**CVE-2012-4792漏洞利用分析**

**0x00 调试环境**

|  |  |
| --- | --- |
| OS: | win7 x86 |
| IE | 8.0.7601.17514 |

**0x01 漏洞原理**

**1. poc**

|  |
| --- |
| <!doctype html>  <html>  <head>  <script>  function exploit() {  var e0 = null;  var e1 = null;  var e2 = null;  try {  e0 = document.getElementById("a");  e1 = document.createElement("div");  e2 = document.createElement("q");  e1.applyElement(e2);  e1.appendChild(document.createElement('button'));  e1.applyElement(e0);  e2.innerHTML = "";  e2.appendChild(document.createElement('body'));  } catch (e) {}  CollectGarbage();  }  </script>  </head>  <body onload="exploit()">  <form id="a">  </form>  </body>  </html> |

在e2.innerHTML = ""前，DOM树结构：

e0: CFormElement

e1: CDivElement

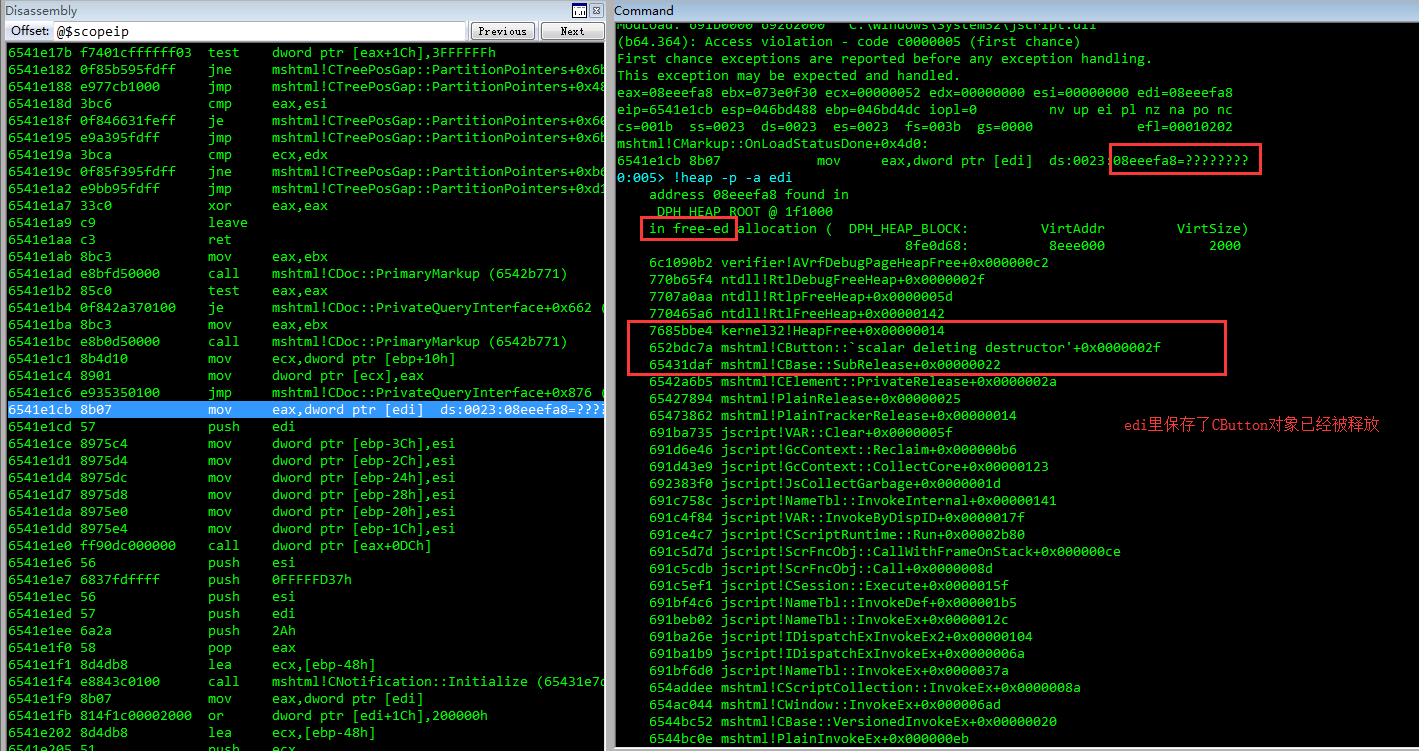
e2: CPhraseElement

CButton

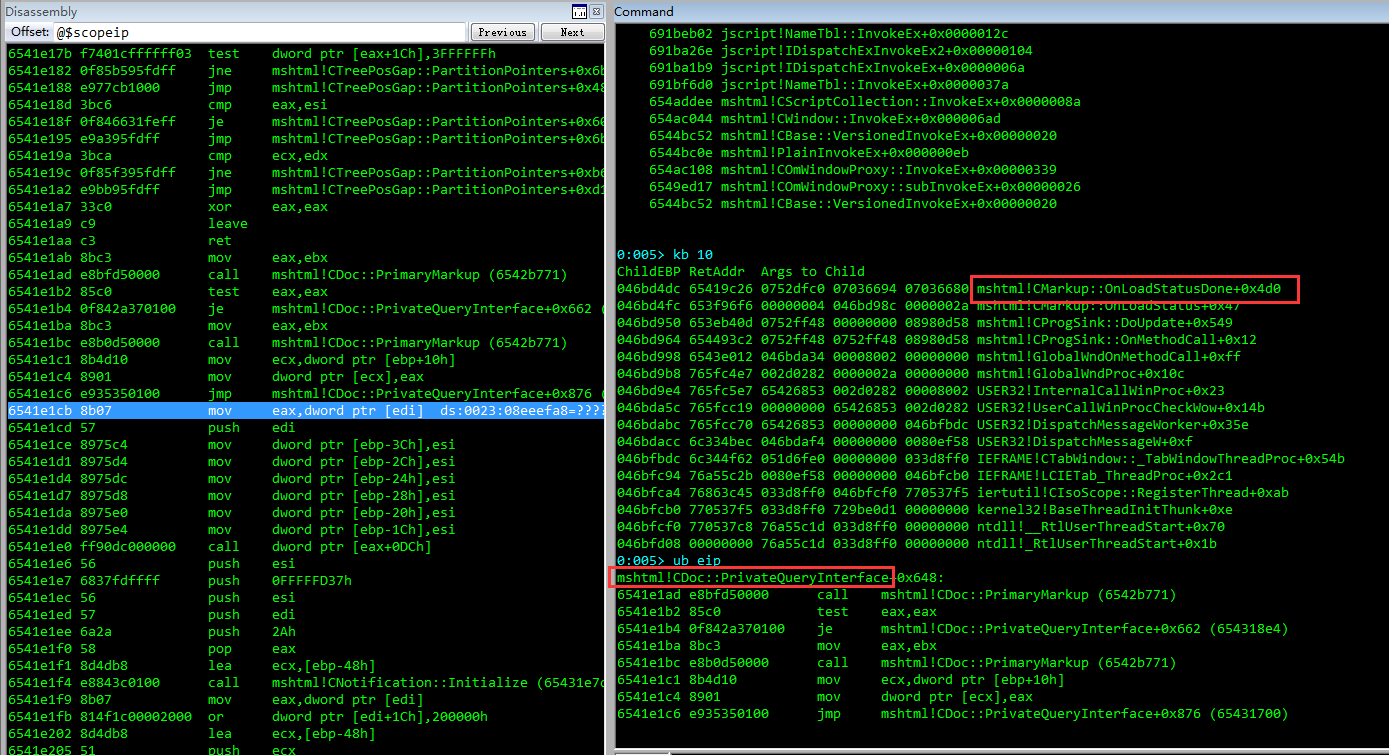
CRootElement

**2. 触发条件**

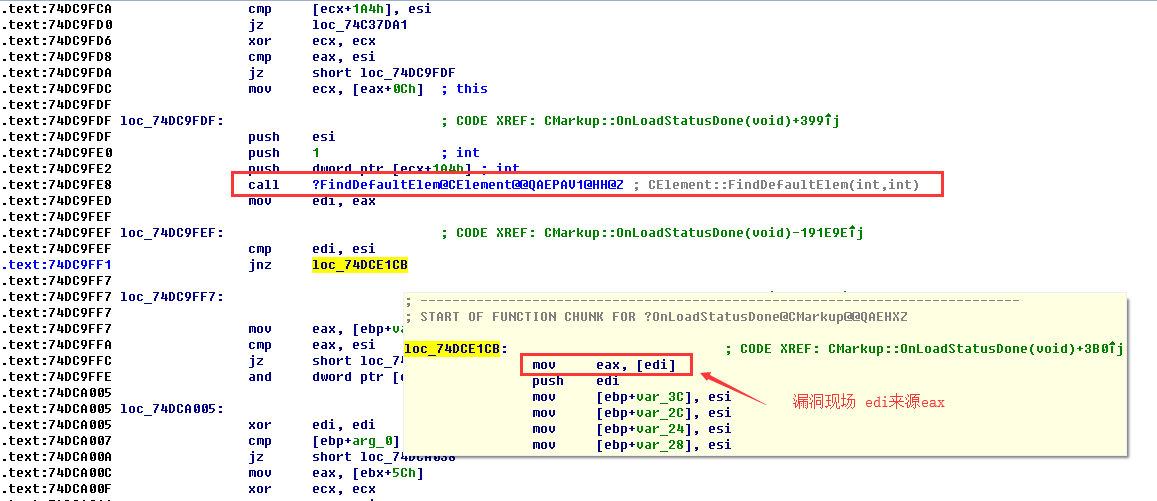
开启hpa和ust，打开页面，ie crash：



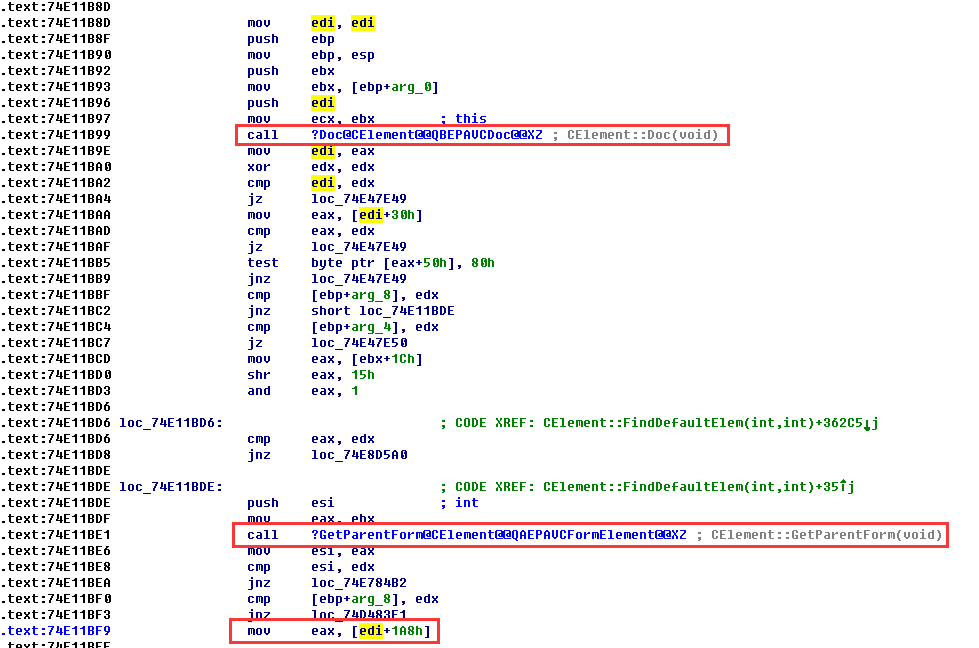
此时edi保存的CButton对象已经被释放。观察调用栈：



漏洞现场是在CMarkup::OnLoadStatusDone+0x4d0，但是ub eip却在CDoc::PrivateQueryInterface所以CMarkup::OnLoadStatusDone+0x4d0前是一处跳转，用IDA查看CMarkup::OnLoadStatusDone函数：

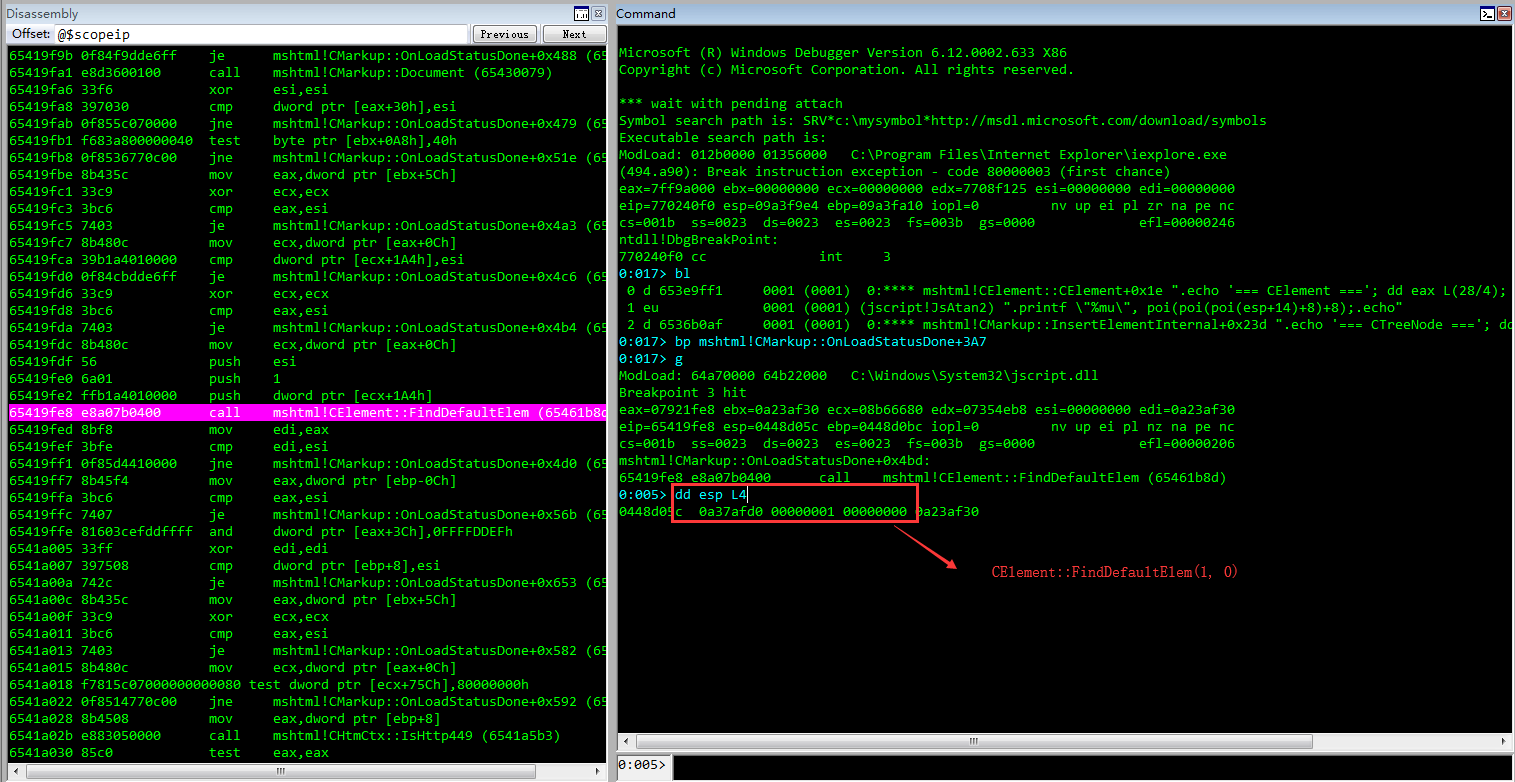


在漏洞现场不远有一处函数调用CElement::FindDefaultElem，保存了已经释放的CButton对象的edi就来自该函数的返回值eax，跟进观察CElement::FindDefaultElem函数：



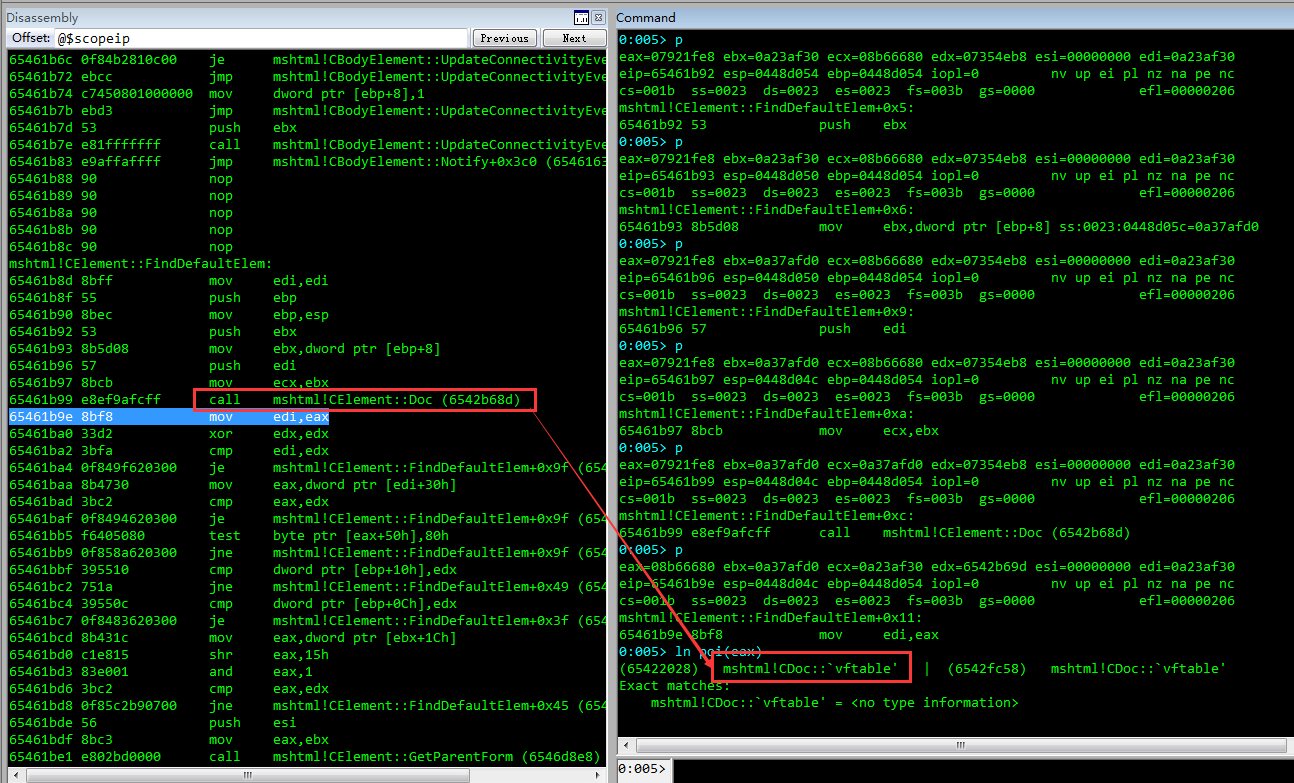
首先通过CElement::Doc(void)获取当前的CDoc对象保存在edi中，然后通过CElement::GetParentForm获取ParentForm再进行比较，最后将edi+1A8h的值保存到eax返回。动态调试，在.text:74DC9FE8 call ?FindDefaultElem@CElement@@QAEPAV1@HH@Z ; CElement::FindDefaultElem(int,int)处下断点：

bp mshtml!CMarkup::OnLoadStatusDone+3A7

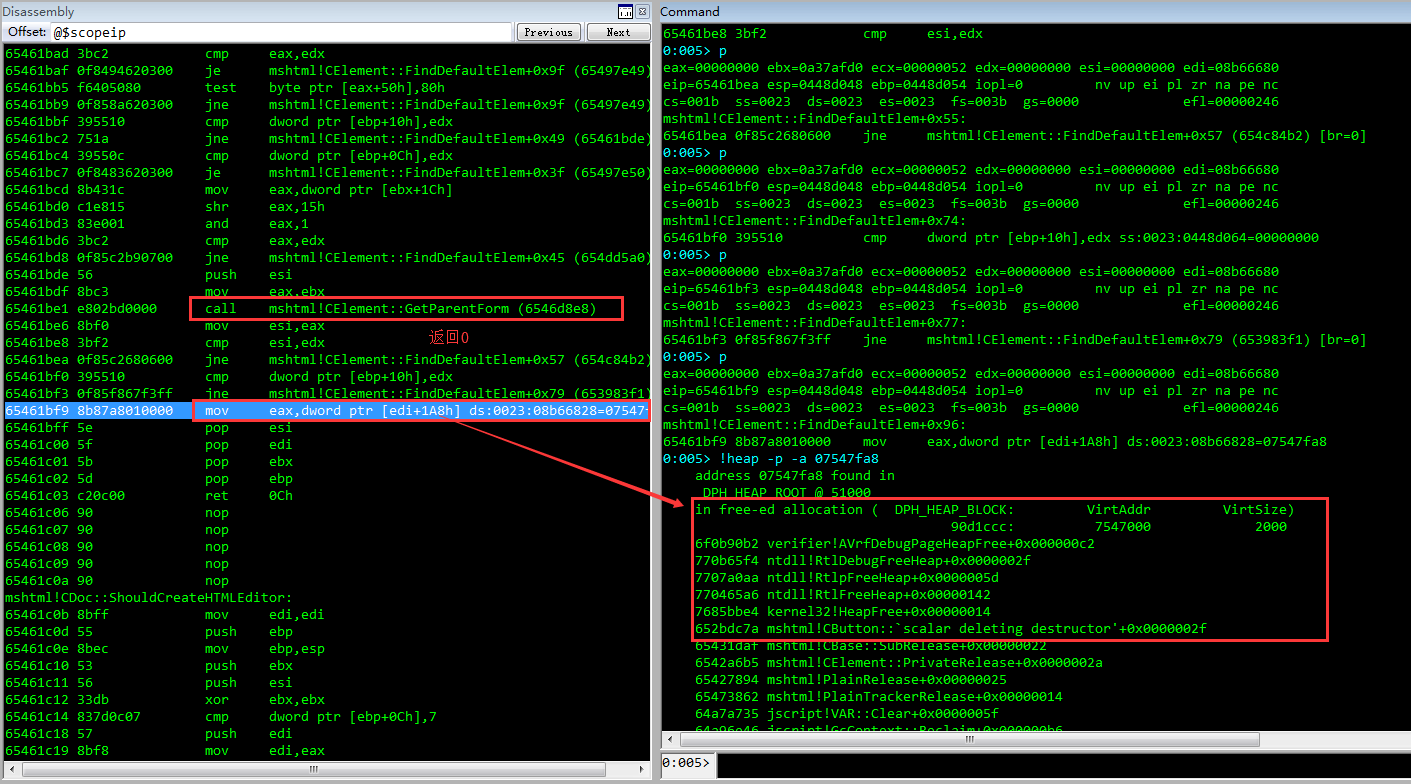


跟进CElement::FindDefaultElem：

1. CElement::Doc(void)获取当前的CDoc对象保存在edi：



1. CElement::GetParentForm返回0，并最终返回[edi+1A8]的值，这里[edi+1A8]保存的就是已经被释放的Cbutton对象：



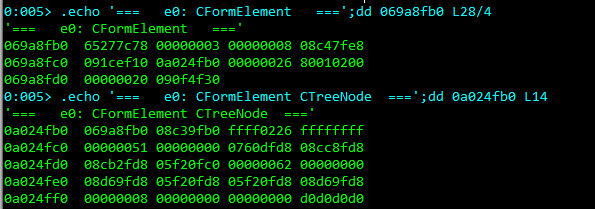
总结一下就是在body标签的onload事件完成时会调用CMarkup::OnLoadStatusDone，然后会调用CElement::FindDefaultElem查找当前CDoc对象的默认元素，获取当前CDoc对象的DefaultElem成员变量时，因为CElement::GetParentForm返回0，从而返回的是记录在CDoc对象的DefaultElem成员变量也就是已经被释放的CButton对象指针，最终造成了UAF漏洞。

**3. 元素的创建和释放**

在e2.innerHTML = "";后会将e2的子元素(CformElement, CdivElement, CButton)从DOM树上卸载下来，也就是将其对应的CTreeNode删除。

**e2.innerHTML = ""前元素创建：**

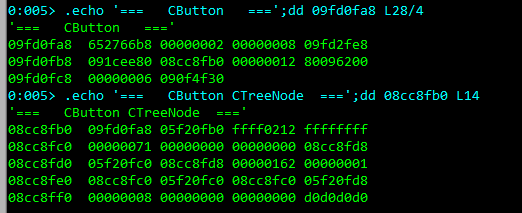
e0: CFormElement创建：



e1: CDivElement 创建：

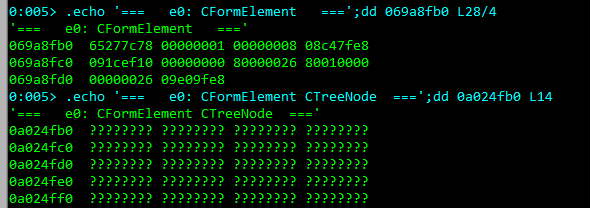


CButton创建：

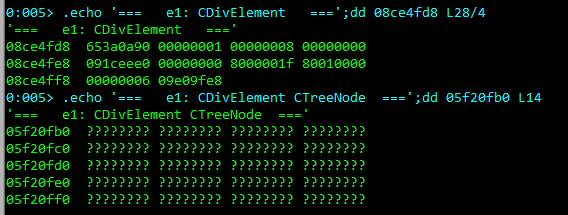


**e2.innerHTML = ""后元素的CTreeNode被释放创建**：

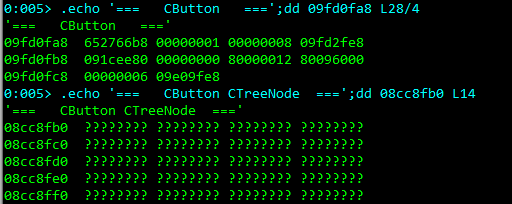
e0: CFormElement CTreeNode释放：



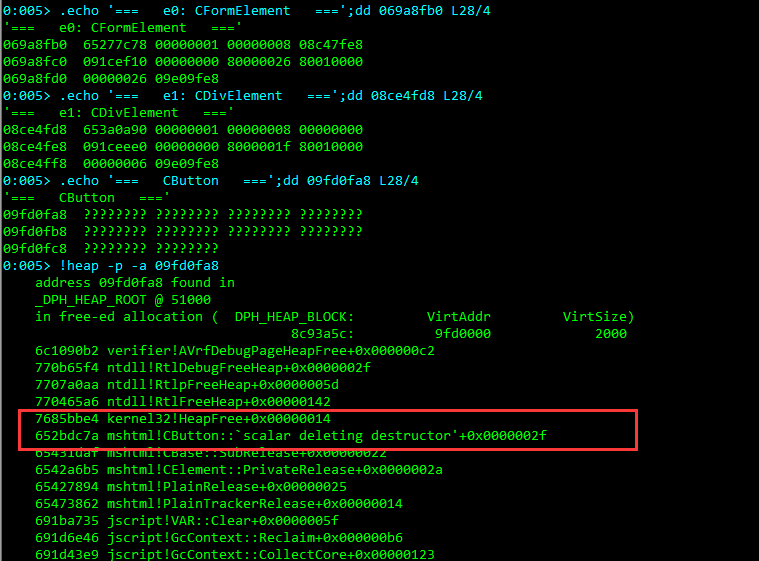
e1: CDivElement CTreeNode释放：



CButton CTreeNode释放：



**CollectGarbage()后元素被释放：**



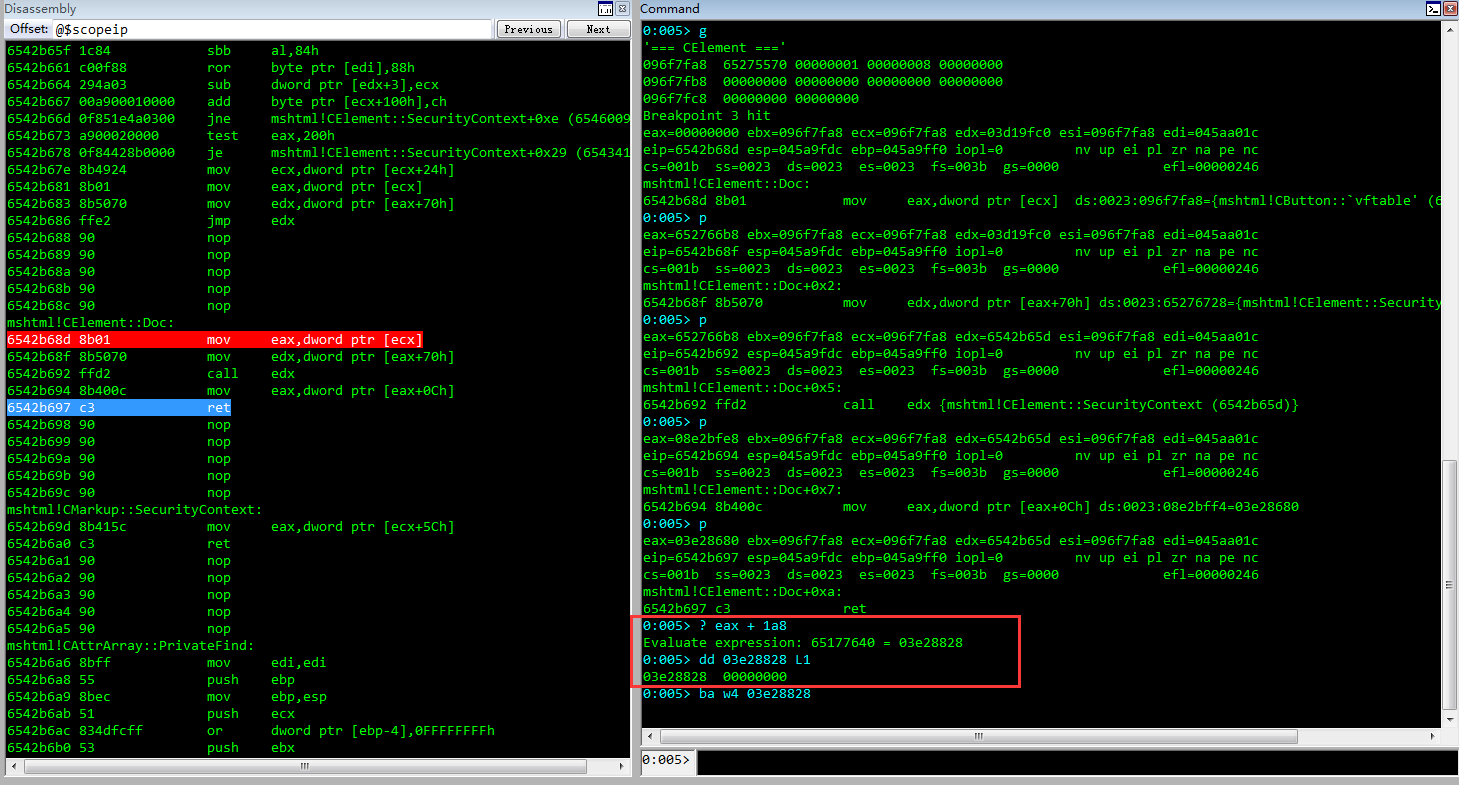
这里可以看到CollectGarbage()后只有CButton对象被释放了，如果把poc中的：

e2.appendChild(document.createElement('body'));注释掉，会发现CButton对象也不会被释放，同时ie也不会crash，因此是在e2.appendChild(document.createElement('body'))后会释放CButton内存。

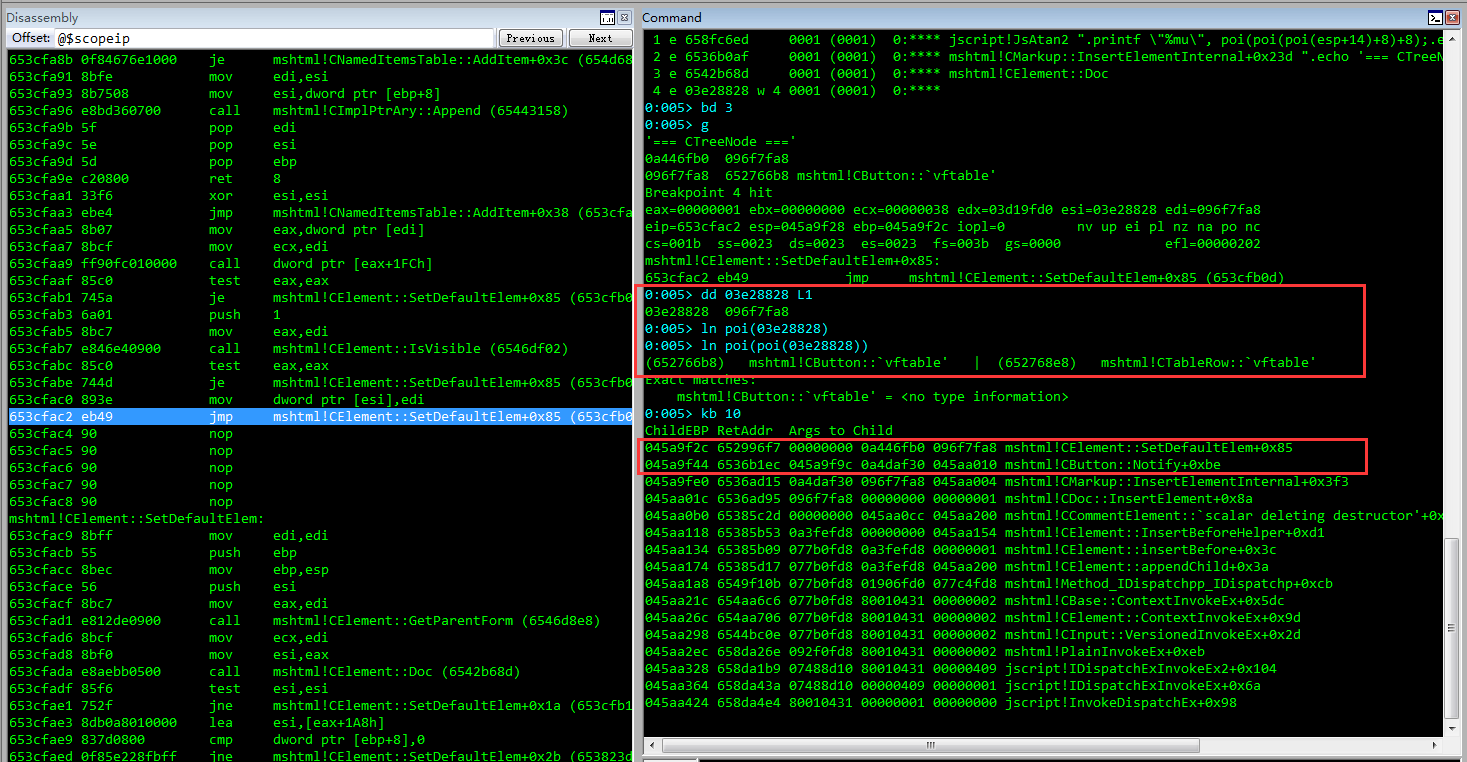
**4. 漏洞本质**

通过前面的分析可以知道在CollectGarbage()后CButton对象会被释放的，但是Onload事件依然会从CDoc对象的DefaultElem成员变量取出该CButton对象，下面跟踪CButton对象是什么时候赋值给CDoc对象的DefaultElem成员变量的。

首先观察CDoc对象，在mshtml!CElement::Doc()处下断点：



此时DefaultElem还是空，对这个地址下写断点：



可以看到在将CButton挂到DOM树上时会调用CButton::Notify-> CElement::SetDefaultElem将CButton对象设置为CDoc的DefaultElem，而当CButton对象被删除后CDoc的DefaultElem仍然保存了CButton对象的引用，最终在CMarkup::OnLoadStatusDone时再次引用了该地址，造成UAF漏洞。

所以漏洞来源于CElement::SetDefaultElem函数，跟入这个函数分析下逻辑：



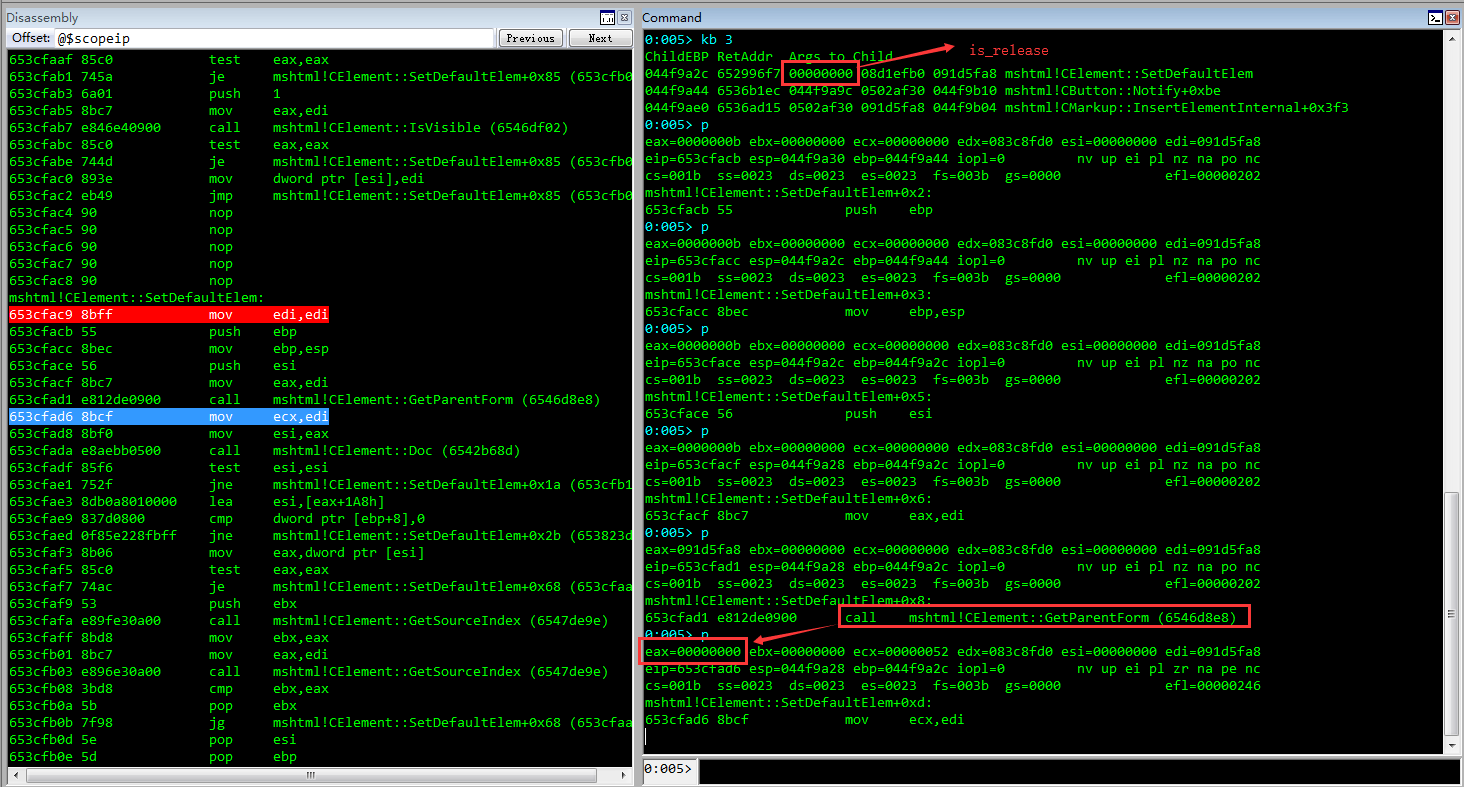
该函数有两个参数：this\_element和is\_release标志。

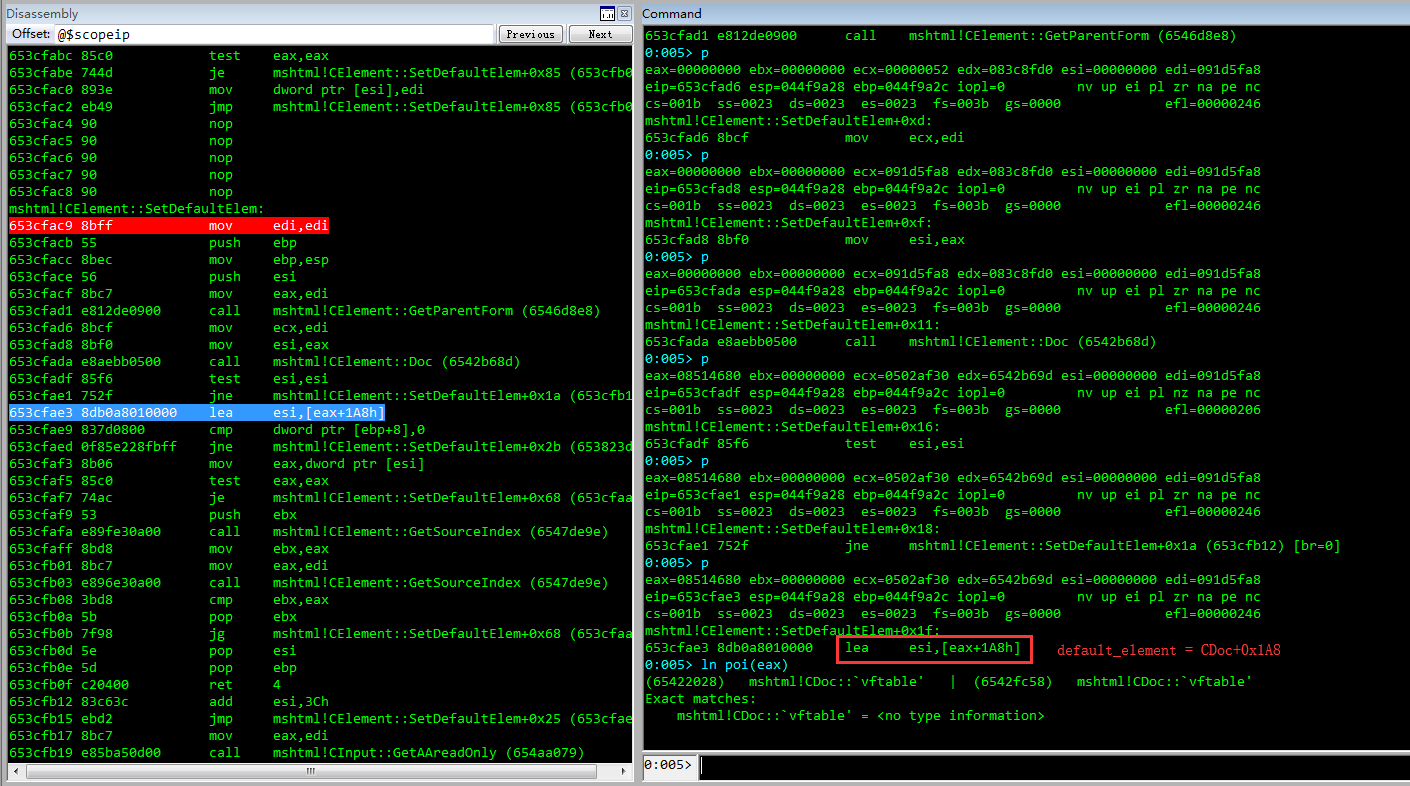
首先通过CElement::GetParentForm(this)取出当前element的ParentForm对象，如果存在则将parentForm对象+0x3C的指针赋值给default\_element，如果不存在parentForm对象则取出CDoc对象+0x1A8的指针赋值给default\_element。

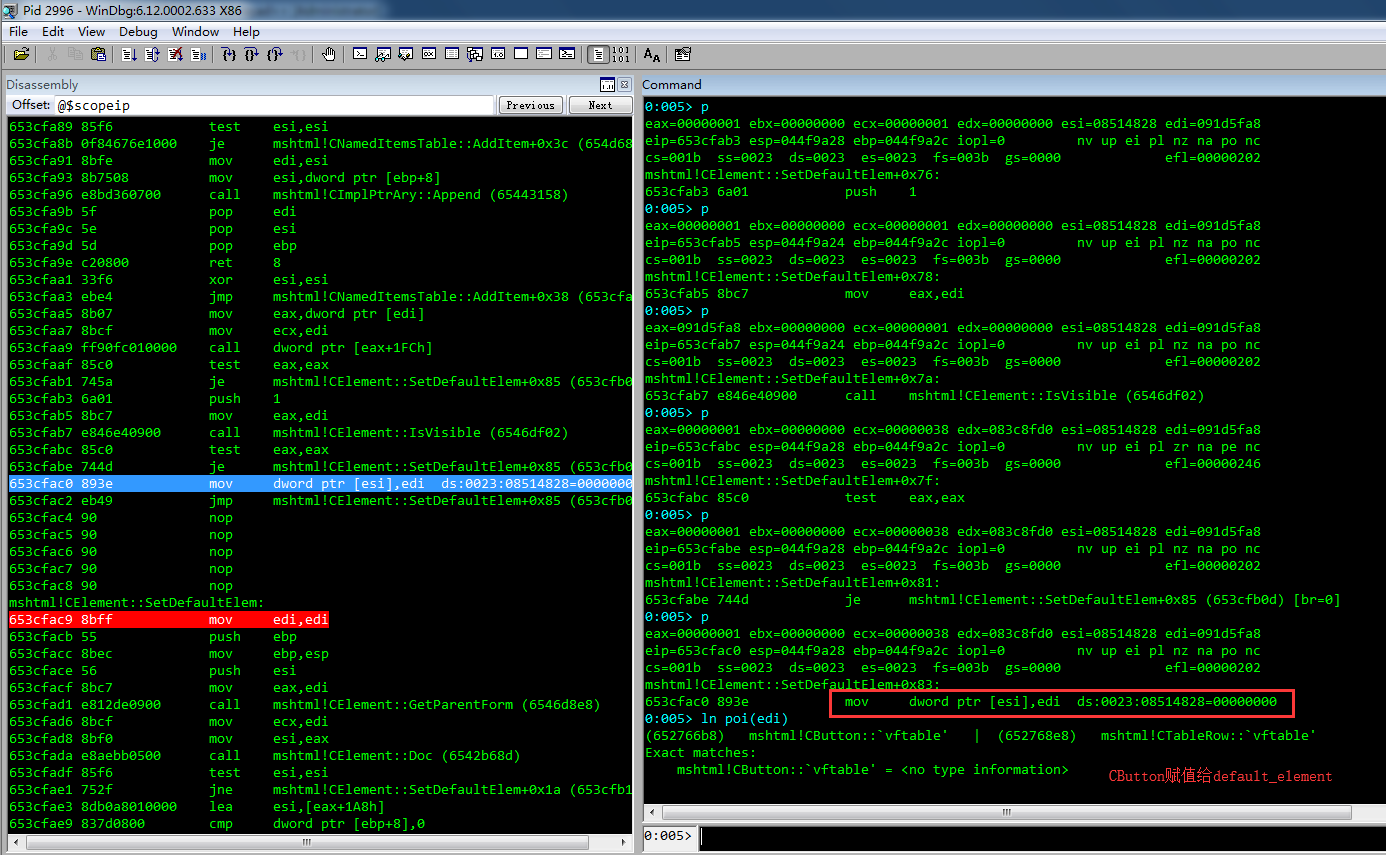
然后判断CElement::SetDefaultElem是否时元素释放时调用的（0 == is\_element），如果是则判断default\_element和this\_element是否相等，不等则将default\_element置0，否则通过CElement::FindDefaultElem查找DefaultElem并赋值给default\_element。

如果CElement::SetDefaultElem不是元素释放时调用的（1 == is\_element），则将this\_element赋值给default\_element。

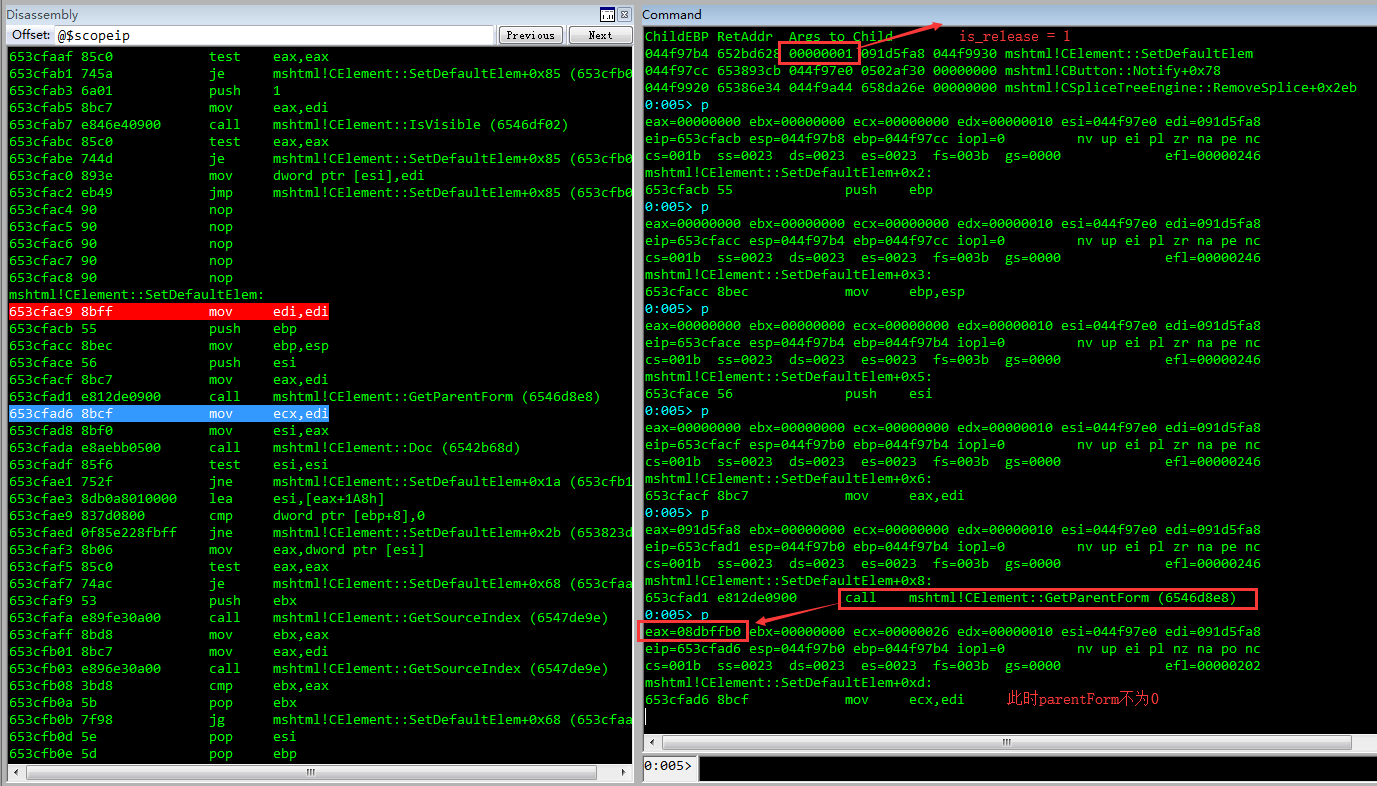
对于POC，首先在“e1.appendChild(document.createElement('button'));“会调用CElement::SetDefaultElem，并且is\_release = 0：

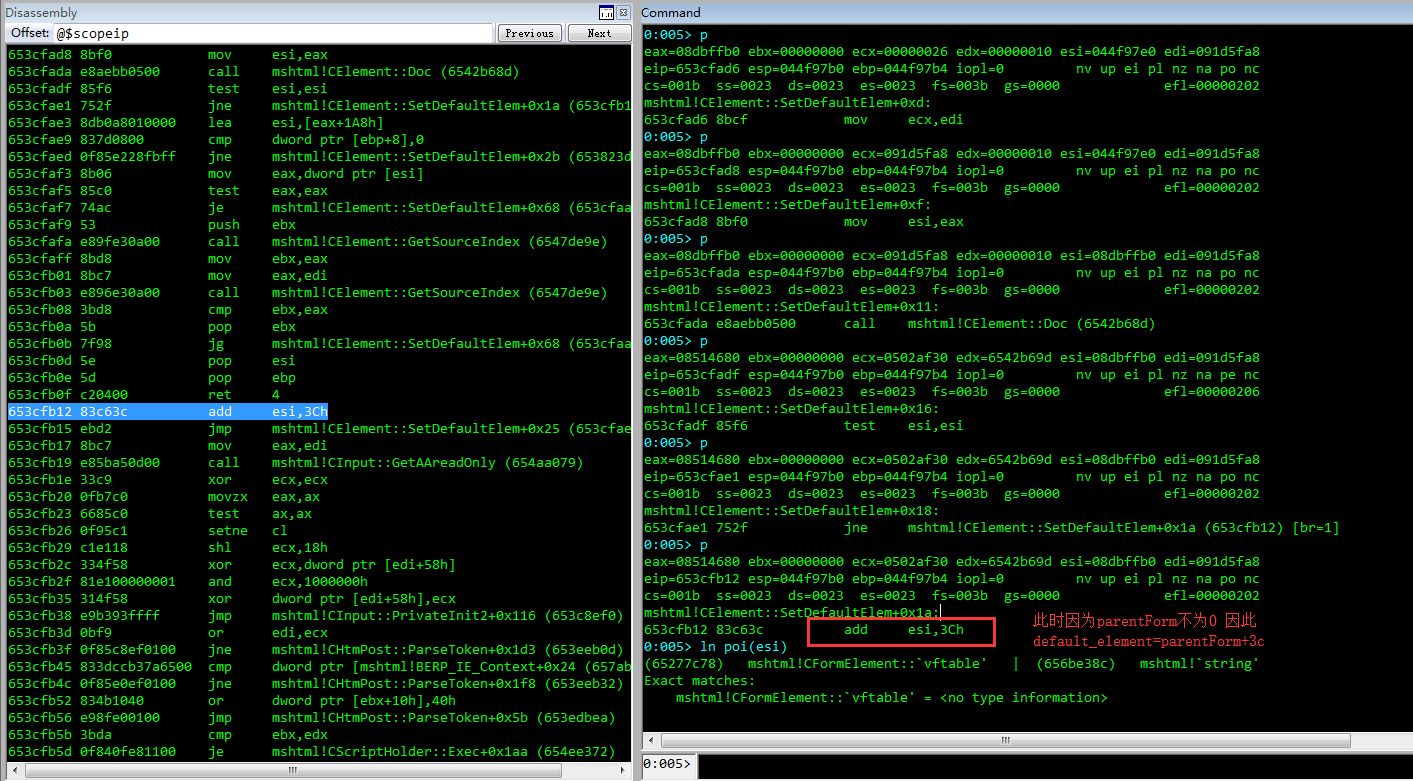


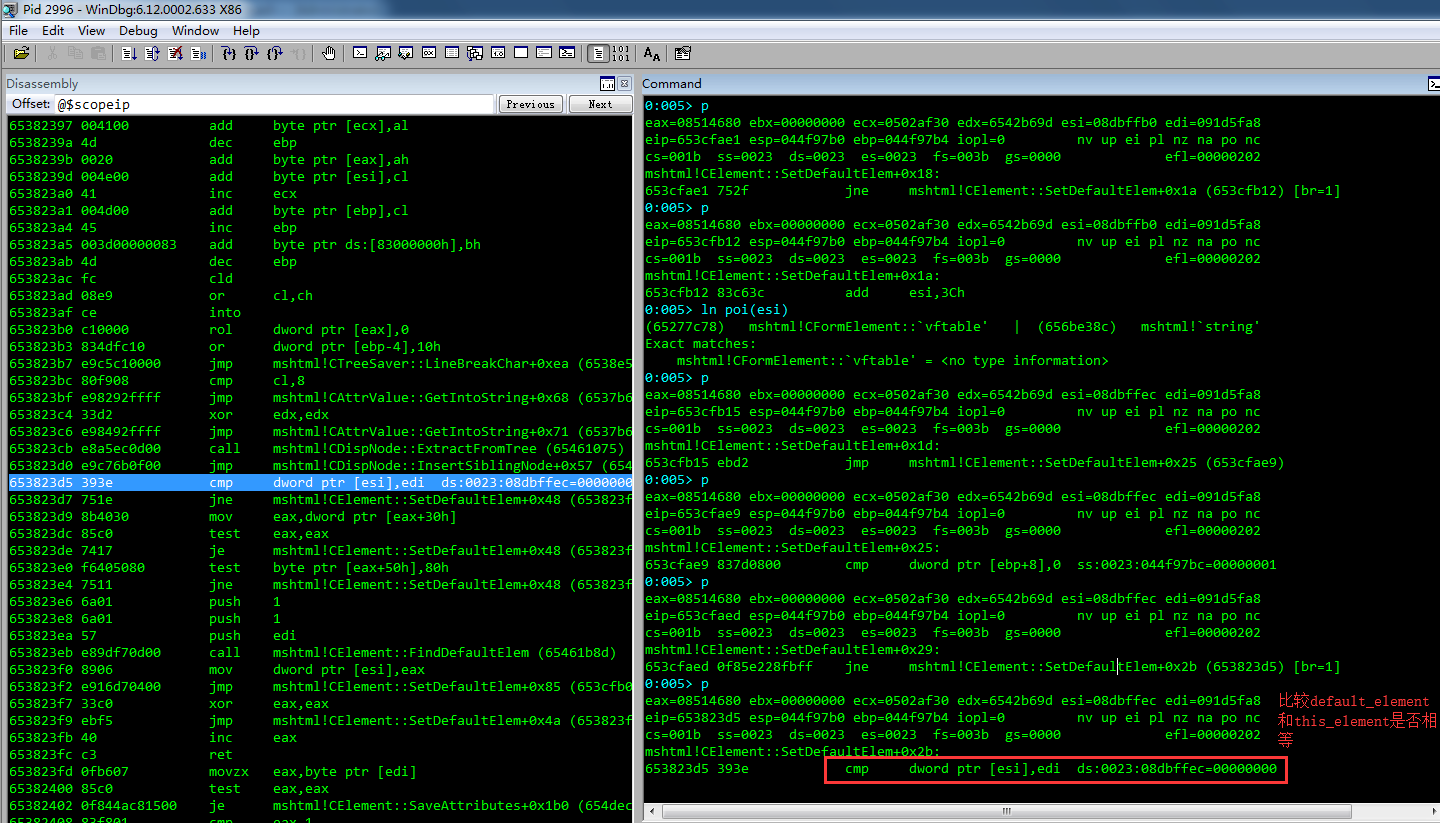


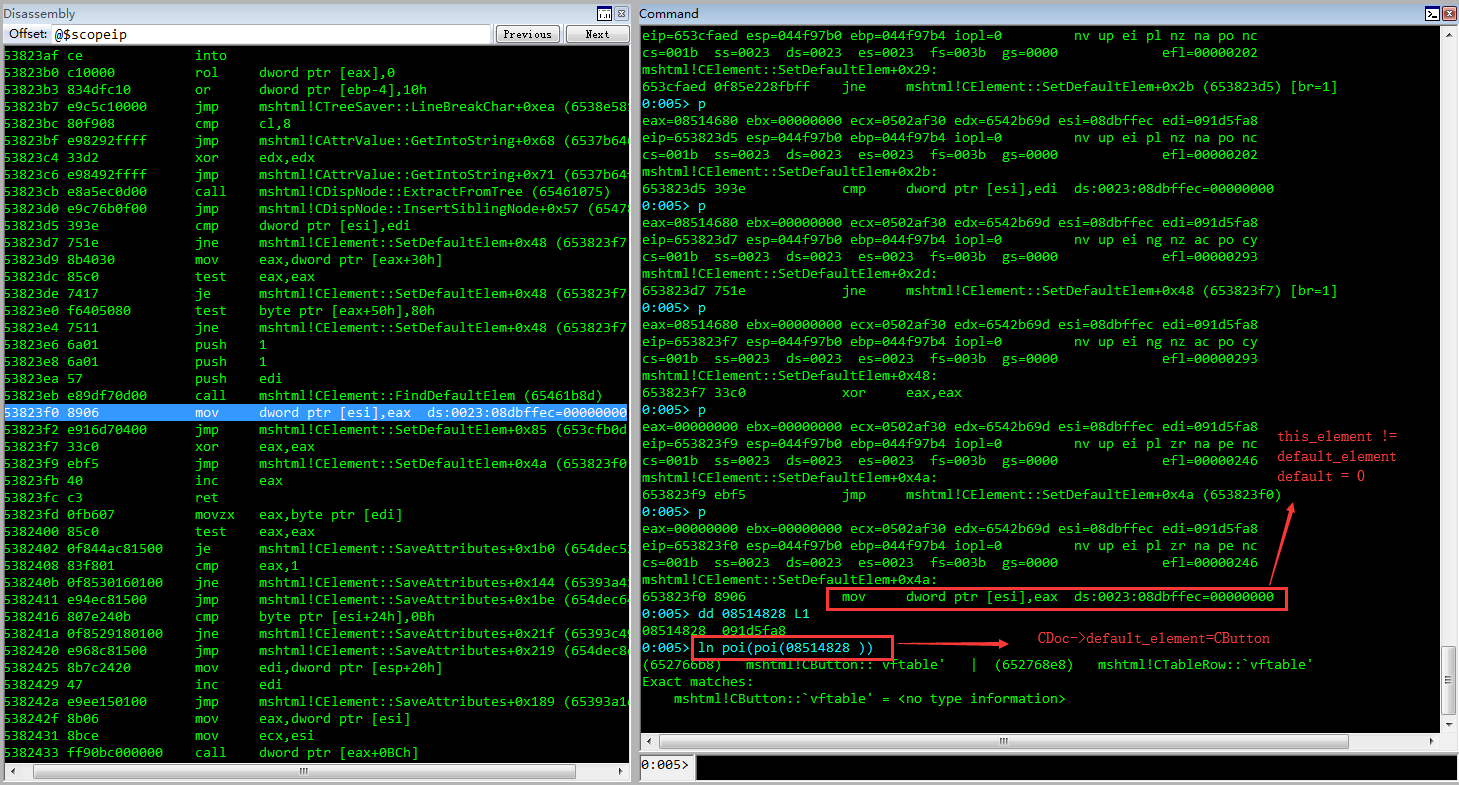


接着在e2.innerHTML = "";会再次调用CElement::SetDefaultElem，并且is\_release = 1：









因此，漏洞的本质是在创建CButton的DOM树时，会在父节点的default\_element保存该CButton的引用，由于POC中创建CButton是父节点不是Form，因此在CDoc的default\_element保存该CButton的引用，但是随后POC中的e1.applyElement(e0);会为CButton对象的DOM树中添加父节点Form，从而在e2.innerHTML = "";后错误的释放了parentForm的default\_element，而CDoc的default\_element仍然保存该CButton的引用，最终在body标签的onload事件完成时调用CMarkup::OnLoadStatusDone，会调用CElement::FindDefaultElem查找当前CDoc对象的默认元素，获取当前CDoc对象的DefaultElem成员变量时，因为CElement::GetParentForm返回0，从而返回的是记录在CDoc对象的DefaultElem成员变量也就是已经被释放的CButton对象指针，最终造成了UAF漏洞。