链滴

短信验证码防攻击策略

作者：Changer0914
原文链接：https://ld246.com/article/1513144889781
来源网站：链滴

虽然短信验证码是现在各大网站和APP等平台向用户提供的一种信息验证的方式，但是随着技术的提，黑客也是越来越高级，各种木马病毒成为了短信验证码的一个拦截漏洞，特别是很多的网购平台，是存在着被刷的风险。

因此不管是企业还是短信验证码服务商，都在思考相应的对策，来避免我们的短信验证码被攻击。为防止短信验证码被攻击，可以通过以下的一些方式去避免。

1、前端增加图文验证码

在获取短信验证码前增加图文验证码是一种较为常见的方式。攻击者一般是采用自动化攻击，增加图验证码后，攻击者要对验证进行识别验证成功后才能进行模拟用户发送请求，这一步需要在页面中进，无法采用自动化攻击。第一种攻击基本上失效，同时会增加第二种的攻击成本（有可能采用人工打方式进行验证）选择验证码时既要考虑用户操作过程的流畅度，又应该考虑到安全度。

2、限制单个手机号每日接收短信次数和时间间隔

对单个手机号进行日接收次数的限制，可以防止单个手机号无限制刷短信，同时设置时间间隔可以有，防止人工刷票。短信接收次数可以根据平台特点进行限制，一般日接受验证码次数为10次左右；同号码发送时间间隔通常为60秒，前后台应保持一致，避免出现只前端做倒计时限制，后台未做限制这低级错误。

3、对IP进行限制

对单IP最大发送量进行限制，可以有效防止单一IP下多手机号被刷的问题。最大发送量限制是防止恶攻击者同IP下不同手机号进行刷短信验证码行为。根据平台实际情况设计一个短信最大发送量的阀值超过阀值将不予返回短信。

```
public class IpAdrressUtils {

    /**
     * 获取真实IP
     *
     * @param request
     * @return IP
     */
    public static String getIpAdrress(HttpServletRequest request) {
        String Xip = request.getHeader("X-Real-IP");
        String XFor = request.getHeader("X-Forwarded-For");
        if (StringUtils.isNotEmpty(XFor) && !"unKnown".equalsIgnoreCase(XFor)) {
            // 多次反向代理后会有多个ip值，第一个ip才是真实ip
            int index = XFor.indexOf(",");
            if (index != -1) {
                return XFor.substring(0, index);
            } else {
                return XFor;
            }
        }
        XFor = Xip;
        if (StringUtils.isNotEmpty(XFor) && !"unKnown".equalsIgnoreCase(XFor)) {
            return XFor;
        }
        if (StringUtils.isBlank(XFor) || "unknown".equalsIgnoreCase(XFor)) {
            XFor = request.getHeader("Proxy-Client-IP");
        }
```

```
        if (StringUtils.isBlank(XFor) || "unknown".equalsIgnoreCase(XFor)) {
            XFor = request.getHeader("WL-Proxy-Client-IP");
        }
        if (StringUtils.isBlank(XFor) || "unknown".equalsIgnoreCase(XFor)) {
            XFor = request.getHeader("HTTP_CLIENT_IP");
        }
        if (StringUtils.isBlank(XFor) || "unknown".equalsIgnoreCase(XFor)) {
            XFor = request.getHeader("HTTP_X_FORWARDED_FOR");
        }
        if (StringUtils.isBlank(XFor) || "unknown".equalsIgnoreCase(XFor)) {
            XFor = request.getRemoteAddr();
        }
        return XFor;
    }

}
```

4、对注册流程进行限定

一般来讲常被攻击的地方是注册页面，一般是从两方面进行触发流程的限定。第一种，可以从前端写指令，只允许在官网主页跳转入注册页面；第二种方法是对注册流程进行分步，先进行账户密码设置设置成功后才可以再进行下一步的短信验证。因为增加前置条件，增加攻击难度两种方法都是能有效止自动化攻击，需要注意的是两种方法对用户体验或多或少的影响，产品经理需要结合自身平台特点择。

5、对发送者进行唯一性识别

为了防止第二种恶意攻击者通过修改传向服务器各项参数，造成多IP多手机号刷短信验证码的行为，以后台应对前台传过来的参数进行验证。方法一般是用token作为唯一性识别验证，后台写一个算法将oken注入到前端，然后前端可以通过相应的规则获取到token，在发送短信验证请求接口数据时带上token，在后端对token进行验证，验证通过才能正常将短信发送。

通常来说，前三种方式是大部分企业使用的，因为这几种方式基本可以防止大部分恶意刷短信验证码行为。防止短信验证码被刷，还是要结合自身产品特点，要未雨绸缪不应到短信被刷才进行防护。

给裸接口加一道防护，避免恶意盗刷和爬取