Avaliação de Segurança em Softwares com desenvolvimento próprio

Fabio Xavier
Ricardo Abade
TCESP

Sumário

- Objetivos
- Metodologia
- Ferramentas utilizadas
  - Aplicações Web
  - Aplicações “executáveis”
- Resultados e conclusões

Objetivos

Objetivo

Minimizar a utilização de softwares com as mais comuns vulnerabilidades de código, como forma de minimizar o risco de ataques

No TCE, no ano de 2013, foram registradas mais **de 1.800 tentativas de ataques por dia**

Foram bloqueadas nos sistemas de segurança 325 endereços e redes IP's

Principais ataques exploravam:

- Buffer Overflow
- Vulnerabili-dades em navegadores
- Vulnerabili-dades em sistemas Web

Metodologia

Metodologia Implantada

Foi criada uma instrução de trabalho divulgada para todas as áreas de desenvolvimento

Itens da instrução de trabalho:

Definição de que todo software (novo sistema ou versão) deve passar pela avaliação da Seção de Segurança da Informação

- Após a homologação do usuário e antes de entrar em produção

Definição de itens que devem ser informados à Seção de SI:

- URL completa ou localização do executável
- Login e senha para teste
- Instruções básicas para navegação

Ferramentas Utilizadas

Ferramentas Utilizadas

Para avaliação de aplicações Web

- OWASP SAP Proxy (http://www.owasp.org)
- W3AF (http://w3af.org/)
- Distribuições Linux que têm essas ferramentas:
  - KALI Linux (https://www.kali.org/)
  - BlackArchLinux (http://blackarch.org/tools.html)
  - Pentoo (http://www.pentoo.ch/)

Para avaliação de executáveis

- Cuckoo Sandbox (http://www.cuckoosandbox.org)

Avaliação de Aplicações Web

Importância:

- Maioria das vulnerabilidades do CVE (Common Vulnerabilities and Exposures)
  - *Cross-site Scripting (XSS)*
  - *SQL Injection*

Características:

- Independência tecnológica: ASP, PHP, JSP, etc
- Facilidade de uso
- Alto nível de Automação

Avaliação de Aplicações Web

ESTRUTURA GERAL

- **Descoberta**
  - Identificação de URLs e pontos de entrada da aplicação (formulários, variáveis URL, campos hidden, cookies e cabeçalhos http)
- **Ataque**
  - Tentativas de ataques aos pontos de entrada
- **Análise**
  - Detecção de vulnerabilidades pelas respostas às tentativas de ataques

OWASP

Open Web Application Security Project

OWASP

- Open Web Application Security Project
  - Entidade sem fins lucrativos com objetivo de aumentar a segurança de software

"The Top 10 Most Critical Web Application Security Risks"
(https://www.owasp.org/index.php/Top_10_2013-Top_10)

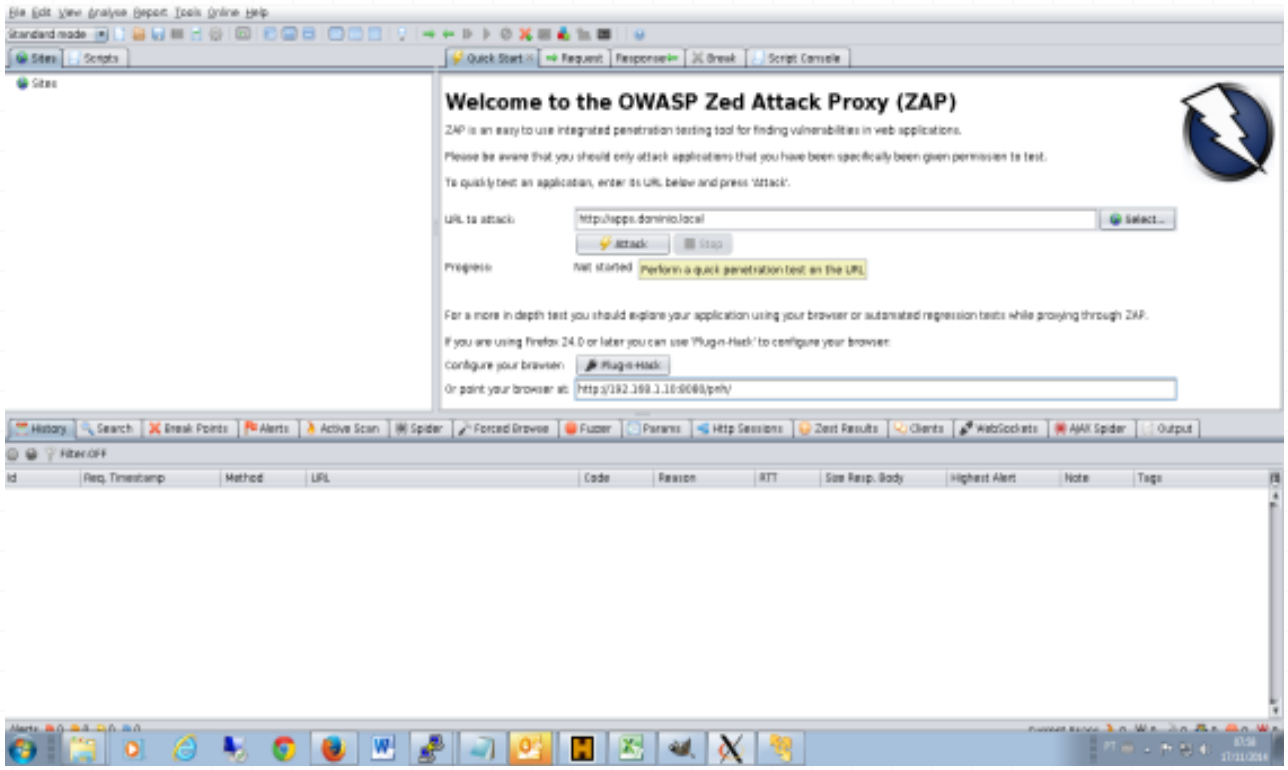
- Injeção de Código
- Quebra de autenticação e Gerenciamento de Sessão
- Cross-site scripting
- Referência insegura e direta a objetos
- Configuração incorreta de segurança
- Exposição de dados sensíveis
- Falta de função para controle do nível de acesso
- Cross-site Request Forgery (CSRF)
- Utilização de Componentes vulneráveis conhecidos
- Redirecionamentos e encaminhamentos inválidos

OWASP-ZAP

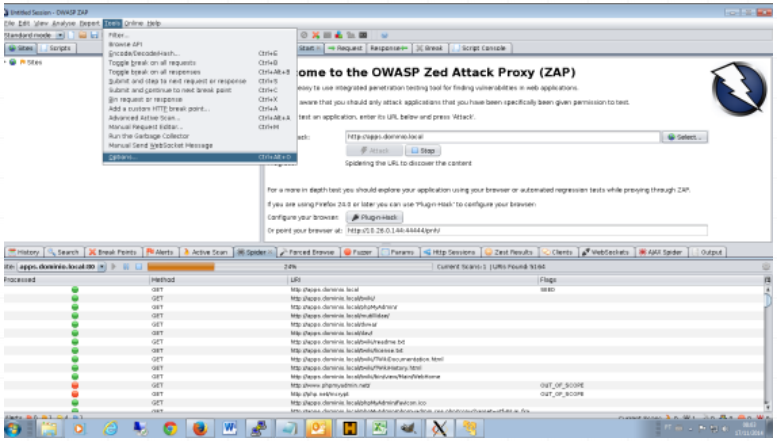
Zed Attack Proxy (http://code.google.com/p/zaproxy/wiki/Downloads?tm=2)

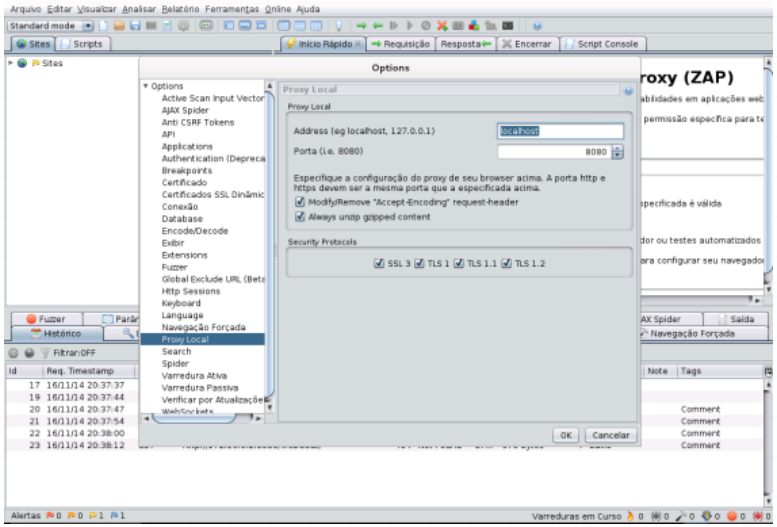
- Opensource multiplataforma, com versão em português
- Iniciativa apoiada por grandes empresas (Microsoft, Google, Mozilla, Ernst & Young, dentre outras)
- Possui scanners automatizados
- Conjunto de ferramentas para encontrar vulnerabilidades manualmente
- perfis pré-definidos que trazem alguns plug-ins selecionados de acordo com a finalidade do perfil
  - OWASP_TOP10
  - audit_high_risk
  - bruteforce
  - fast_scan
  - full_audit

OWASP-ZAP
Tela Inicial

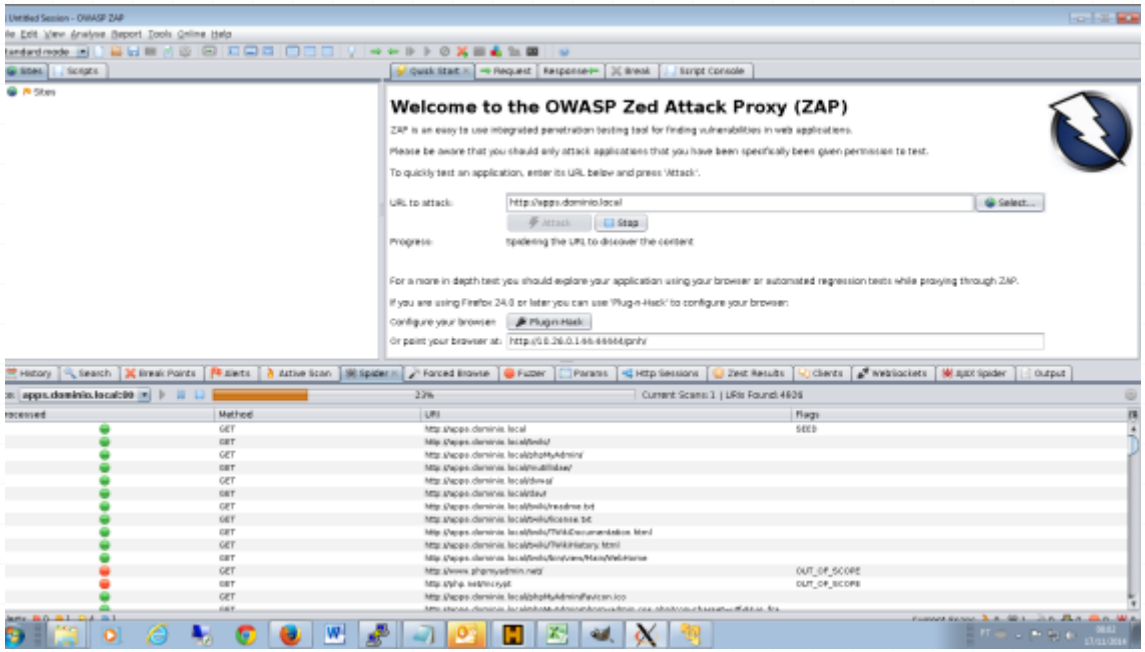


OWASP-ZAP
Configuração de Proxy

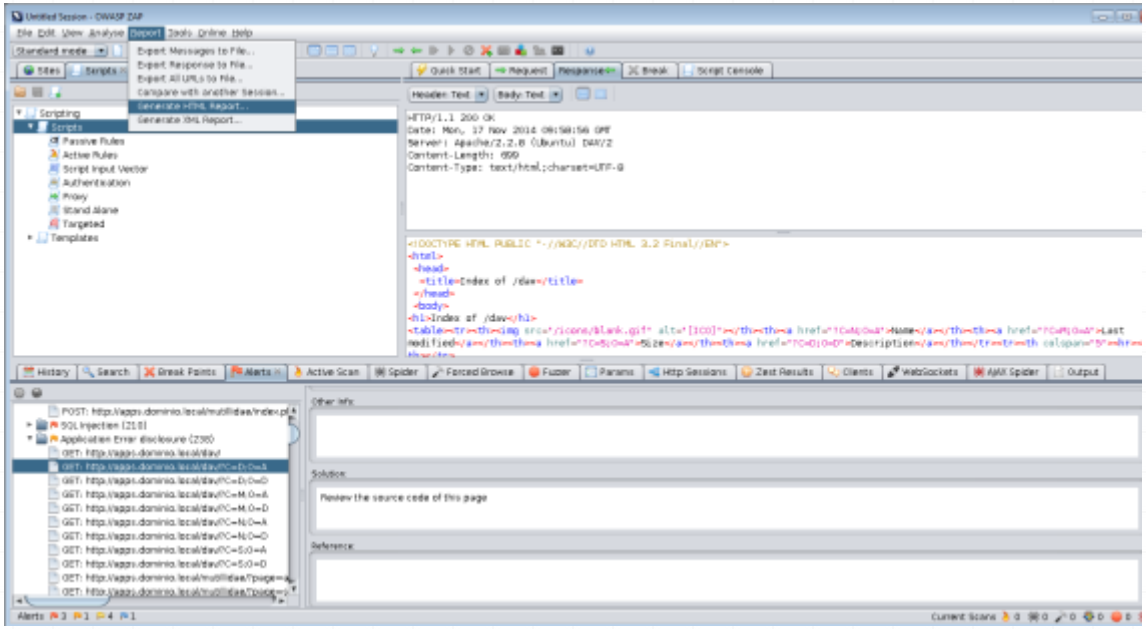




OWASP-ZAP
Execução de Scan



OWASP-ZAP
Geração de Relatório



OWASP-ZAP
Geração de Relatório

ZAP Scanning Report

Summary of Alerts

Risk Level	Number of Alerts
High	1220
Medium	235
Low	4580
Informational	4181

Alert Detail

High (Warning)	Path Traversal
Description	The Path Traversal attack technique allows an attacker access to files, directories, and commands that potentially reside outside the web document root directory. An attacker may manipulate a URL in such a way that the web site will execute or reveal the contents of arbitrary files anywhere on the web server. Any device that exposes an HTTP-based interface is potentially vulnerable to Path Traversal. Most web sites restrict user access to a specific portion of the file-system, typically called the "web document root" or "CGI root" directory. These directories contain the files intended for user access and the executable necessary to drive web application functionality. To access files or execute commands anywhere on the file-system, Path Traversal attacks will utilize the ability of special-characters sequences. The most basic Path Traversal attack uses the "../" special-character sequence to alter the resource location requested in the URL. Although most popular web servers will prevent this technique from escaping the web document root, alternate encodings of the "../" sequence may help bypass the security filters. These method validations include valid and invalid Unicode-encoding ("..%u2215" or "..%c0%af") of the forward slash character, backslash characters ("..\") on Windows-based servers, URL encoded characters "%2e%2e%2f"), and double URL encoding ("..%255c") of the backslash character. Even if the web server properly restricts Path Traversal attempts in the URL path, a web application itself may still be vulnerable due to improper handling of user-supplied input. This is a common problem of web applications that use template mechanisms or load static text from files. In variations of the attack, the original URL parameter value is substituted with the file name of one of the web application's dynamic scripts. Consequently, the results can reveal source code because the file is interpreted as text instead of an executable script. These techniques often employ additional special characters such as the dot (".") to reveal the listing of the current working directory, or "%00" NULL characters in order to bypass subsequent file extension checks.
URL	http://apps.domino.local/mutillidae/index.php?page=%2Fetc%2Fpasswd
Parameter	page
Attack	root:x:0:0
Solution	Assume all input is malicious. Use an "accept known good" input validation strategy, i.e., use a whitelist of acceptable inputs that strictly conform to specifications. Reject any input that does not strictly conform to specifications, or transform it into something that does. Do not rely exclusively on looking for malicious or malformed inputs (i.e., do not rely on a blacklist). However, blacklists can be useful for detecting potential attacks or determining which inputs are so malformed that they should be rejected outright. When performing input validation, consider all potentially relevant properties, including length, type of input, the full range of acceptable values, missing or extra inputs, syntax, consistency across related fields, and conformance to business rules. As an example of business rule logic, "boat" may be syntactically valid because it only contains alphanumeric characters, but it is not valid if you are expecting colors such as "red" or "blue."

OWASP-ZAP
Geração de Relatório

High (Warning)	SQL injection
Description	SQL injection may be possible.
URL	http://apps.domino.local/tekstein/view/Main/TokyoOffice%20AND%201=1%20--?topic=ZAP
Parameter	TokyoOffice
Attack	TokyoOffice AND 1=1 --
Other information	The page results were successfully manipulated using the boolean conditions [TokyoOffice AND 1=1 --] and [TokyoOffice AND 1=2 --] The parameter value being modified was stripped from the HTML output for the purposes of the comparison Data was returned for the original parameter. The vulnerability was detected by successfully restricting the data originally returned, by manipulating the parameter
Solution	Do not trust client side input, even if there is client side validation in place. In general, type check all data on the server side. If the application uses JDBC, use PreparedStatement or CallableStatement, with parameters passed by '?' If the application uses ASP, use ADO Command Objects with strong type checking and parameterized queries. If database Stored Procedures can be used, use them. Do *not* concatenate strings into queries in the stored procedure, or use 'exec', 'exec immediate', or equivalent functionality! Do not create dynamic SQL queries using simple string concatenation. Escape all data received from the client. Apply a 'whitelist' of allowed characters, or a 'blacklist' of disallowed characters in user input. Apply the privilege of least privilege by using the least privileged database user possible. In particular, avoid using the 'sa' or 'db-owner' database users. This does not eliminate SQL injection, but minimizes its impact. Grant the minimum database access that is necessary for the application.
Reference	https://www.owasp.org/index.php/Top_10_2010-A1 https://www.owasp.org/index.php/SQL_Injection_Prevention_Cheat_Sheet
CWE Id	89
WASC Id	19

W3AF

Web Application Attack and Audit Framework

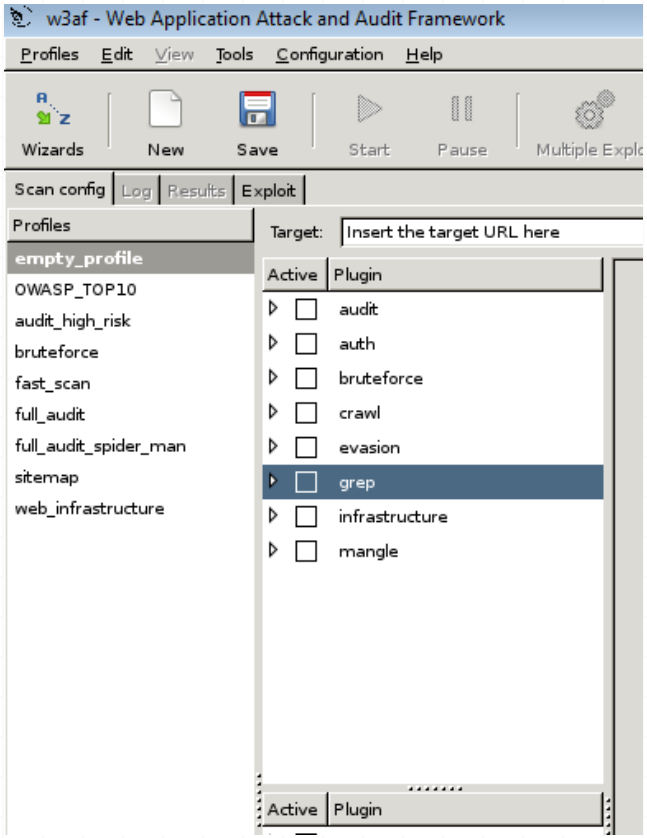
http://w3af.org/

W3AF

Web Application Attack and Audit Framework (http://w3af.org/)

- Desenvolvido em Python
- Interface gráfica e linha de comando
- Multiplataforma (Windows, Linux, Mac, FreeBSD e OpenBSD)
- Arquitetura
  - Core
  - Interface de usuário
  - Plugins

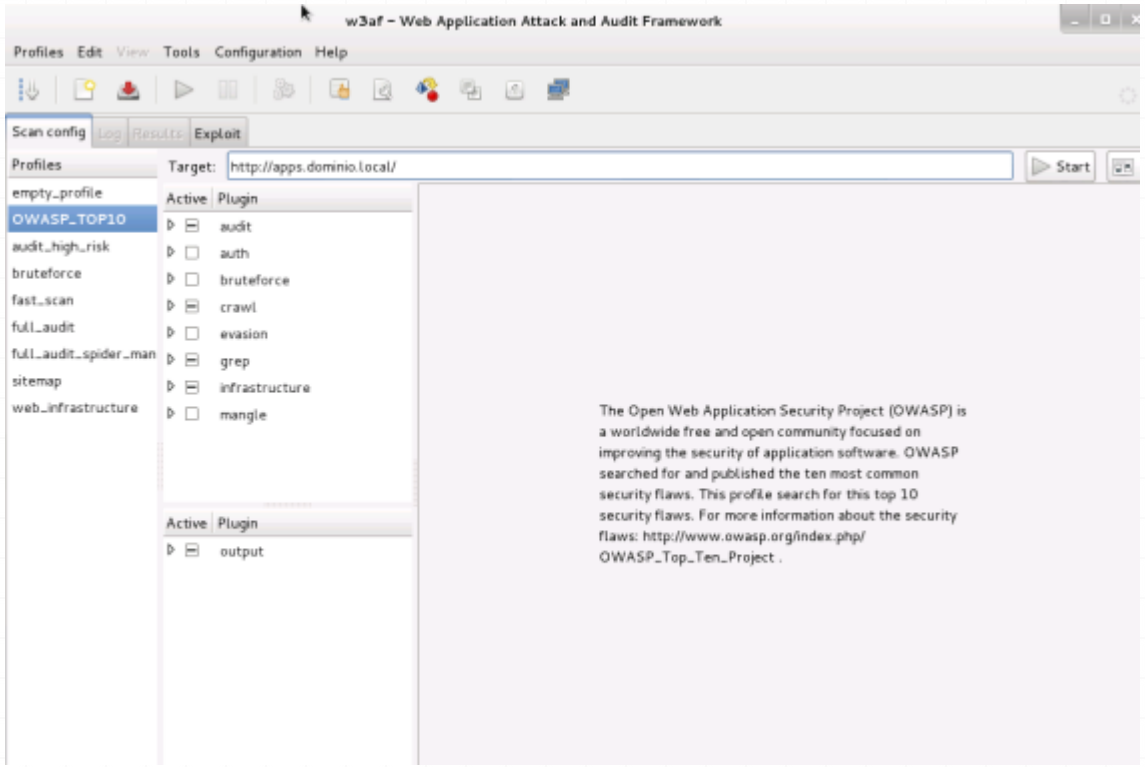
W3AF - Plugins



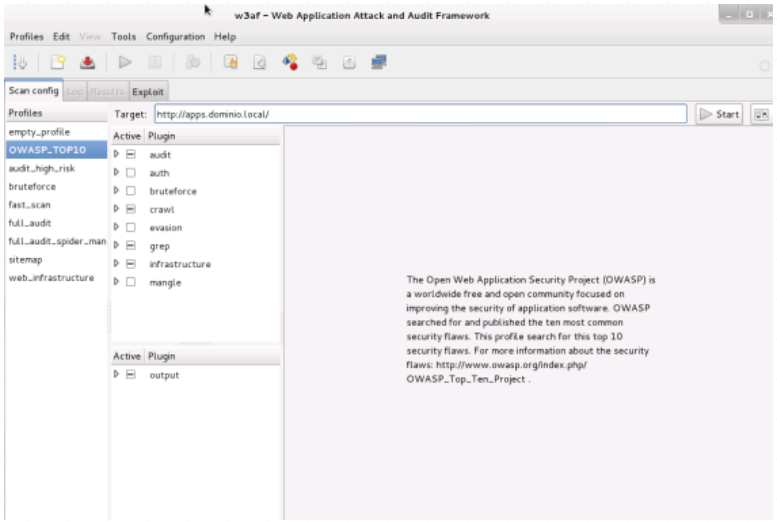
- Auditoria (audit)
- Autenticação (auth)
- Força bruta (bruteforce)
- Descoberta (crwal)
- Evasão (evasion)
- Grep
- Infraestrutura
- Mangle
- Saída (output)

W3AF – Interface Gráfica

- Executável w3af_gui



W3AF – Interface Gráfica



Perfil	Descrição
empty_profile	É um perfil vazio, sem nenhum plug-in selecionado. Pode ser usado como ponto de partida para a definição de um novo perfil.
OWASP_TOP10	Estão habilitados os plug-ins que detectam as vulnerabilidades contidas no relatório anual TOP10 da OWASP
audit_high_risk	Estão habilitados os plug-ins que detectam vulnerabilidades de maior risco, tais como SQL Injection, upload inseguro de arquivos etc.
bruteforce	Estão habilitados os plug-ins que executam ataques de força bruta, usando credenciais padrões
fast_scan	Estão habilitados alguns plug-ins de descoberta e os plug-ins de auditoria mais rápidos
full_audit	Permite executar uma auditoria completa. A descoberta de novas URLs é feita pelo plug-in web_spider
full_audit_spider_man	Permite executar uma auditoria completa. A descoberta de novas URLs é feita pelo plug-in spider_man(proxy)
sitemap	Estão habilitados apenas os plug-ins que permitem a criação de um mapa do sistema alvo
web_infraestructure	Estão habilitados os plug-ins relacionados à descoberta de informações da infraestrutura do sistema alvo. Exemplos: descoberta de virtual hosts, versão do sistema operacional, detecção de proxy reverso etc.

Executáveis

Cuckoo Sandbox

http://www.**cuckoosandbox**.org

Executáveis

Cuckoo Sandbox (http://www.**cuckoosandbox**.org)

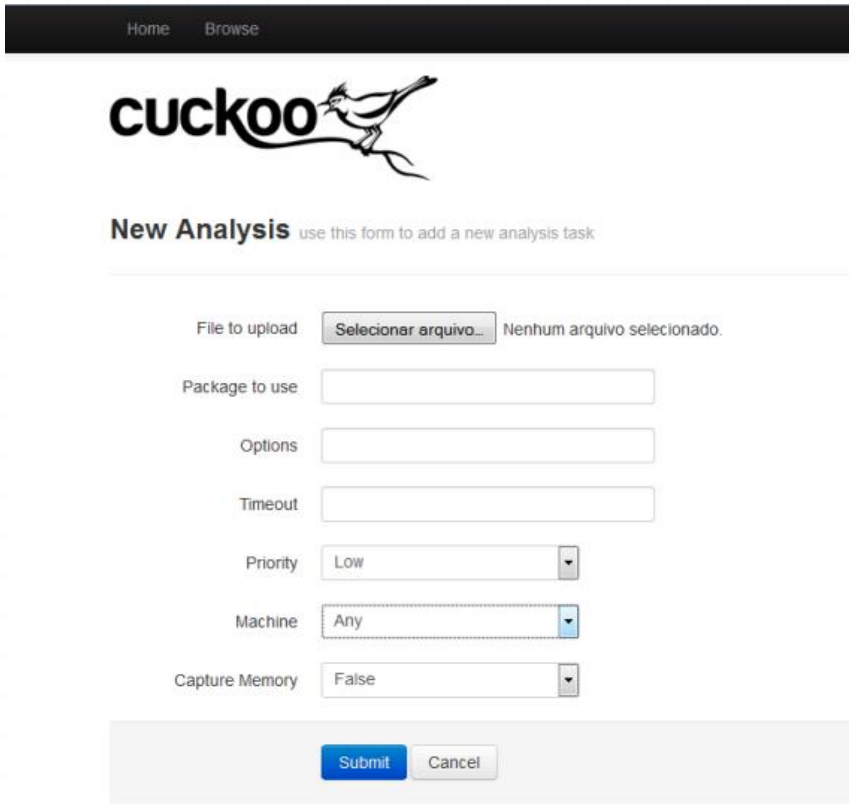
- Análise de malware
  - Verificação de chamadas API e processos
  - Registro de tráfego de rede
  - Verificação de arquivos criados, modificados ou excluídos
  - Gravação de imagens das telas
  - Arquivos dll, pdf, msoffice, scripts, jar, applet, zip
  - Geração de relatórios (csv, html, xml, json, mongodb)

Executáveis

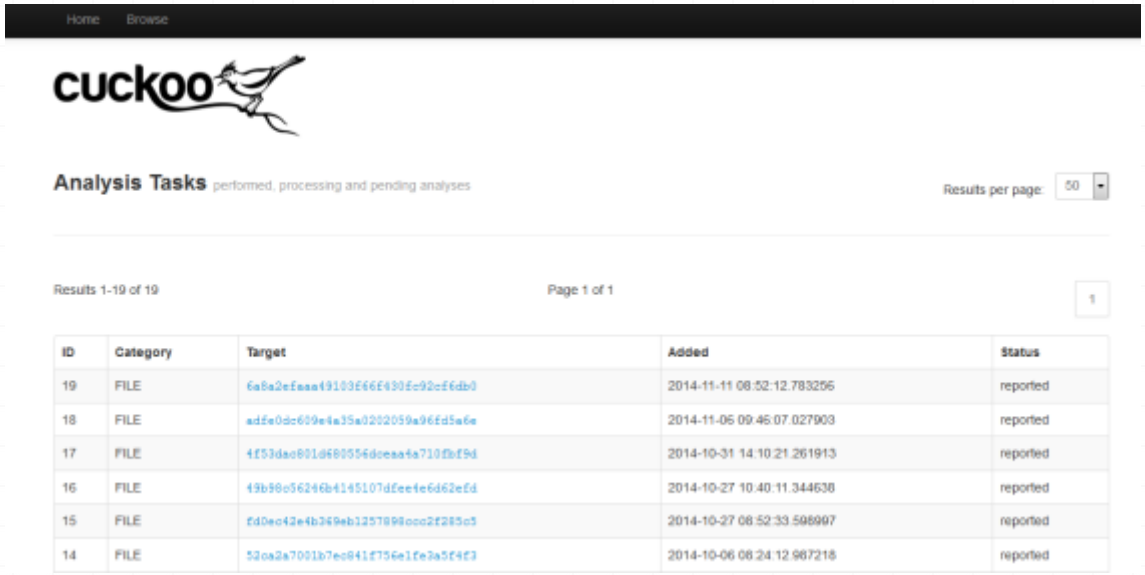
Cuckoo Sandbox (http://www.**cuckoosandbox**.org)

- Sandbox: ambiente de execução protegido
- Software de gerenciamento central (máquina host) e máquinas virtuais guests (um para análise de cada arquivo enviado, criado pelo host)
- Automatização do processo de homologação
  - Envio do executável
  - Execução e análise
  - Emissão de relatório

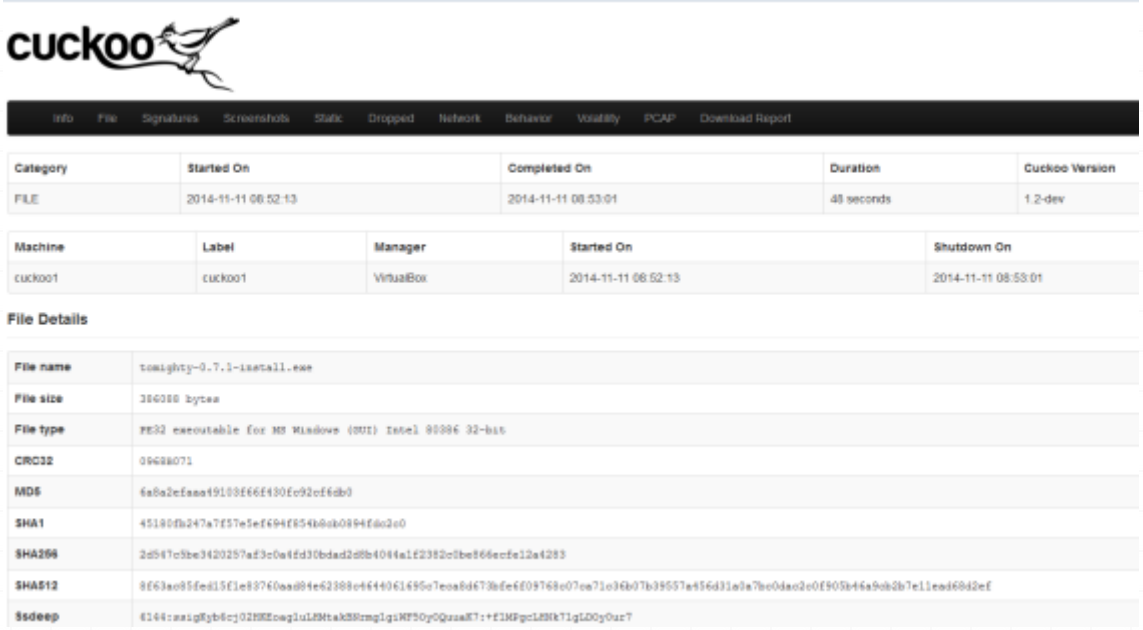
Cuckoo Sandbox
Envio de arquivo



Cuckoo Sandbox Relatórios



Cuckoo Sandbox Relatórios



cuckoo

Info | File | Signatures | Screenshots | Static | Dropped | Network | Behavior | Volatility | PCAP | Download Report

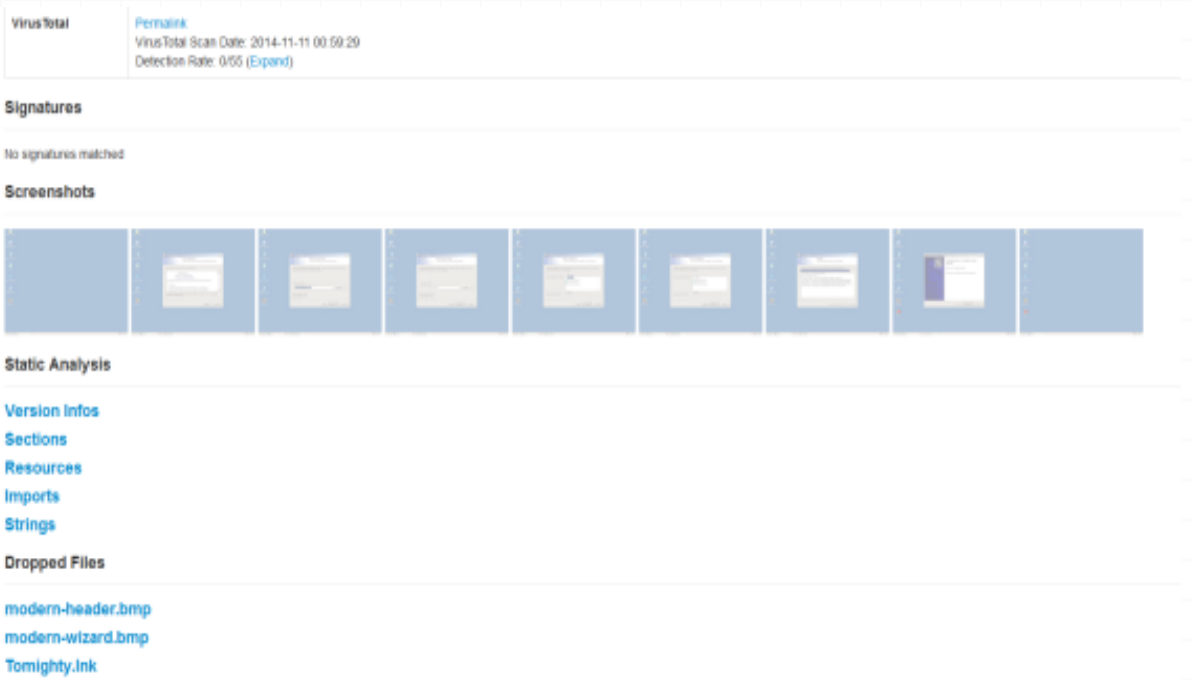
Category	Started On	Completed On	Duration	Cuckoo Version
FILE	2014-11-11 08:52:13	2014-11-11 08:53:01	48 seconds	1.2-dev

Machine	Label	Manager	Started On	Shutdown On
cuckoo1	cuckoo1	VirtualBox	2014-11-11 08:52:13	2014-11-11 08:53:01

File Details

File name	tomighty-0.7.1-install.exe
File size	386388 bytes
File type	PE32 executable for MS Windows (GUI) Intel 80386 32-bit
CRC32	096B8071
MD5	6a8a2efaaa49103f66f430fc92cf6db0
SHA1	45180fb247a7f57e5ef694f854b0ab0894fd62c0
SHA256	2d567c5be3420257af3c0a4fd30bdad2d8b4044a1f2382c8be866ecfe12a4283
SHA512	8f63ac85fed15f1e83760aad84e62388c4644061695c7eca8d673bfe6f09768c07ca71c36b07b39557a456d31a0a7bc0dac2c0f905b46a9ab2b7e11ead68d2ef
Ssdeep	6144:zsigKyb6cj02HKEoag1uL8MtakSSrmg1g1MF5OyOQzzaE7:+flMFgcL8Nk71gLDOyOz7

Cuckoo Sandbox Relatórios



VirusTotal

Permalink
VirusTotal Scan Date: 2014-11-11 00:59:29
Detection Rate: 0/55 (Expand)

Signatures

No signatures matched

Screenshots

Static Analysis

Version Infos
Sections
Resources
Imports
Strings

Dropped Files

modern-header.bmp
modern-wizard.bmp
Tomighty.lnk

Resultados

Resultados

- Redução drástica de softwares vulneráveis em produção;
- Redução de incidentes de segurança relacionados à vulnerabilidades de software;
- Definição/inclusão de nova etapa no ciclo de desenvolvimento de software interno;
- Disseminação da cultura de segurança da informação

Obrigado!

Fabio Xavier (fxavier@tce.sp.gov.br)
Ricardo Abade (abade@tce.sp.gov.br)