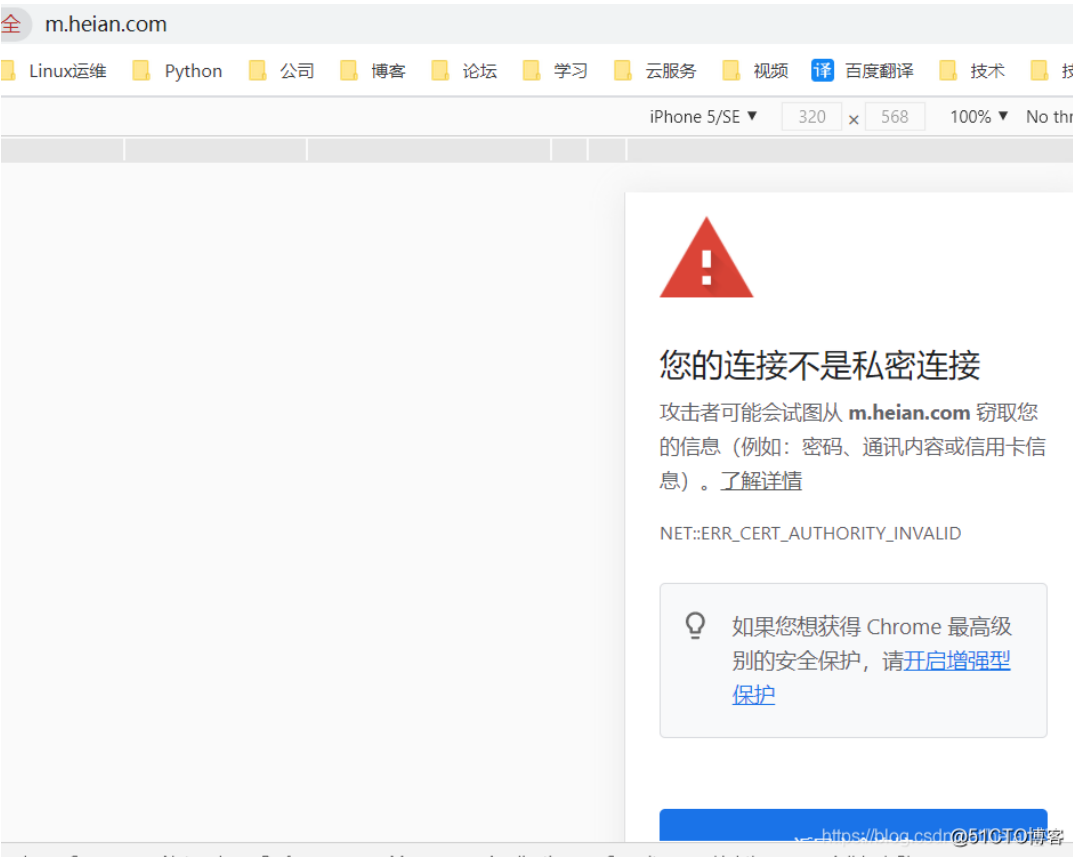
```
1.   ---
2.   apiVersion: networking.k8s.io/v1beta1
3.   kind: Ingress
4.   metadata:
5.     annotations:
6.       kubernetes.io/ingress.class: nginx
7.       nginx.ingress.kubernetes.io/server-snippet: |-
8.         set $agentflag 0;
9.
10.        if ($http_user_agent ~* "(iPhone)") {
11.          set $agentflag 1;
12.        }
```

文章目录

- 6、Ingress-nginx的前后端分
- 7、Ingress-nginx的SSL配置
- 8、Ingress-nginx的黑白名单
 - (1)、白名单 添加白名单的
 - (2)、黑名单 黑名单就只能
- 9、Ingress-nginx的匹配请求:
- 10、Ingress-nginx的速率限
- 11、Ingress-nginx的基本认
- (1)、创建密码, 我这里用h
 - (2)、创建secret
 - (3)、配置Ingress
- 12、Ingress-nginx实现灰度全
- 13、Ingress-nginx自定义错误



```
16.     }
17.     creationTimestamp: null
18.     name: ingress-heian
19.   spec:
20.     rules:
21.     - host: ingress.heian.com
22.       http:
23.         paths:
24.         - backend:
25.             serviceName: ingress-heian
26.             servicePort: 80
27.           path: /
28.           pathType: ImplementationSpecific
29.   status:
30.     loadBalancer: {}
```



文章目录

- 6、Ingress-nginx的前后端分
- 7、Ingress-nginx的SSL配置
- 8、Ingress-nginx的黑白名单
 - (1)、白名单 添加白名单的
 - (2)、黑名单 黑名单就只能
- 9、Ingress-nginx的匹配请求:
- 10、Ingress-nginx的速率限
- 11、Ingress-nginx的基本认
 - (1)、创建密码, 我这里用h
 - (2)、创建secret
 - (3)、配置Ingress
- 12、Ingress-nginx实现灰度全
- 13、Ingress-nginx自定义错

10、Ingress-nginx的速率限制

- 1. 限速¶
- 2. 这些注释定义了对连接和传输速率的限制。这些可以用来减轻DDoS攻击。
- 3.
- 4. nginx.ingress.kubernetes.io/limit-connections: 单个IP地址允许的并发连接数。超出此限制时，将返回503错误。
- 5. nginx.ingress.kubernetes.io/limit-rps: 每秒从给定IP接受的请求数。突发限制设置为此限制乘以突发乘数，默认乘数为5。
- 6. nginx.ingress.kubernetes.io/limit-rpm: 每分钟从给定IP接受的请求数。突发限制设置为此限制乘以突发乘数，默认乘数为5。
- 7. nginx.ingress.kubernetes.io/limit-burst-multiplier: 突发大小限制速率的倍数。默认的脉冲串乘数为5，此注释将覆盖默认值。
- 8. nginx.ingress.kubernetes.io/limit-rate-after: 最初的一千字节数，在此之后，对给定连接的响应的进一步传输将受到速率的限制。
- 9. nginx.ingress.kubernetes.io/limit-rate: 每秒允许发送到给定连接的一千字节数。零值禁用速率限制。必须在启用代理缓冲的情况下使用。
- 10. nginx.ingress.kubernetes.io/limit-whitelist: 客户端IP源范围要从速率限制中排除。该值是逗号分隔的CIDR列表。



```
3. metadata:
4.   name: ingress-nginx
5.   annotations:
6.     kubernetes.io/ingress.class: "nginx"
7.     nginx.ingress.kubernetes.io/limit-rate: 100K
8.     nginx.ingress.kubernetes.io/limit-whitelist: 10.1.10.100
9.     nginx.ingress.kubernetes.io/limit-rps: 1
10.    nginx.ingress.kubernetes.io/limit-rpm: 30
11. spec:
12.   rules:
13.     - host: iphone.coolops.cn
14.       http:
15.         paths:
16.           - path:
17.             backend:
18.               serviceName: ng-svc
19.               servicePort: 80
20.
21.
22.
23. nginx.ingress.kubernetes.io/limit-rate: 限制客户端每秒传输的字节数
24. nginx.ingress.kubernetes.io/limit-whitelist: 白名单中的IP不限速
25. nginx.ingress.kubernetes.io/limit-rps: 单个IP每秒的连接数
26. nginx.ingress.kubernetes.io/limit-rpm: 单个IP每分钟的连接数
```

文章目录

- 6、Ingress-nginx的前后端分
- 7、Ingress-nginx的SSL配置
- 8、Ingress-nginx的黑白名单
 - (1)、白名单 添加白名单的
 - (2)、黑名单 黑名单就只能
- 9、Ingress-nginx的匹配请求:
- 10、Ingress-nginx的速率限
- 11、Ingress-nginx的基本认
 - (1)、创建密码, 我这里用h
 - (2)、创建secret
 - (3)、配置Ingress
- 12、Ingress-nginx实现灰度
- 13、Ingress-nginx自定义错

11、Ingress-nginx的基本认证

有些访问是需要认证访问的, 比如dubbo-admin, 我们在访问的时候会先叫你输入用户名和密码。ingress nginx也可以实现这种。

(1)、创建密码, 我这里用http的命令工具来生成

```
1. [root@k8s-master01 ingress]# httpasswd -c auth heian
2. New password:
3. Re-type new password:
4. Adding password for user heian
5. [root@k8s-master01 ingress]# ls
6. auth  tls.cert  tls.key
7. [root@k8s-master01 ingress]# cat auth
8. heian:$apr1$8LffOJL7$ZIGV4XRNSuginqO5GMxAZ.
9. [root@k8s-master01 ingress]#
```

(2)、创建secret

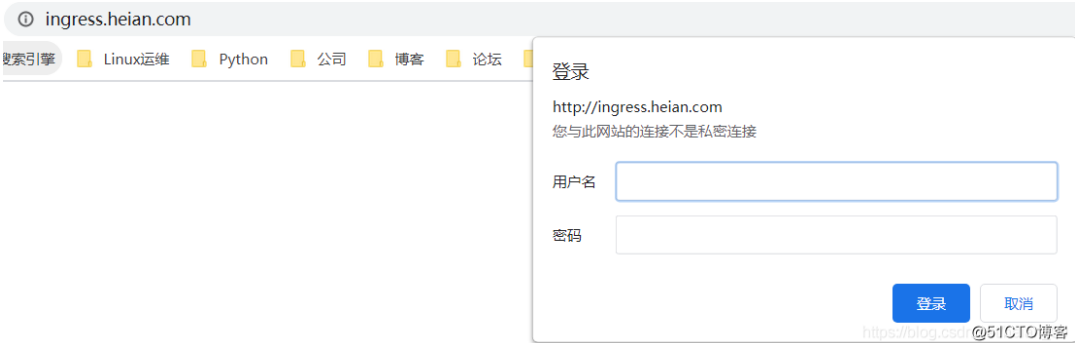
```
1. [root@k8s-master01 ingress]# kubectl create secret generic basic-auth --from-file=auth -n heian
2. secret/basic-auth created
```

(3)、配置Ingress

```
1. ---
2. apiVersion: networking.k8s.io/v1beta1
3. kind: Ingress
4. metadata:
5.   annotations:
6.     kubernetes.io/ingress.class: nginx
7.     nginx.ingress.kubernetes.io/auth-realm: Need to login
8.     nginx.ingress.kubernetes.io/auth-secret: basic-auth
9.     nginx.ingress.kubernetes.io/auth-type: basic
```



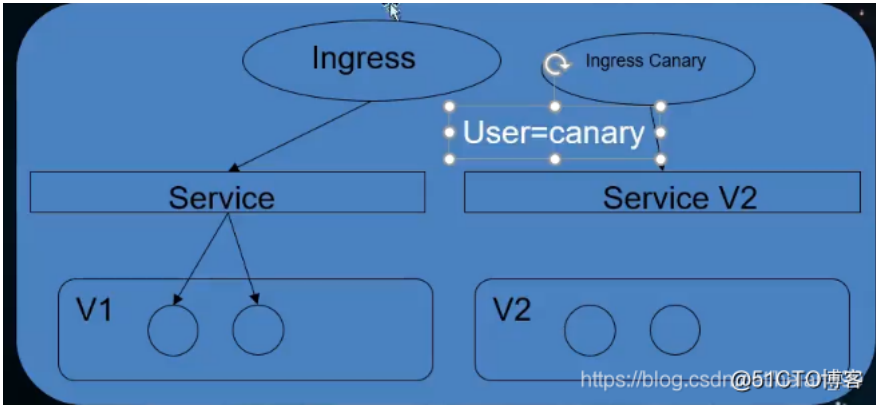
```
13. rules:
14. - host: ingress.heian.com
15. http:
16. paths:
17. - backend:
18.     serviceName: ingress-heian
19.     servicePort: 80
20. path: /
21. pathType: ImplementationSpecific
22. status:
23. loadBalancer: {}
```



文章目录

- 6、Ingress-nginx的前后端分
- 7、Ingress-nginx的SSL配置
- 8、Ingress-nginx的黑白名单
 - (1)、白名单 添加白名单的
 - (2)、黑名单 黑名单就只能
- 9、Ingress-nginx的匹配请求:
- 10、Ingress-nginx的速率限
- 11、Ingress-nginx的基本认证
 - (1)、创建密码, 我这里用h
 - (2)、创建secret
 - (3)、配置Ingress
- 12、Ingress-nginx实现灰度全
- 13、Ingress-nginx自定义错误

12、Ingress-nginx实现灰度金丝雀发布



- Nginx Annotations 支持以下 4 种 Canary 规则:
- nginx.ingress.kubernetes.io/canary-by-header: 基于 Request Header 的流量切分, 适用于灰度发布以及 A/B 测试。当 Request Header 设置为 always时, 请求将会被一直发送到 Canary 版本; 当 Request Header 设置为 never时, 请求不会被发送到 Canary 入口; 对于任何其他 Header 值, 将忽略 Header, 并通过优先级将请求与其他金丝雀规则进行优先级的比较。
 - nginx.ingress.kubernetes.io/canary-by-header-value: 要匹配的 Request Header 的值, 用于通知 Ingress 将请求路由到 Canary Ingress 中指定的服务。当 Request Header 设置为此值时, 它将被路由到 Canary 入口。该规则允许用户自定义 Request Header 的值, 必须与上一个 annotation (即: canary-by-header) 一起使用。
 - nginx.ingress.kubernetes.io/canary-weight: 基于服务权重的流量切分, 适用于蓝绿部署, 权重范围 0 - 100 按百分比将请求路由到 Canary Ingress 中指定的服务。权重为 0 意味着该金丝雀规则不会向 Canary 入口的服务发送任何请求。权重为 100 意味着所有请求都将被发送到 Canary 入口。
 - nginx.ingress.kubernetes.io/canary-by-cookie: 基于 Cookie 的流量切分, 适用于灰度发布与 A/B 测试。用于通知 Ingress 将请求路由到 Canary Ingress 中指定的服务的cookie。当 cookie 值设置为 always



创建两个同域名的ingress

v2版

```
1.  apiVersion: networking.k8s.io/v1beta1
2.  kind: Ingress
3.  metadata:
4.  annotations:
5.    nginx.ingress.kubernetes.io/canary: "true"
6.    nginx.ingress.kubernetes.io/canary-by-header: heian
7.    nginx.ingress.kubernetes.io/canary-by-header-value: v2
8.    nginx.ingress.kubernetes.io/canary-weight: "50"
9.  name: ingress-heian2
10. namespace: heian
11. spec:
12.  rules:
13.  - host: ingress.heian.com
14.    http:
15.      paths:
16.      - backend:
17.          serviceName: ingress-heian2
18.          servicePort: 80
19.        path: /
20.        pathType: ImplementationSpecific
```

```
[root@k8s-node02 ~]# for i in `seq 1 10`;do curl -H "heian:v2" ingress.heian.com ;done
Hello MyApp | Version: v2 | <a href="hostname.html">Pod Name</a>
Hello MyApp | Version: v2 | <a href="hostname.html">Pod Name</a>
Hello MyApp | Version: v2 | <a href="hostname.html">Pod Name</a>
Hello MyApp | Version: v2 | <a href="hostname.html">Pod Name</a>
Hello MyApp | Version: v2 | <a href="hostname.html">Pod Name</a>
Hello MyApp | Version: v2 | <a href="hostname.html">Pod Name</a>
Hello MyApp | Version: v2 | <a href="hostname.html">Pod Name</a>
Hello MyApp | Version: v2 | <a href="hostname.html">Pod Name</a>
Hello MyApp | Version: v2 | <a href="hostname.html">Pod Name</a>
Hello MyApp | Version: v2 | <a href="hostname.html">Pod Name</a>
[root@k8s-node02 ~]#
```

@51CTO博客

文章目录

- 6、Ingress-nginx的前后端分
- 7、Ingress-nginx的SSL配置
- 8、Ingress-nginx的黑白名单
 - (1)、白名单 添加白名单的
 - (2)、黑名单 黑名单就只能
- 9、Ingress-nginx的匹配请求:
- 10、Ingress-nginx的速率限
- 11、Ingress-nginx的基本认
- (1)、创建密码, 我这里用
- (2)、创建secret
- (3)、配置Ingress
- 12、Ingress-nginx实现灰度
- 13、Ingress-nginx自定义错

13、Ingress-nginx自定义错误页面

github地址: <https://github.com/kubernetes/ingress-nginx/blob/master/docs/examples/customization/custom-errors/custom-default-backend.yaml>

1. kubectl apply -f error.yaml -n heian

修改ds配置文件, 添加这个

1. --default-backend-service=heian/nginx-errors

```
7  --- app.kubernetes.io/instance: ingress-nginx
8  --- app.kubernetes.io/name: ingress-nginx
9  --- spec:
10 ---   affinity: {}
11 ---   containers:
12 ---   - args:
13       - /nginx-ingress-controller
14       - --publish-service=$(POD_NAMESPACE)/ingress-nginx-controller
15       - --election-id=ingress-controller-leader
16       - --ingress-class=nginx
17       - --default-backend-service=heian/nginx-errors
18       - --configmap=ingress-nginx/ingress-nginx-controller
19       - --validating-webhook=:8443
20       - --validating-webhook-certificate=/usr/local/certificates/cert
21       - --validating-webhook-key=/usr/local/certificates/key
22 ---   env:
23 ---   - name: POD_NAME
```

<https://blog.csdn.net/51cto博客>



```
[root@k8s-master01 ~]# curl http://ingress.heian.com/das
<span>The page you're looking for could not be found.</span>[root@k8s-master01 ~]# curl http://ingress.heian.com/4040
<span>The page you're looking for could not be found.</span>[root@k8s-master01 ~]#
```

@51CTO博客

验证

执行 `kubectl get pod -n ingress-nginx` 查看pod是否已经重启过， 如果没有自动重启， 需要杀掉pod。
打开一个不存在的链接， 查看是否显示的是定义的错误页面。
部分浏览器不支持页面显示， 如谷歌浏览器会显示“无法显示此网站”

赞

收藏


评论

分享

举报

上一篇：Python的文件操作

下一篇：Kubernetes通过插件，自动发现注册Rabbitmq...



提问和评论都可以，用心的回复会被更多人看到

评论

相关文章

Kubernetes + ingress-Nginx 搭建

> 使用kubeadm部署Kubernetes集群 --- | 主机名 | IP地址 | 操作系统|描述| |-----|-----|-----|-----| | master | 192.16...

Kubernetes进阶之ingress-nginx

Kubernetes进阶之ingress-nginx **目录： ** 一 从外部访问应用最佳方式 二 配置管理 三 数据卷与数据持久卷 四 再谈有状态应用...

部署kubernetes/ingress-nginx(踩坑)

nginx-ingress-controller:0.25.0有问题， 所以这里采用nginx-ingress-controller:0.30.0 [root@k8s-master ~]# wget https://raw.git...

Kubernetes之Ingress-nginx部署使用

>博文大纲： 一、Ingress简介 1) Ingress组成 2) Ingress工作原理 3) Ingress可以解决什么问题？ 二、配置Ingress-nginx 1) ...

Kubernetes Ingress-Nginx实现高可用

假定我们在Kubernetes 指定两个worker节点中部署了ingress nginx来为后端的pod做proxy， 这时候我们就需要通过keepalived...

ingress-nginx

root@ubuntu:~/nginx_ingress# kubectl apply -f https://raw.githubusercontent.com/kubernetes/ingress-nginx/controller-v0.4...

Kubernetes Ingress-Nginx 实现蓝绿、灰度发布

文章目录

- 6、Ingress-nginx的前后端分
- 7、Ingress-nginx的SSL配置
- 8、Ingress-nginx的黑白名单
 - (1)、白名单 添加白名单的
 - (2)、黑名单 黑名单就只能
- 9、Ingress-nginx的匹配请求:
- 10、Ingress-nginx的速率限
- 11、Ingress-nginx的基本认
 - (1)、创建密码，我这里用h
 - (2)、创建secret
 - (3)、配置Ingress
- 12、Ingress-nginx实现灰度全
- 13、Ingress-nginx自定义错误



部署ingress-nginx

源文件：<https://raw.githubusercontent.com/kubernetes/ingress-nginx/nginx-0.26.2/deploy/static/mandatory.yaml> 可以通过wg...

使用ingress-nginx

使用Ingress-nginx K爷 DevOps视角 * Ingress简介 * 部署Ingress * 部署ingress-nginx * 部署Service * 部署应用 * 部署Service...

Kubernetes系列之Kubernetes使用ingress-nginx作为反向代理

Kubernetes系列之Kubernetes使用ingress-nginx作为反向代理 #一、Ingress简介 >在Kubernetes中，服务和Pod的IP地址仅可以...

ingress-nginx外部认证

<https://www.lijiaocn.com/soft/k8s/ingress-nginx/auth-ext.html>

ingress-nginx 添加https证书

1.生成证书文件 openssl req -x509 -nodes -days 2920 -newkey rsa:2048 -keyout tls.key -out tls.crt -subj "/CN=*.jdd966.cn/O=...

ingress-nginx设置https证书

创建自签署证书 注意证书中的CN=tls.echo.example改成自己的域名地址。 echo "生成自签署的 ca 证书" openssl req -x509 -sh...

双栈 部署ingress-nginx

下载ingress-nginx yaml `` wget <https://raw.githubusercontent.com/kubernetes/ingress-nginx/master/deploy/static/provider...>

ingress-nginx的各种nginx规则定义

官方文档：<https://github.com/kubernetes/ingress-nginx/blob/master/docs/user-guide/nginx-configuration/annotations.md> 工...

ingress-nginx自带认证功能【nginx自带】

问题：通过nginx可以给某些web网站设置登录使用的用户名和密码，现在网站部署到k8s中，通过配置nginx-ingre...



容器云平台No.8~kubernetes负载均衡之ingress-nginx

容器云平台No.8~kubernetes负载均衡之ingress-nginx scofield 菜鸟运维杂谈 ### Ingress 是什么? ----- Ingress 公开了从集...

kubernetes上部署jenkins并使用ingress-nginx提供域名访问

在 K8S 中部署 jenkins github上有一个1.5k star的项目：<https://github.com/jenkinsci/kubernetes-plugin> 上面提供...



Ingress-nginx灰度发布功能详解

灰度发布使用背景最近公司一直在推进DevOps，主要目标是减少对个人的依赖，降低团队之间的损耗，在保证质...



文章目录

- 6、Ingress-nginx的前后端分
- 7、Ingress-nginx的SSL配置
- 8、Ingress-nginx的黑白名单
 - (1)、白名单 添加白名单的
 - (2)、黑名单 黑名单就只能
- 9、Ingress-nginx的匹配请求:
- 10、Ingress-nginx的速率限
- 11、Ingress-nginx的基本认
- (1)、创建密码，我这里用
 - (2)、创建secret
 - (3)、配置Ingress
- 12、Ingress-nginx实现灰度
- 13、Ingress-nginx自定义错



返回顶部