链滴

Let'"'"'s Encrypt 证书生成，certbot-auto 生成 ssl 通用证书 配置 https 自动续期

作者：450370050

原文链接：https://ld246.com/article/1531709298417

来源网站：链滴

Let’ s Encrypt是一个 CA 机构，但这个 CA 机构是免费的！！！ 签发证书不需要任何费用, 为了实现配符证书，Let’ s Encrypt 对 ACME 协议的实现进行了升级，只有 v2 协议才能支持通配符证书。

https://certbot.eff.org/ 证书生成工具， 我们可以通过网站选择对应的系统软件来生成。

我们通过certbot-auto自动生成工具来操作。

下载

```
wget https://dl.eff.org/certbot-auto
chmod +x certbot-auto
```

生成证书

```
./certbot-auto certonly --email 450370050@qq.com -d *.xxxx.com --manual --preferred-challenges dns --server https://acme-v02.api.letsencrypt.org/directory
```

certonly 表示安装模式，Certbot 有安装模式和验证模式两种类型的插件。

--manual 表示手动安装插件，Certbot 有很多插件，不同的插件都可以申请证书，用户可以根据需自行选择

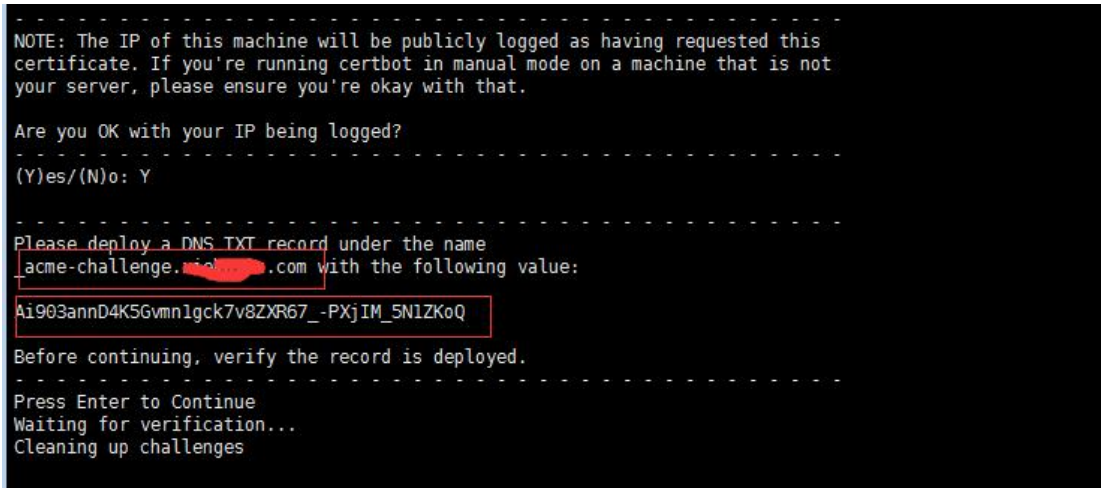
-d 为那些主机申请证书，如果是通配符，输入 *.xxxx.com （可以替换为你自己的域名）

--preferred-challenges dns，使用 DNS 方式校验域名所有权

--server，Let's Encrypt ACME v2 版本使用的服务器不同于 v1 版本，需要显示指定。

操作交互流程

- 是否同意 Let's Encrypt 协议要求
- 询问是否对域名和机器（IP）进行绑定
- 域名验证 dns解析增加TXT配置

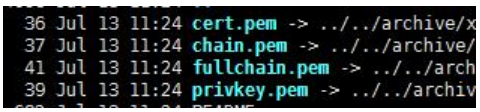


注：最后一步需要dns配置成功生效后再继续

出现TXT验证时先配置dns的txt解析并验证txt解析生效

dig -t txt _acme-challenge.xxxx.com @8.8.8.8

创建成功 /etc/letsencrypt/live/xxxx.com/下会生成4个文件，请勿更改ssl文件位置，这样可以减少动续期时的操作



cert.pem - Apache服务器端证书
chain.pem - Apache根证书和中继证书
fullchain.pem - Nginx所需要ssl_certificate文件
privkey.pem - 安全证书KEY文件
Nginx环境，就只需要用到fullchain.pem和privkey.pem两个证书文件

测试证书

openssl x509 -in /etc/letsencrypt/live/xxxx.com/fullchain.pem -noout -text

nginx 配置

增加443端口监听

```
listen 443 ssl;
```

增加ssl配置

```
ssl_protocols TLSv1 TLSv1.1 TLSv1.2;
ssl_ciphers EECDH+CHACHA20:EECDH+CHACHA20-draft:EECDH+AES128:RSA+AES128:EECD
+AES256:RSA+AES256:EECDH+3DES:RSA+3DES:!MD5;
ssl_session_cache builtin:1000 shared:SSL:10m;
resolver 8.8.8.8 8.8.4.4 valid=300s;
resolver_timeout 5s;
ssl_prefer_server_ciphers on;
ssl_certificate /etc/letsencrypt/live/xxxx.com/fullchain.pem;
ssl_certificate_key /etc/letsencrypt/live/xxxx.com/privkey.pem;
ssl_session_timeout 5m;
ssl_session_tickets on;
ssl_stapling on;
ssl_stapling_verify on;
```

http强制跳转https

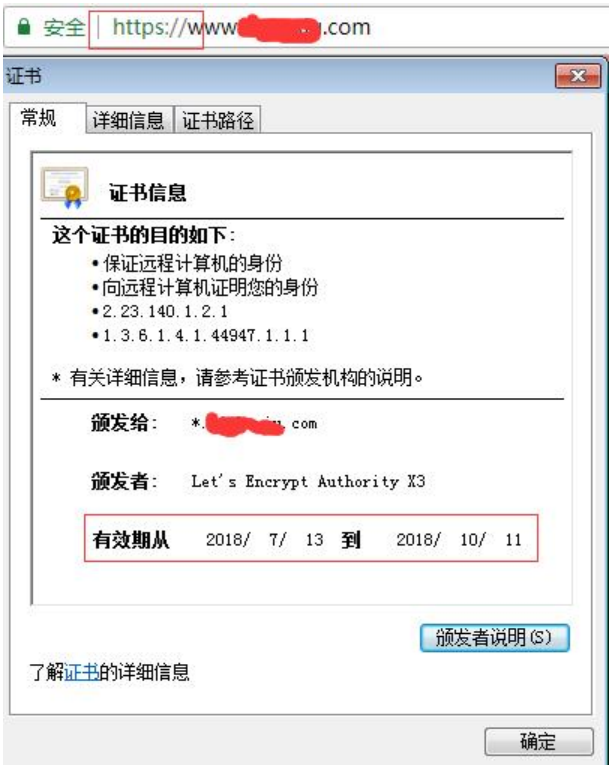
```
server
   {
      listen 80 ;
      server_name www.xxxx.com;
      rewrite ^(.*)$  https://$host$1 permanent;
   }
```

nginx重新加载配置

service nginx reload

验证

我们可以看到https已经生效，有效期3个月



续期

因为有效期只有3个月我们需要自动续期来延长有效期。

通配证书只能通过dns的方式验证域名归属，我们需要通过脚本自动完成验证 --manual-auth-hook 设定验证脚本，否则无法自动更新

./certbot-auto renew --cert-name xxxxx.com --manual-auth-hook /home/certbot-sh/au.sh --dry-run

重要提示：为避免遇到操作次数的限制，加入 dry-run 参数，可以避免作限制，等执行无误后，再进行真实的renew 操作。

au.sh写的是什么呢?

au.sh

au.sh是运行更改dns配置的php文件的一个shell脚本。

```
#!/usr/bin/bash

# PHP 脚本位置
PHPPROGRAM="/home/certbot-sh"
```

```
# 要配置的域名
DOMAIN="xxxx.com"
# 脚本路径
PATH="/home/certbot-sh"

# 要为那个 DNS RR 添加 TXT 记录
CREATE_DOMAIN="_acme-challenge"

# $CERTBOT_VALIDATION 是 Certbot 的内置变量，代表需要为 DNS TXT 记录设置的值
echo $PATH"/alydns.php"

# 调用 PHP 脚本，自动设置 DNS TXT 记录。
/usr/bin/php   $PATH"/alydns.php"  $DOMAIN $CREATE_DOMAIN  $CERTBOT_VALIDATION
/var/log/certdebug.log

# DNS TXT 记录刷新时间
/usr/bin/sleep 30
```

alydns.php

alydns是通过阿里云的接口动态修改dns设置的接口请求文件

```
<?php

date_default_timezone_set("GMT");

//这两个值需要去阿里云申请
define("accessKeyId", "xxxx");
define("accessSecrec", "xxxx");

//manager domain
$obj = new AliDns(accessKeyId, accessSecrec, $argv[1]);
$data = $obj->DescribeDomainRecords();
$data = $data["DomainRecords"]["Record"];
if (is_array($data)) {
    foreach ($data as $v) {
        if ($v["RR"] == $argv[2]) {
            $obj->DeleteDomainRecord($v["RecordId"]);
        }
    }
}

print_r($obj->AddDomainRecord("TXT", $argv[2],$argv[3]));

class AliDns {
    private $accessKeyId = null;
    private $accessSecrec = null;
    private $DomainName = null;


    public function __construct($accessKeyId, $accessSecrec, $domain) {
        $this->accessKeyId = $accessKeyId;
        $this->accessSecrec = $accessSecrec;
        $this->DomainName = $domain;
```

```
}

public function DescribeDomainRecords() {
    $requestParams = array(
        "Action" => "DescribeDomainRecords"
    );
    $val = $this->send($requestParams);
    return $this->out($val);
}


public function UpdateDomainRecord($id, $type, $rr,$value){
    $requestParams = array(
        "Action" => "UpdateDomainRecord",
        "RecordId" => $id,
        "RR" => $rr,
        "Type" => $type,
        "Value" => $value,
    );
    $val = $this->send($requestParams);
    return $this->out($val);
}
public function DeleteDomainRecord($id) {
$requestParams = array(
        "Action" => "DeleteDomainRecord",
        "RecordId" => $id,
    );
    $val = $this->send($requestParams);
    return $this->out($val);
}

public function AddDomainRecord($type, $rr, $value) {
    $requestParams = array(
        "Action" => "AddDomainRecord",
        "RR" => $rr,
        "Type" => $type,
        "Value" => $value,
    );
    $val = $this->send($requestParams);
    return $this->out($val);
}

private function send($requestParams) {
    $publicParams = array(
    "DomainName" => $this->DomainName,
    "Format" => "JSON",
    "Version" => "2015-01-09",
    "AccessKeyId" => $this->accessKeyId,
    "Timestamp" => date("Y-m-d\TH:i:s\Z"),
    "SignatureMethod" => "HMAC-SHA1",
    "SignatureVersion" => "1.0",
    "SignatureNonce" => substr(md5(rand(1, 99999999)), rand(1, 9), 14),
    );
```

```
        $params = array_merge($publicParams, $requestParams);
        print_r($params);
    $params['Signature'] = $this->sign($params, $this->accessSecrec);
        $uri = http_build_query($params);
        $url = 'http://alidns.aliyuncs.com/?'.$uri;
        return $this->curl($url);
    }


    private function sign($params, $accessSecrec, $method = "GET") {
        ksort($params);
        $stringToSign = strtoupper($method).'&'.$this->percentEncode('/').'&';
        $tmp = "";
        foreach($params as $key => $val){
            $tmp .= '&'.$this->percentEncode($key).'='.$this->percentEncode($val);
        }
        $tmp = trim($tmp, '&');
        $stringToSign = $stringToSign.$this->percentEncode($tmp);
        $key = $accessSecrec.'&';
        $hmac = hash_hmac("sha1", $stringToSign, $key, true);
        return base64_encode($hmac);
    }

    private function percentEncode($value = null){
        $en = urlencode($value);
        $en = str_replace("+", "%20", $en);
        $en = str_replace("*", "%2A", $en);
        $en = str_replace("%7E", "~", $en);
        return $en;
    }

    private function curl($url) {
        $ch = curl_init();
        curl_setopt($ch, CURLOPT_URL, $url );
        curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1 );
        $result = curl_exec ($ch);
        curl_close($ch);
        return $result;
    }

    private function out($msg) {
        return json_decode($msg, true);
    }
}
```

以上代码来自 https://github.com/ywdblog/certbot-letencrypt-wildcardcertificates-alydns-au 改得到。

有了这两个脚本，我们就可以通过crontab 每月定时执行一次续期，并重新加载nginx配置。续期命检测到续期时间小于30天时，会重新请求生成新证书