WebShells: survey and development of a framework for penetration testing
The Boring part ;)

- **Elena Kropochkina** elena[.]kropochkina[at]gmail[.]com
  - Elena Kropochkina begins her professional career in Devoteam Audit Security team. She was graduated by Ecole Polytechnique and Telecom ParisTech (France) with a M.S. in Computer Science, by Novosibirsk State University (Russia) with B.S. in Mathematics. She is specialized in IT Security.

- **Joffrey Czarny** joffrey[.]czarny[at]devoteam[.]com
Disclaimer

- The presented study is in order to carried out Ethical Hacking

- Some tools presented in this slide maybe Unlawful in some country

- Locale legislation must be apply
Summary

Problematic & Objective

State of Art
- Environment study
- WebShell survey
- Obfuscation and protection tools

Conception

Proof-of-concept
- Pieces of code
- Demonstration

Conclusion & perspectives
Summary

Problematic & Objective

State of Art
- Environment study
- WebShell survey
- Obfuscation and protection tools

Conception

Proof-of-concept
- Pieces of code
- Demonstration

Conclusion & perspectives
Problematic and Objective

■ Context
  • As many pentester, we had some small WebShells “PHP, Java, ASP” quickly developed in order to control and escalate his privileges on compromised server during penetration testing
  • Lots of public Webshell are detected and blocked by some security product so unusable in the real life.

■ Develop an intrusion Webshell toolkit
  • Standardized and centralized Webshells
  • Add obfuscation features and tried to bypass IPS/WAF signatures

■ Followed Steps:
  • State of existing Webshells and their specificities related to the different platforms
  • Study of obfuscation methodologies on Web languages (PHP,ASP and Java)
  • Define a master Webshell with his primary modules, features needed (needful) and interesting features (nice to have)
  • Development of master
  • Development modules for each Web languages
Summary

Problematic & Objective

State of Art
- Environment study
- WebShell survey
- Obfuscation and protection tools

Conception

Proof-of-concept
- Pieces of code
- Demonstration

Conclusion & perspectives
Environment study

Web server types

Web server market by Netcraf

Web server market by W3Tech, February 2011
Environment study

OS types on the web servers

Operating system share for SSL sites, to January 2009, Netcraft

Operating system share for Apache serveurs, Security Space
Programmation language share for web servers

Programmation language use on the web servers, to February 2011, by W3Techs
Environment study

- Most used environments
  - Apache+PHP; IIS +ASP.NET; Tomcat, Weblogic +Java

- Pentester's feedback
  - Java dominance (often used for internal application of big companies)

- Priorities identified
  - Apache + PHP
  - Tomcat, Weblogic +Java
  - IIS + ASP.NET
Summary

Problematic & Objective

State of Art
- Environment study
- **WebShell** survey
- Obfuscation and protection tools

Conception

Proof-of-concept
- Pieces of code
- Demonstration

Conclusion & perspectives
WebShell survey

Most famous WebShells:
- C99 Shell
- FaTaListiCz

Most interesting WebShells (Ethical Hacking point of view):
- PHP
  - C99 Shell
  - FaTaListiCz
  - NFM
  - R57
  - Iron Shell
  - PHPJackal
- Java
  - JspWebshell
  - JspSpy
- ASP
  - Zehir 4
  - ASP Spyder

Antivirus detection tests:
- McAfee
- Kaspersky
- VirusTotal
WebShell survey: C99 Shell

- File Manager, "user-friendly" interface
- File upload/download
- File Editor
- Command execution cmd
- Information on open ports
- Encoding/decoding in base64, dec2hex, URL, hash : md5, sha1, crypt, crc32
- Actif process control
- Binding port connection
- Security information (safe mode on/off, open databases)
- FTP Client
- Advanced SQL Manager, like phpMyAdmin
- PHP code evaluation

Antivirus detection:
- Kaspersky: Backdoor.php.WebShell.bb
- McAfee: BackDoor-DNF
- VirusTotal: 23/43
WebShell survey: C99 Shell
WebShell survey: FaTaLisTiCz_Fx29 Shell

- System information
- File Manager "user-friendly"
- File upload/download.
- Search of folders with writing rights
- File editor
- Command execution cmd
- Encoding/decoding base64, dec2hex, URL; hashage : md5, sha1, crypt, crc32
- Actif process control
- Advanced SQL Manager, like phpMyAdmin. Possible to view open SQL connections and SQL server environment variables
- Code PHP evaluation
- Mail sending
- Update from author site and feedback

- Antivirus Detection
  - Kaspersky: Trojan-Downloader.PHP.Small.i
  - McAfee: is not detected
  - VirusTotal: 8/43
WebShell survey: PHP Shell 2.1

- Command execution cmd
- File navigation by command line

- Authentification:
  - password hashed by md5 or shal

- Antivirus Detection:
  - Not detected
  - VirusTotal 0/43
WebShell survey: PHPJackal

- System information
- File Manager "user-friendly"
- File upload/download.
- File editor
- Command execution cmd
- Encoding/decoding base64, dec2hex, binary, hash: md5, sha1
- Port scanner, sub-domains and folder scanner
- Advanced SQL Manager
- Code PHP evaluation
- Mail sending
- FTP Client
- Bind/reverse shell
- Password cracker and brute force on hash by Dictionaries or log files
- Steganography
- Web proxy

- Antivirus Detection:
  - Not really detected „only Avast“, virustotal 3/41
WebShell survey: Iron Shell

- PHP information
- File Manager
- File upload/download.
- File editor
- Command execution cmd
- SQL Requests without data return
- Code PHP evaluation
- Brute force on md5 hash
- HTTP headers view

- Antivirus Detection
  - Detected
  - VirusTotal 16/41

- Authentification:
  - nothing by default, possible with password hashed by md5
WebShell survey: R57 Shell

- File upload/download (direct or by FTP).
- Command execution cmd
- MySQL Manager, MySQL dump
- Code PHP evaluation
- Mail sending
- Text search in files

- Antivirus Detection
  - Kaspersky: Backdoor.PHP.Rst.ai
  - McAfee: is not detected
  - VirusTotal: 7/42

- Authentication:
  - Password and login hashed by md5
WebShell survey: NFM – Network File Manager

- File Manager
- File upload/download.
- Command execution cmd
- Hash : md5, sha1
- Port scanner
- MySQL Manager
- Archiving
- Mail ICQ flood
- FTP Client
- Brute force on md5 hash
- Visualisation etc/passwd, cpanel.log, httpd.conf
- Exploits (bash shell bindtt.c, Local ROOT for linux 2.6.20 - mremap, Local ROOT for linux 2.6.20 - ptrace, psyBNC 2.3.2-4,BRK - Local Root Unix 2.4., Glftpd DupeScan Local Exploit, Traceroute v1.4a5 exploit by sorbo, Traceroute v1.4a5 exploit by sorbo

- Antivirus Detection
  - Kaspersky : Backdoor.PHP.NFMshell.a
  - McAfee: not detected
  - VirusTotal: 8/39
  - Comments: in Russian, adapted only for Linux, authentication was not working
WebShell survey: JspWebShell

- File Manager
- File upload/download.
- File editor
- Command execution cmd

- Comments: in Chinese

- Antivirus Detection
  - Not really detected
  - SuspectCRC
  - VirusTotal: 2/41

- Authentication: password
WebShell survey: JspSpy

- System and java information
- File Manager
- File upload/download.
- File editor
- Command execution cmd and program execution
- Port scanner
- SQL Manager without grafic interface (Oracle, MySQL, SQL Server, Access)
- Back connection
- Dynamic screenshots

- Antivirus Detection
  - Not detected
  - VirusTotal : 0/43

- Authentication: password
WebShell survey: Zehir4

- System information
- File Manager
- File upload/download.
- SQL Manager

- Comments: in Turkish

- Antivirus Detection
  - Detected
  - VirusTotal: 34/43

- Authentication: nothing
WebShell survey: ASP Spider

- System information
- File Manager
- File upload/download.
- SQL Manager

- Antivirus Detection
  - Not detected

- Authentification: password
16 WebShells analyzed in details:
- 11 PHP, 2 JSP, 3 ASP

Missing features:
- Some feature are not working well on all server configurations (often database Manager)

Created for the malicious purposes => are not adapted for pentester needs

Detected by the antivirus

Insufficient protection: password

Work only on Linux or Windows
**Summary**

### Problematic & Objective

### State of Art
- Environment study
- WebShell survey
- **Obfuscation and protection tools**

### Conception

### Proof-of-concept
- Pieces of code
- Demonstration

### Conclusion & perspectives
PHP protection and obfuscation

- Two categories
  - With additional modules installation
  - Without additional modules installation

- PHP code protection technics
  - Encoding
    - Bytecode
    - Base64
  - Obfuscation
    - Classes and variables names are transformed in incomprehensible strings (by applying md5, sha1, III11, …)
    - Clearning spaces, newlines, comments
  - Encryption

- Tools:
  - PHP Obfuscator – Raizlabs
  - Obfusc PHP
  - Code Eclipse
  - Source Guardian
  - IonCube PHP Encoder
  - FOPO Free Online PHP Obfuscator
  - Codelock 2.7
  - Zend Guard
  - Nu-Coder
  - Byterun
  - SourceCop for PHP
  - phpCipher
  - PHP LockIt!

...
Three levels of encoding
- Null
- Basic
- Freelancer

Possible to choose manually the entities to obfuscate

Example of PHP obfuscation tool: Obfusc PHP 3.0.0
Example of PHP obfuscation tool: Codelock 2.7

www.codelock.co.nz

- PHP, HTML and JavaScript encryption
- Possible to rely encryption key to defined IP address and define expiration date
- Encryption key is located inside generated code but it’s encoded in base64!
ASP protection and obfuscation: Stunnix VBS-Obfus

- **Obfuscation modes:**
  - Code Mangling
    - Replacement of variable names and functions by incomprehensible strings by applying md5, III..1, random permutation of characters, the shortest possible names
  - Cleaning comments
  - Hiding constants
  - HTML Mangling
    - Cleaning comments, spaces in html text
    - Transforming string characters in entities starting by &#
    - Transforming tag names in different formats (Uppercase/Lowercase)
  - Encoding (asp brut and asp inserted in html with script)

- **Others Tools:**
  - ASP Expert Obfuscator
  - Stunnix VBS-Obfus
  - Spices.Obfuscator.Net 5.1
  - DeepSea Obfuscator
  - Eziriz .NET Reactor
ASP protection and obfuscation

Stunnix VBS-Obfus

```vbscript
Function BufferContent(data)
    Dim strContent(64)
    Dim i
    ClearString strContent
    For i = 1 To LenB(data)
        AddString strContent, Chr(AscB(MidB(data, i, 1)))
    Next
    BufferContent = fnReadString(strContent)
End Function
```

```vbscript
<% Function ReplacementFor_buffercontent(ReplacementFor_data) Dim ReplacementFor_strcontent(64) Dim ReplacementFor_1 ReplacementFor_cleanstring ReplacementFor_strcontent
For ReplacementFor_i = 4h120d+401 To LenB(ReplacementFor_data) ReplacementFor_addstring ReplacementFor_strcontent, Chr(AscB(MidB(ReplacementFor_data, ReplacementFor_1, (s h147+2434-6Mide8)))) Next ReplacementFor_buffercontent=ReplacementFor_cleanstring(ReplacementFor_strcontent) %>
```
Java protection and obfuscation

- Data and variable obfuscation:
  - Changing variable storage (transformation from local type to global)
  - Variable encoding
  - Data aggregation (array of two dimensions to one dimension array)

- Function and control obfuscation:
  - Function aggregation
  - Changing operation order (loop inversion etc)
  - Changing function names (on the most shortest possible names, applying md5 etc)
  - Code complexified elements:
    - Add never executing functions
    - Introduction of goto in bytecode
    - Add redundant conditions for loops

Tools:
- Zelix KlassMaster
- Cinnabar Canner
- Jmangle The Java Class Mangler
- RetroGuard
- JODE

...
Java protection and obfuscation

```java
/*
 * public class Shell extends HttpServlet {
 *     protected void processRequest(HttpServletRequest request, HttpServletResponse response)
 *         throws ServletException, IOException {
 *             response.setContentType("text/html;charset=UTF-8");
 *             PrintWriter out = response.getWriter();
 *             String commande_result = "";
 *             try {
 *                 printPageStart(out);
 *                 Enumeration en = request.getParameterNames();
 *                 while (en.hasMoreElements()) {
 *                     String paramName = (String) en.nextElement();
 *                     if (paramName.equals("cmd"))
 *                         commande_result = executeCmd(request.getParameter(paramName));
 *                 }
 *                 out.println(commande_result);
 *                 printPageEnd(out);
 *             } finally {
 *                 out.close();
 *             }
 *         }
 * 
 * private void printPageStart(PrintWriter out) {
 *     out.println("<html>");
 *     out.println("<head>");
 *     out.println("<title>Servlet Cmd Servlet</title>");
 *     out.println("<head>");
 *     out.println("<body>");
 *     out.println("<FORM METHOD=GET ACTION='Shell'\n\n\n<INPUT name='cmd' type=text><INPUT type=submit value='Run'>\n\n</FORM>"):
 * 
 * }
 * 
 * private void printPageEnd(PrintWriter out) {
 *     out.println("</body>");
 *     out.println("</html>");
 *     out.close();
 * }
 * 
 * public String executeCmd(String input) {
 *     String o = null;
 *     String output = "";
 *     try {
 *         Process p = Runtime.getRuntime().exec("cmd.exe /C " + input);
 *         BufferedReader si = new BufferedReader(\n *             new InputStreamReader(p.getInputStream()));
 *         while ((s = si.readLine()) != null) {
 *             output += s;
 *         }
 *     } catch (IOException e) {
 *         e.printStackTrace();
 *     } catch (Exception e) {
 *         e.printStackTrace();
 *     }
 *     return output;
 * }
 */
```
Java protection and obfuscation

```java
public String a(String s)
{
    boolean flag = false;
    Object obj = null;
    String s2 = "*";
    try {
        Process process = Runtime.getRuntime().exec((new StringBuilder()).append("B[7]").append(s).toString());
        BufferedReader bufferedreader = new BufferedReader(new InputStreamReader(process.getInputStream()));
        String s1;
        if ((s1 = bufferedreader.readLine()) == null)
        {
```

```java
            byte byte0 = 3;
            byte0;
            JVM INSTR ixor ;
            (char);
            JVM INSTR castore ;
            int;
            JVM INSTR swap ;
            JVM INSTR dup_x1 ;
            JVM INSTR lstore 201;
            goto L9 L3
```

```java
        }
```
Protection and obfuscation tools: summary

**Protection mechanisms:**

- **Encoding**
  - Reversible
  - Hide program structure
  - Decoders exist
- **Obfuscation**
  - Irreversible
  - Doesn't permit to hide completely program structure
  - Existence of « Code Beautifiers »
- **Detection of protection and obfuscation method applied by automatic tools:**
  - Zendecode (decoding tool for: PHPCipher, Codelock, Truebug, Sourcecop, Byterun, ElearningForce, PHPLockit, and PHPion)
  - PCL's PHPiD (detect obfuscation and protection tools used)
- **Encrypting:** best solution for PHP and ASP
- **Obfuscation on JAVA class/code gives good results**
Summary

Problematic & Objective

State of Art
• Environment study
• WebShell survey
• Obfuscation and protection tools

Conception

Proof-of-concept
• Pieces of code
• Demonstration

Conclusion & perspectives
WebShell conception: functional model
WebShell conception: UML model « General use case »

- **Architecture Client-Agent**
  - **Client**:
    - Creation of Agent script
    - Command execution initiated by pentester
    - Management of modules
    - Management of deployed Agent scripts
  - **Agent**:
    - Command execution initiated by Client
  - **Characteristics**:
    - Client-Agent communication is encrypted
    - Agent code source is encrypted
Conception details

- Encryption AES128 (communication, source code) based on tree parameters:
  - A password chosen by a pentester
  - Target server IP address
  - Pentester machine IP address
**WebShell conception: our answer and PoC**

- **Homogenization and centralization of PHP, JSP and ASP WebShells**
  - Elaboration of the unique framework capable to generate agent PHP, ASP and JSP scripts

- **Protection against a third non authorized person's use**
  - Encryption of executable code
  - Client-Agent architecture
  - Integrality verification for the agent (no implemented yet)
  - Key encryption locking on the pentester machine IP address

- **Bypass IDS/IPS and WAF**
  - Communication encryption between target server and pentester device
  - WebShell source code encryption
Protection against WebShell steal and its reutilization for the malicious purposes
- WebShell code source encryption based on the password chosen by pentester
- Unique password for each deployed agent

Future WebShell evolution with possibility to add functionalities by pentesters
- Modular structure
- Module management and creation feature

Traceability for deployed WebShells
- Project structure
- Projects management, management of deployed agents
- Future evolution: centralization of information on the central server, agents signed with private pentester client key
WebShell functionalities

« Need to have » functionalities

- System information
- Grafical file manager
- File upload/download
- Command line cmd
- SQL manager

« Nice to have » functionalities (by their priority)

- Network discovery (ICMP/Traceroute) & Port scan (SYN)
- Bind/reverse connection
- Text search in files
- Actif process control
- File and folders archivator
- FTP client
- MySQL dump
- Safe Mode bypass
- Converter base64, hex, hashage md5, sha1
- Brute)force password breaking
- Mail sending
Summary

Problematic & Objective

State of Art
- Environment study
- WebShell survey
- Obfuscation and protection tools

Conception

Proof-of-concept
- Pieces of code
- Demonstration

Conclusion & perspectives
Client WebShell: Java Swing, Netbeans IDE
WebShell proof-of-concept

Interface / Demo
Summary

Problematic & Objective

State of Art
- Environment study
- WebShell survey
- Obfuscation and protection tools

Conception

Proof-of-concept
- Pieces of code
- Demonstration

Conclusion & perspectives
Conclusion and perspectives

- **WebShell framework was developed to meet:**
  - Homogenization: versions PHP, ASP and Java available with a single interface
  - Modular structure: features adopted to auditors and possibility to add their own modules
  - Protection: encrypted source code and encrypted communications

- **Outlook**
  - Finalization of the development of modules like TCP tunnel (Reduh style)
  - Tests on different platforms / infrastructures

- **Distribution**
  - No distribution of this PoC is planned
Questions / Answers
The presented study is in order to carried out Ethical Hacking

Some tools presented in this slide maybe Unlawful in some country

Locale legislation must be apply