BinSecSweeper: Binary Security Posture Verification

OWASP Spain
Barcelona 2014

VULNEX

ME & VULNEX

Simon Roses Femerling

- Founder & CEO, VULNEX www.vulnex.com
-  @simonroses | @vulnexsl
- Former Microsoft, PwC, @Stake
- Black Hat, RSA, OWASP, SOURCE, AppSec, DeepSec, TECHNET, HITB
- CISSP, CSSLP & CEH

VULNEX

- CyberSecurity Startup
- Services & Training
- Product development: BinSecSweeper and Computer Forensics Solution

http://www.simonroses.com/es/2014/06/mi-visita-al-pentagono/

VULNEX

TALK OBJECTIVES

- Secure development
- Verification technologies
- Assess software security posture

AGENDA

1. Secure Development: Verification
2. BinSecSweeper
3. Case Studies
4. Conclusions

1. Secure Development: Verification

VULNEX

1. SOFTWARE == HOUSE

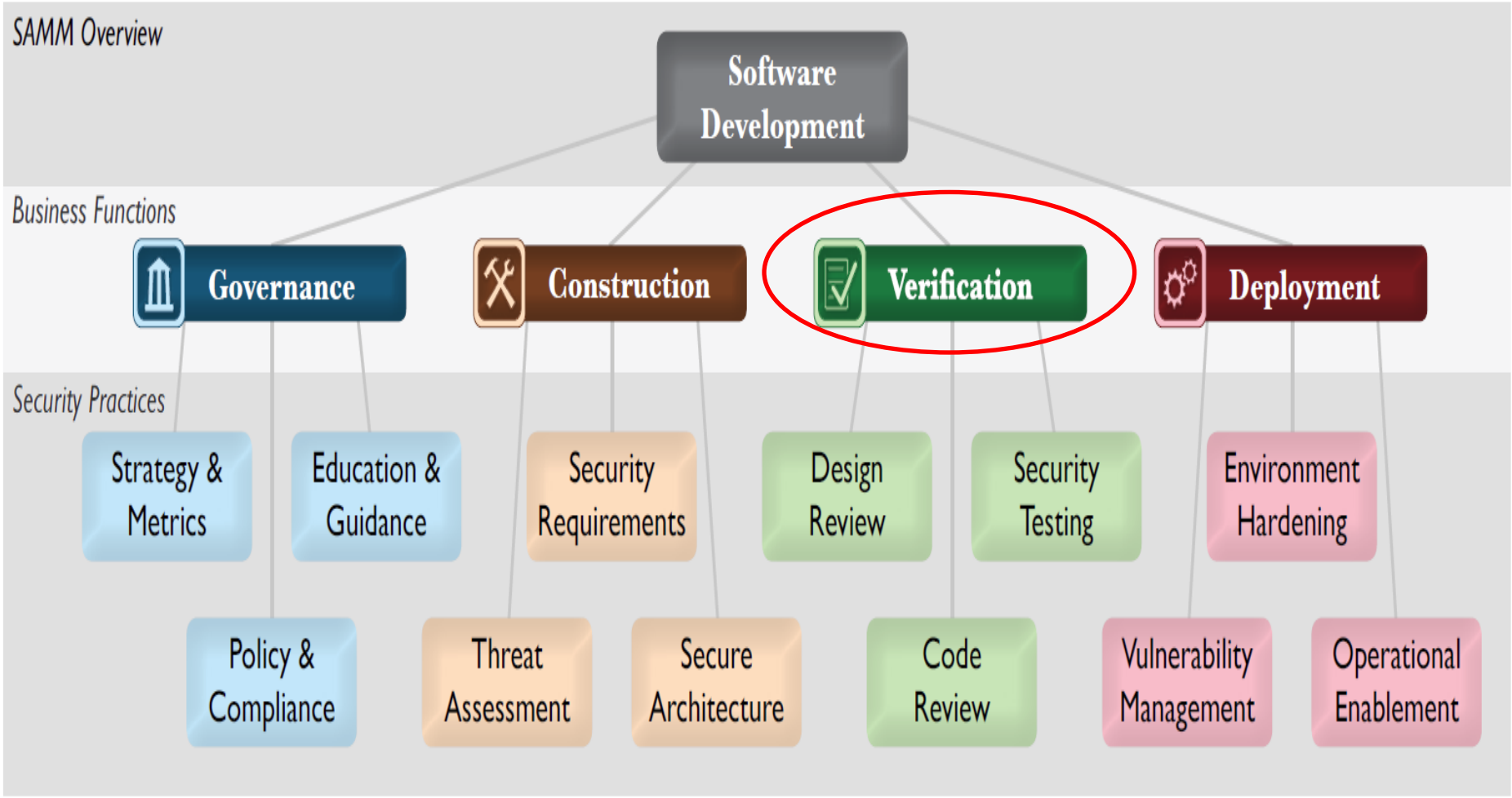
VULNEX

1. SECURE DEVELOPMENT: VERIFICATION

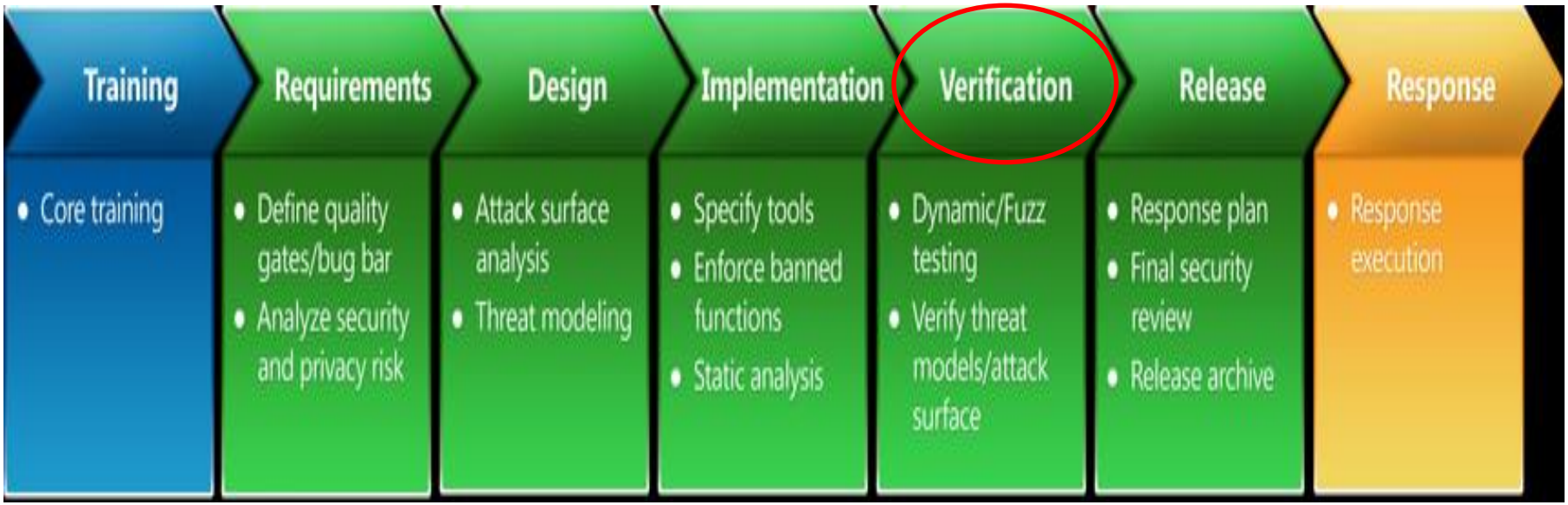
- MS SDL
  - "This phase involves a comprehensive effort to ensure that the code meets the security and privacy tenets established in the previous phases."

- Software Assurance Maturity Model (SAMM)
  - "Verification is focused on the processes and activities related to how an organization checks and tests artifacts produced throughout software development. This typically includes quality assurance work such as testing, but it can also include other review and evaluation activities."

1. OPENSAMM


SAMM Overview
Software Development
Business Functions
Governance
Construction
Verification
Deployment
Security Practices
Strategy & Metrics
Education & Guidance
Security Requirements
Design Review
Security Testing
Environment Hardening
Policy & Compliance
Threat Assessment
Secure Architecture
Code Review
Vulnerability Management
Operational Enablement

1. MICROSOFT SDL



1. IT'S ABOUT SAVING MONEY!

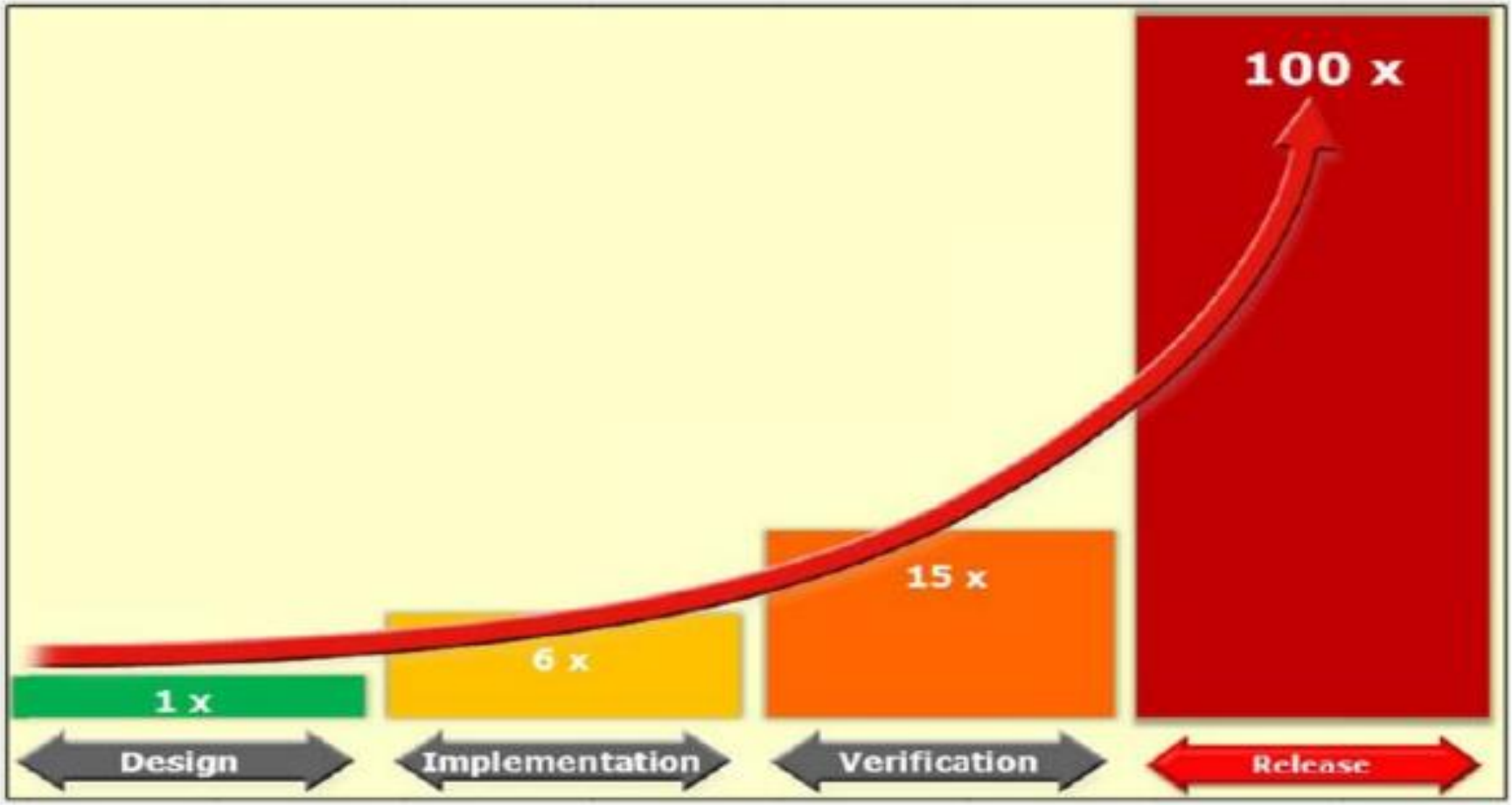


Figure 1: Cost of Bug Elimination in the Software Development Lifecycle [NIST 2002]

VULNEX

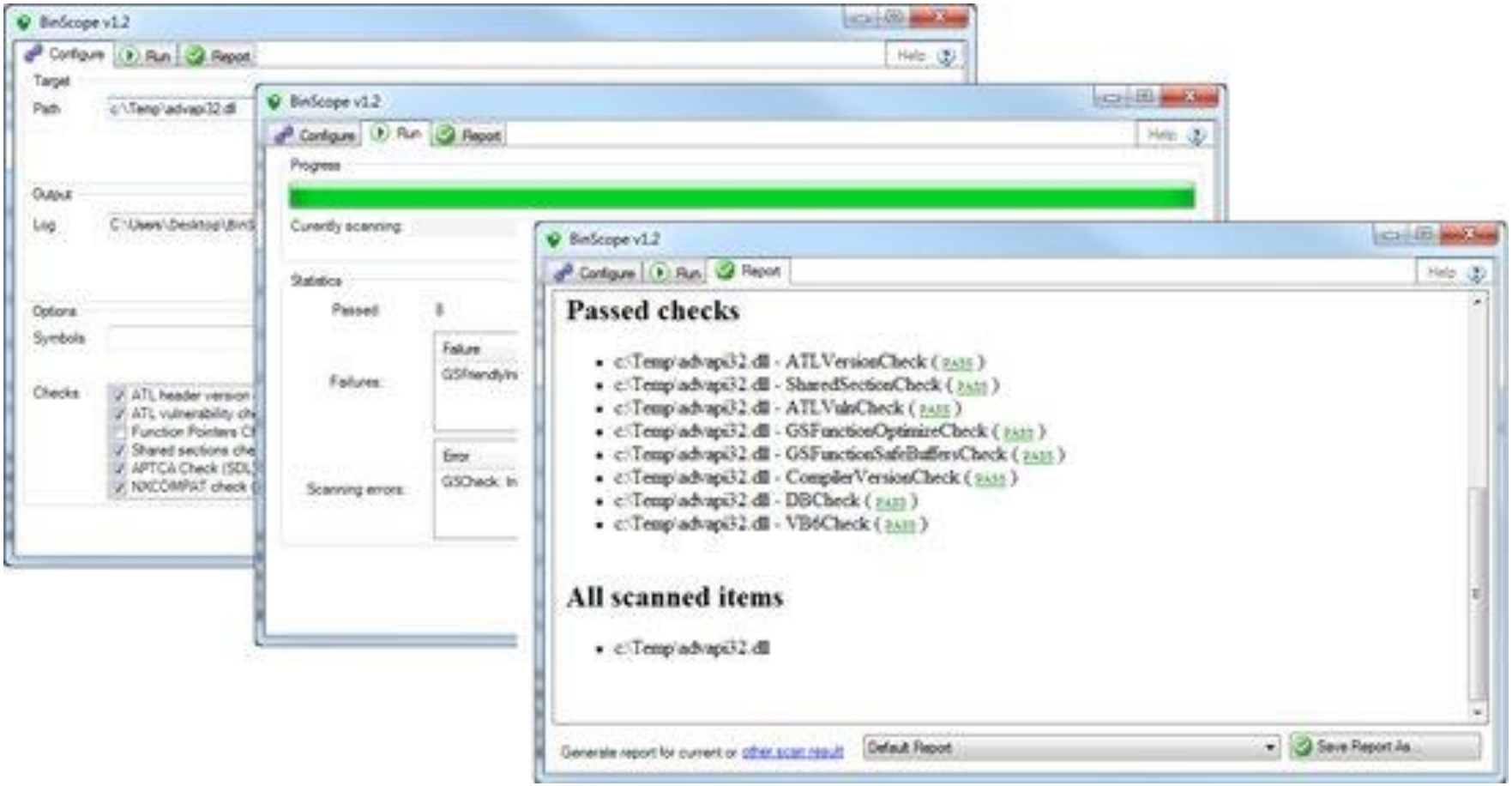
1. OTHER VERIFICATION TOOLS

- Microsoft BinScope
  http://www.microsoft.com/en-us/download/details.aspx?id=11910

- RECX Binary Assurance for Windows
  http://www.recx.co.uk/products/exeaudit.php

- ErrataSec Looking Glass
  http://blog.erratasec.com/search/label/LookingGlass#.UodWXJ2DN9A

1. BINSCOPE



VULNEX

1. CURRENT VERIFICATION TOOLS

- Platform specific
  - Windows: BinScope, Looking Glass & Binary Assurance
  - Linux: checksec.sh and custom scripts

- Limited set of checks
  - Check for defenses but what about:
    - Compiler used
    - External libs used
    - Malware
    - You name it…

- Not easy to extend

VULNEX

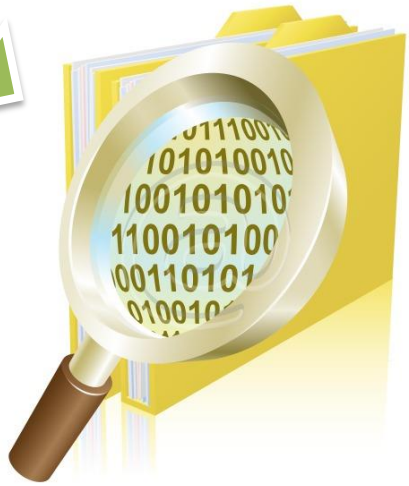
1. BINARY INTELLIGENCE

File Information

- Size
- Hash
- Timestamp
- Strings

Compiler

- Name
- Version



Security Mitigations

- DEP
- ASLR
- Stack Cookies

Vulnerabilities

- Unsafe API
- Weak Crypto
- Backdoors

VULNEX

2. BinSecSweeper

VULNEX

2. WHY BINSECSWEEPER?

- BinSecSweeper is VULNEX binary security verification tool to ensure applications have been built in compliance with Application Assurance best practices

- The goal for BinSecSweeper is a tool:
  - Developers can use to verify their output binaries are safe after compilation and before releasing their products
  - IT security pros to scan their infrastructure to identify binaries with weak security defenses or vulnerabilities.

- BinSecSweeper is a cross platform tool (works on Windows and Linux) and can scan different file formats: PE and ELF.

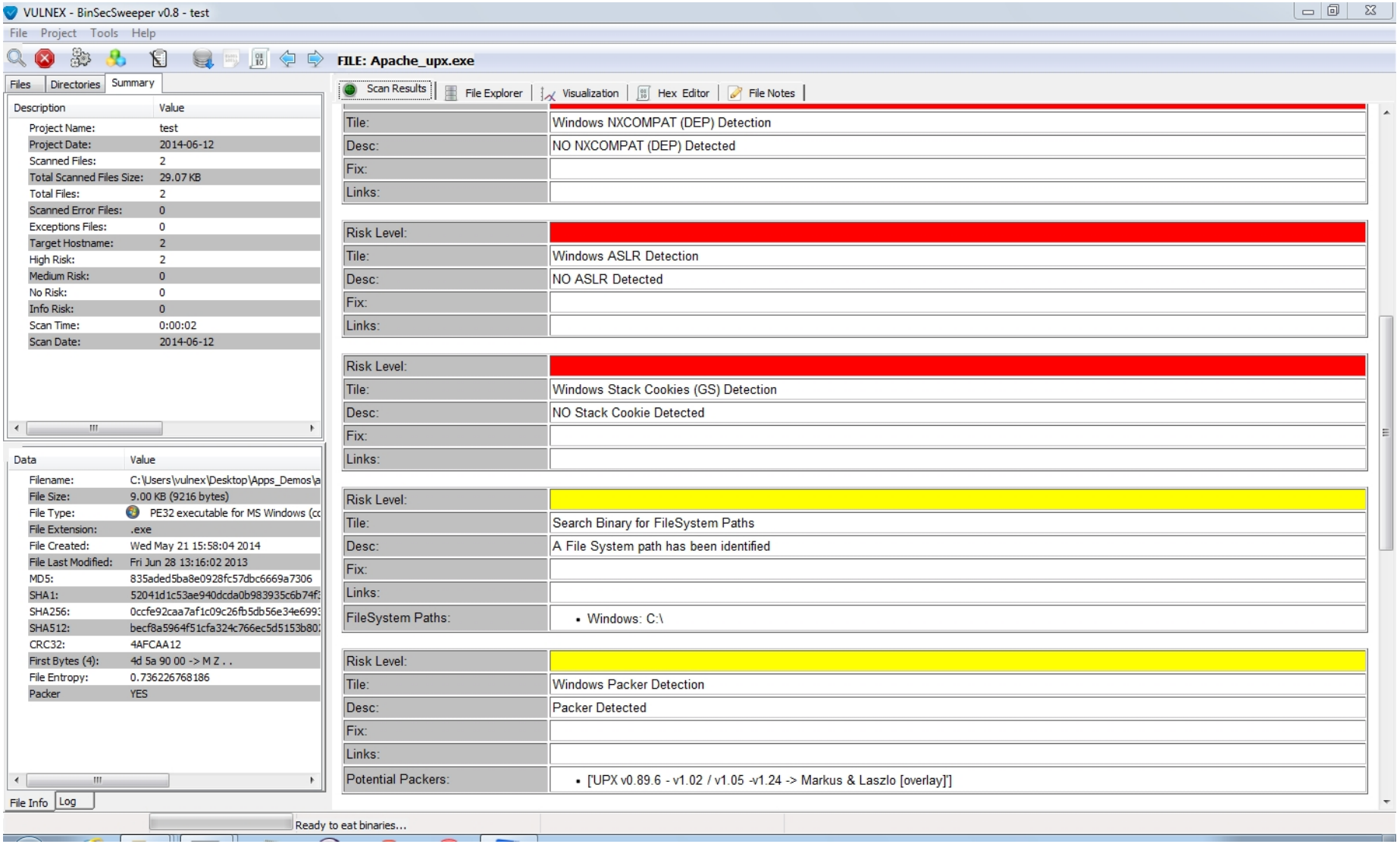
BINSECSWEEPER FEATURES

- 100% in Python
- Easy to use
- Cross-platform: works on Windows & Linux
- Scans Windows (PE) and Unix (ELF) files for security checks
- Configurable
- Analysis Engine
- Extensible by plugins
- Visualizations
- Reporting
- Perform mass binary scanning

Supported files: exe, dll, sys, msi, so, a, scr, cpl, ocx, dry

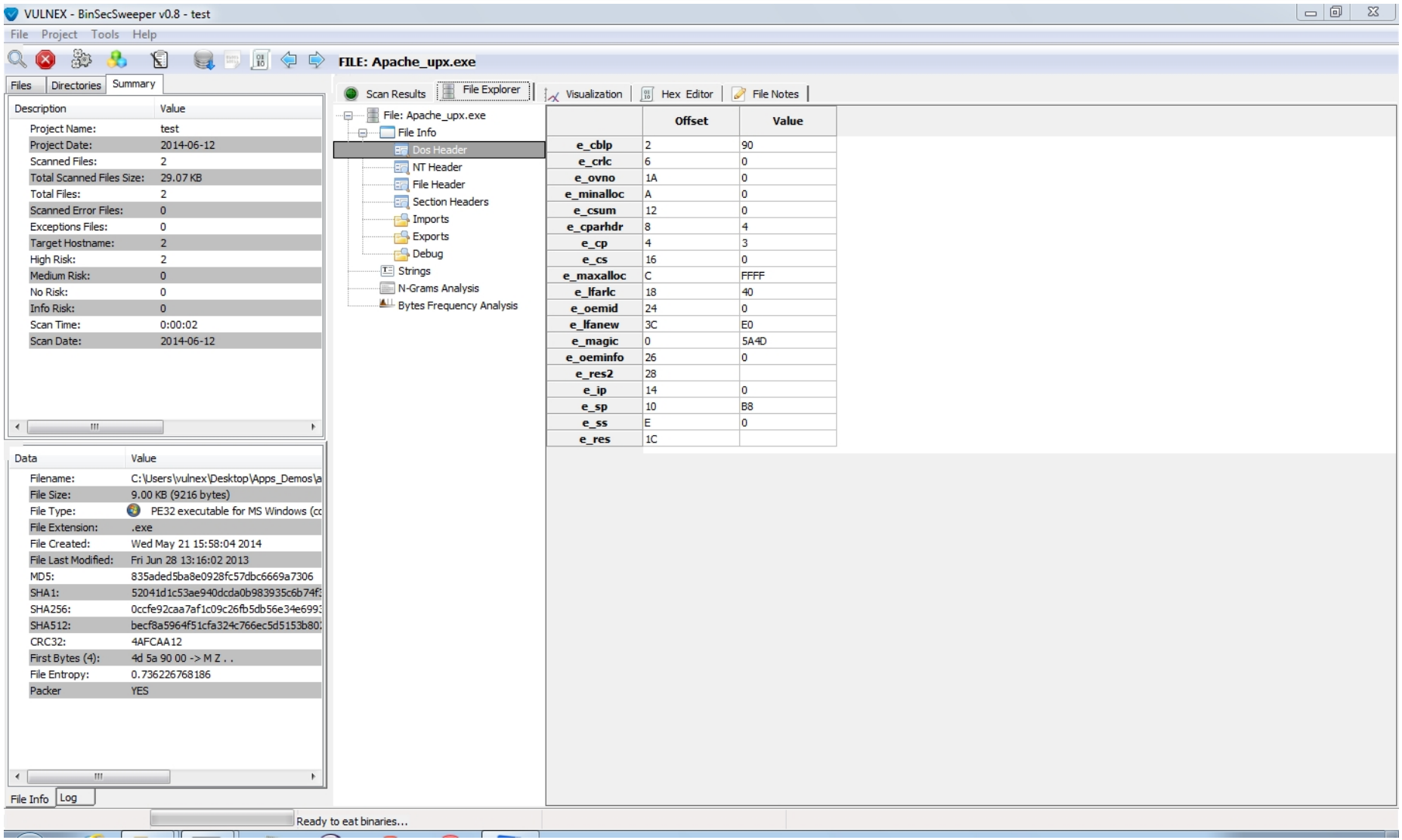
VULNEX

2. BINSECSWEEPER IN ACTION (I)



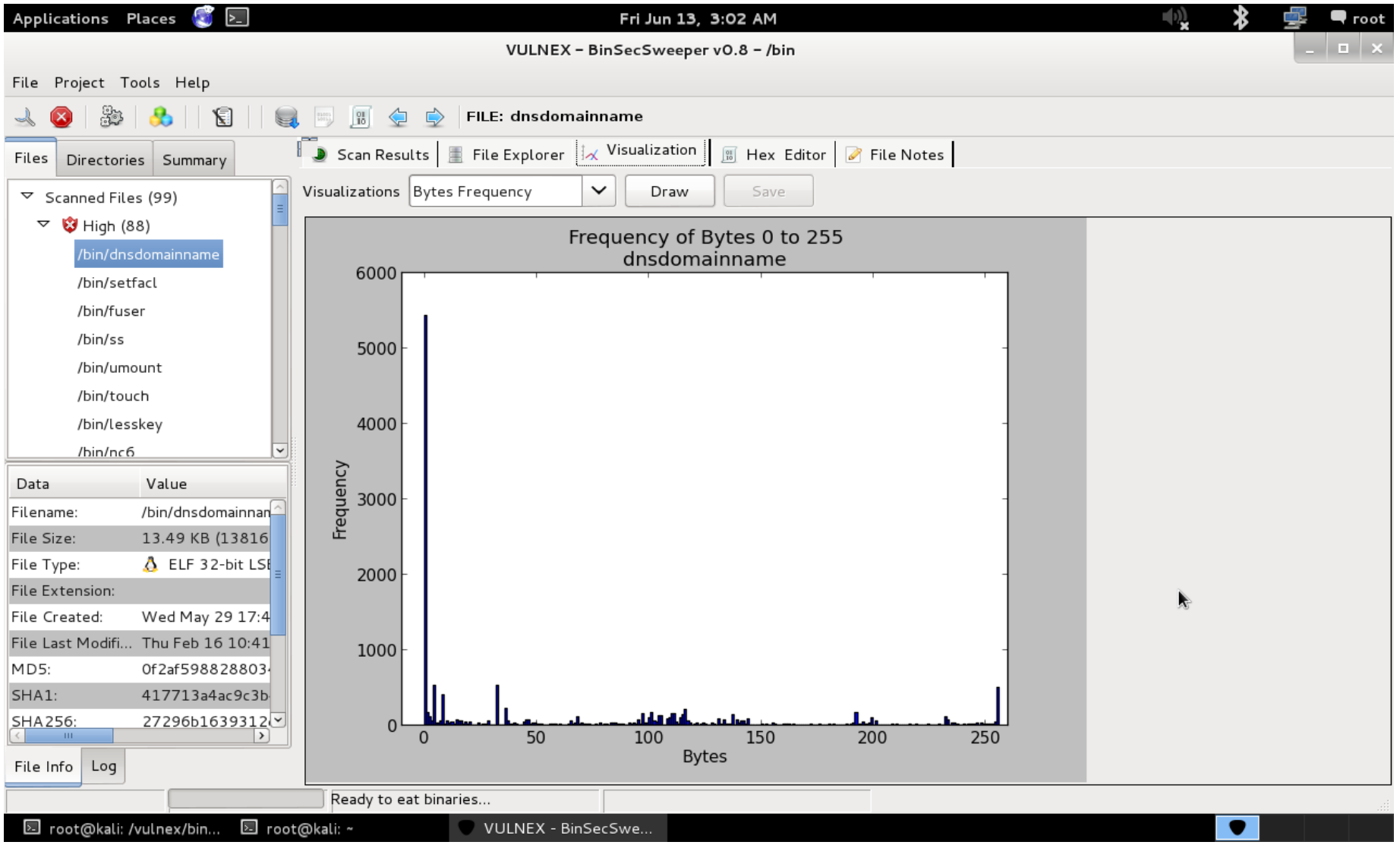
VULNEX

2. BINSECSWEEPER IN ACTION (II)



VULNEX

2. BINSECSWEEPER IN ACTION (III)



VULNEX

2. CURRENT WINDOWS CHECKS

CHECK	DESCRIPTION
Address space layout randomization (ASLR)	Checks if binary has opted the ASLR. Link with /DYNAMICBASE
Stack Cookies (GS)	Verifies if binary was compiled with Stack Cookies protection. Compile with /GS
HotPatch	Checks if binary is prepared for hot patching. Compile with /hotpatch
Compatible with Data Execution Prevention (NXCOMPAT)	Validates if binary has opted hardware Data Execution Prevention (DEP). Link with /NXCOMPAT
Structured Exception Handling (SEH)	Checks if binary was linked with SafeSEH. Link with /SAFESEH
Abobe Malware Classifier	Analyzes binary for malware behavior using machine learning algorithms
Visual Studio Compiler Fingerprinting	Identifies if binary was compiled with Visual Studio and version (2008, 2010 & 2012)
Packer	Checks if binary has been packed
Insecure API	Check if binary uses banned API
VM Detection	Check if binaries contains VM detection code

2. CURRENT LINUX CHECKS

CHECK	DESCRIPTION
Fortify Source	Checks if binary was compiled with buffer overflow protection (bounds checking). Compile with –D_FORTIFY_SOURCE=X
Never eXecute (NX)	Verifies if binary was compiled with NX to reduce the area an attacker can use to perform arbitrary code execution.
Position Independent Code (PIE)	Checks if binary was compiled with PIE to protects against "return-to-text" and generally frustrates memory corruption attacks. Compile with –fPIE -pie
RELocation Read-Only (RELRO)	Validates if binary was compiled with RELRO (partial/full) to harden data sections. Compile with –z,relro,-z,now
Stack Canary	Checks if binary was compiled with stack protector to protect against stack overflows. Compile with –fstack-protector

2. PLUGIN EXAMPLE: WINDOWS ASLR

```
class win_aslr_detect(scanpluginclass):
    def __init__(self):
        super(win_aslr_detect, self).__init__()

        self.RegisterPlugin()

    def RegisterPlugin(self):
        d = {"name":"Windows ASLR Detection",
             "os":"Windows",
             "arch":"any",
             "code":"native"
             }
        self.SetPluginInfoNew(d)

    def ActivatePlugin(self):

        safe = self.risk_red
        istr= ""

        pe_class = self.GetFileParser()
        pe = pe_class.GetFP()

        if pe == None: return

        if pe.OPTIONAL_HEADER.DllCharacteristics & pe_class.DYNAMICBASE_FLAG:
            istr = "ASLR Detected"
            safe = self.risk_green
        else:
            istr = "NO ASLR Detected"
            safe = self.risk_red

        d1 = {"name": self.GetPluginInfoData(),
              "safe":safe,
              "category":"info",
              "title":"Windows ASLR Detection",
              "desc": istr,
             }

        self.SetPluginResultsNew(d1)
```

VULNEX

2. PLUGIN EXAMPLE: LINUX FORTIFY_SOURCE

```
def ActivatePlugin(self):

    fs = 1
    add_data = []
    fs_funcs = []
    count_fs = 0

    elf_class = self.GetFileParser()
    elf = elf_class.GetFP()

    if elf == None: return

    for section in elf.iter_sections():
        if not isinstance(section, SymbolTableSection):
            continue

        if section['sh_entsize'] == 0:
            continue

        for nsym, symbol in enumerate(section.iter_symbols()):
            ss = bytes2str(symbol.name)
            if not "__stack_chk_fail" in ss and "_chk" in ss and not "LIBC" in ss:
                fs = 0
                fs_funcs.append(ss)
                count_fs+=1

    if fs == 0:
        t = "Fortify Source Functions (%s)" % str(count_fs)
        add_data.append((t,fs_funcs))
        d1 = {"name": self.GetPluginInfoData(),
          "safe":self.risk_green,
          "category":"info",
          "title":"Fortify Source Detection",
          "desc": "Fortify Source Detected",
          "add_data":add_data
         }
    else:
        d1 = {"name": self.GetPluginInfoData(),
          "safe":self.risk_red,
          "category":"info",
          "title":"Fortify Source Detection",
          "desc": "NO Fortify Source Detected"
         }

    self.SetPluginResultsNew(d1)
```

VULNEX

2. BINSECSWEEPER: WHERE?

- Sorry, not yet available!
- Download BinSecSweeper software from http://www.vulnex.com/en/binsecsweeper.html

3. Case Studies & Demos

BINSECSWEEPER USE CASES

Developers

- Verify product complies with Software Assurance policies before releasing

IT Pros

- Can assess software in systems for the security posture

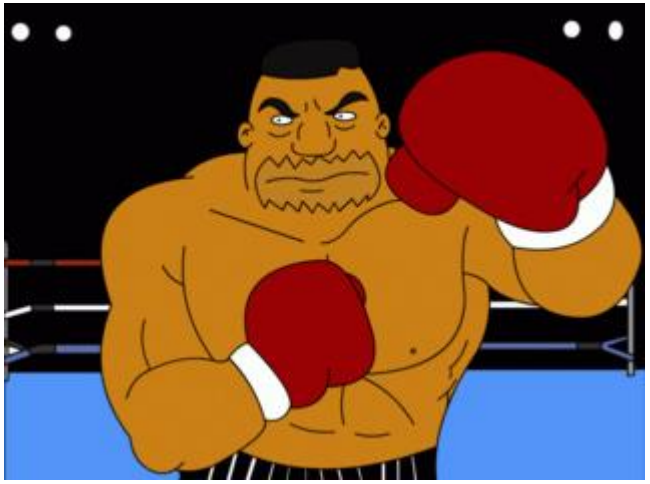
InfoSec & Researchers

- Perform file forensics & analyze malware

VULNEX

3. TIME FOR SOME ACTION

- Case Study I: Verify your own software
- Case Study II: Software Security Posture, ACME inc
- Case Study III: Misc.



VULNEX

3. CASE STUDY I: VERIFY YOUR OWN SOFTWARE

- Is your in-house software following a secure development framework?

- Is your software being checked for:

  1. Compiled with a modern compiler?
  2. Security defenses enabled for Windows or Linux?
  3. No malware included in product?
  4. Using external libraries (DLL, etc.) and what is their security?

3. CASE STUDY I: VERIFY YOUR OWN SOFTWARE

- BinSecSweeper can verify that product (used by development teams):
  - What Visual Studio version has been used? (Windows Only) (MS SDL)
  - What defenses have been enabled?:

Windows	Linux
Stack Cookies	Stack Canary
ASLR	NX
DEP	Fortify Source
SAFESEH	PIE
HotPacthing	RELRO

  - Will audit all files in the project?
- Program security posture: will it Pass / Fail?

3. CASE STUDY II: SOFTWARE SECURITY POSTURE, AMCE INC

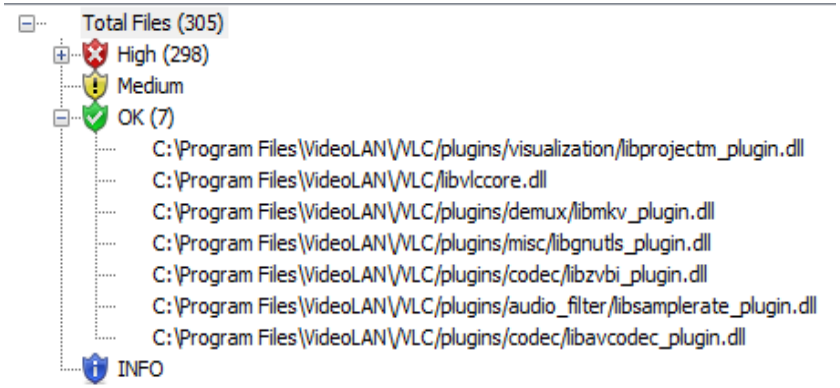
- Do IT know the security posture of all software? You can assess your vendors…

- Now you know where EMET is needed!

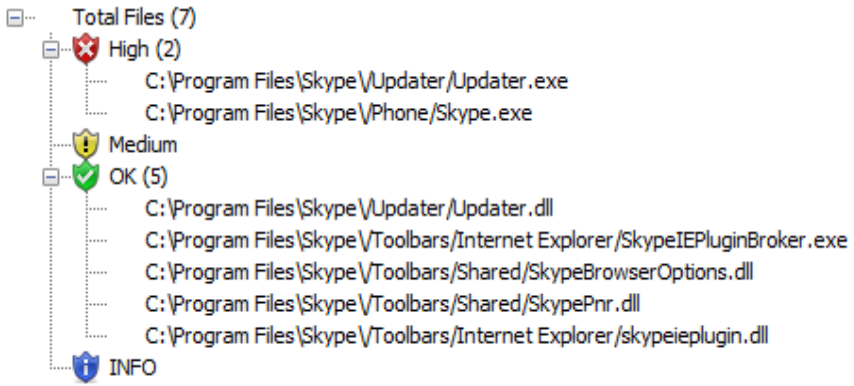
VULNEX

3. CASE STUDY II: SOFTWARE SECURITY POSTURE, AMCE INC

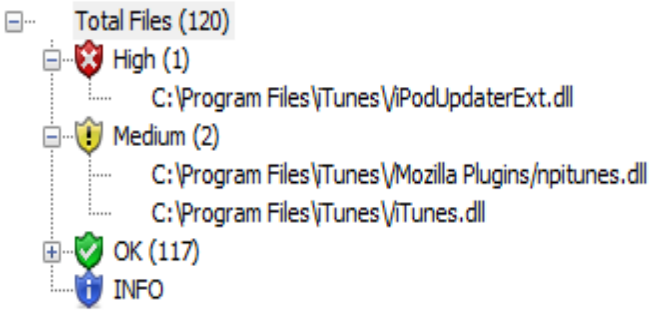
VLC



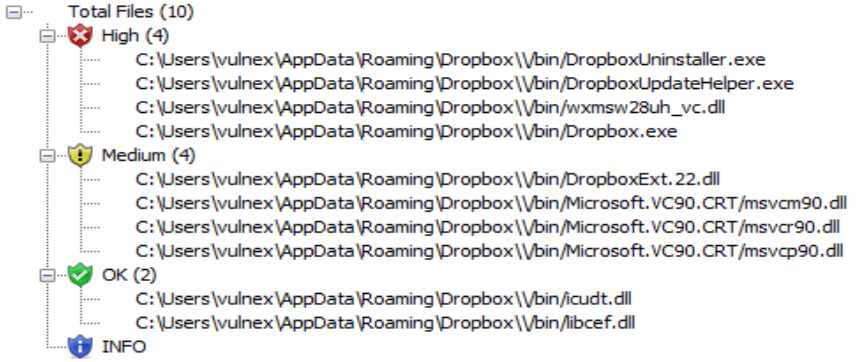
SKYPE



iTunes

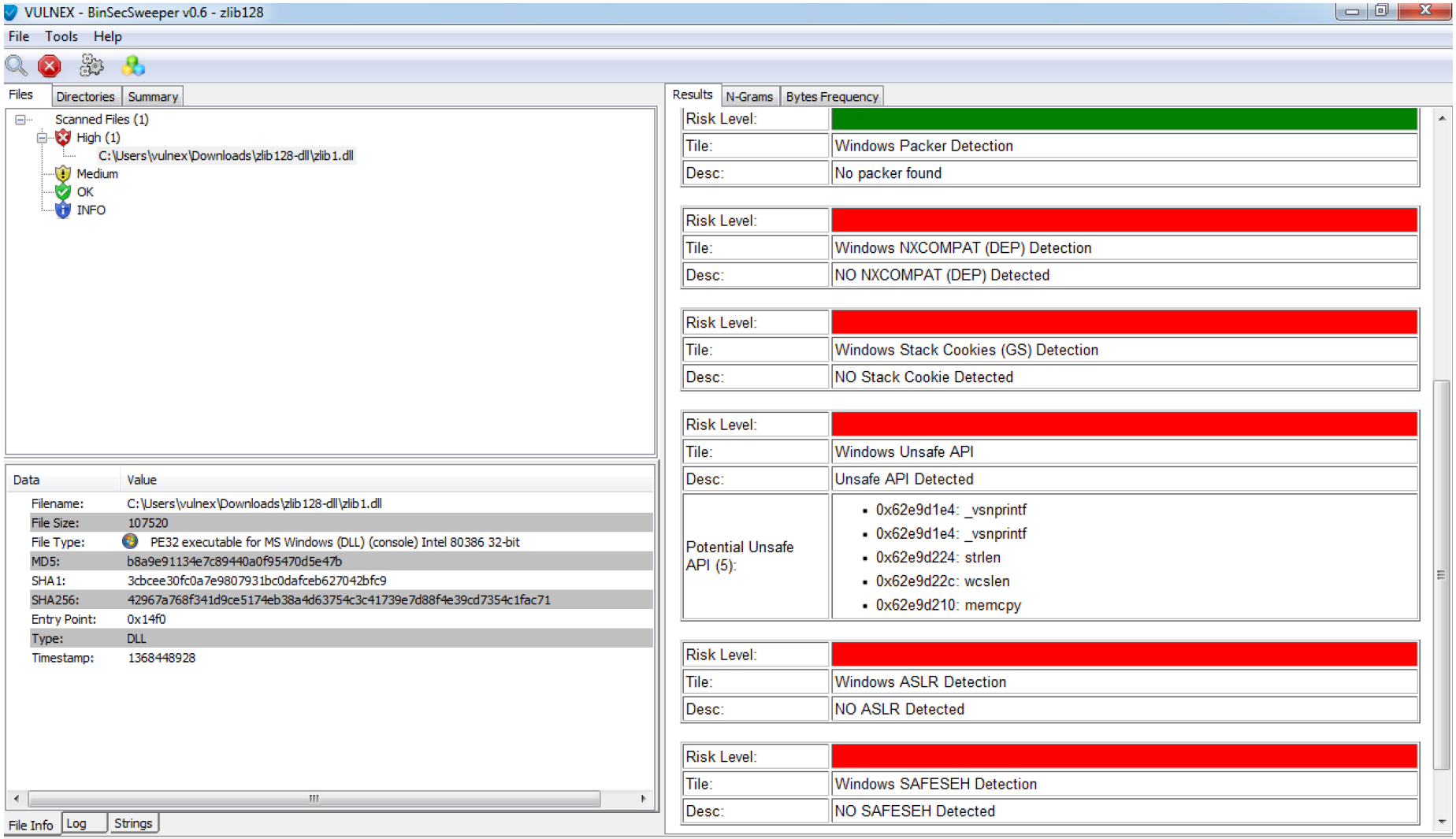


Dropbox



VULNEX

3. CASE STUDY III: ARE YOU COMPILING YOUR APP WITH ZLIB.DLL ?



VULNEX

3. CASE STUDY III: ARE YOUR 3RD PARTY COMPONENTS IMPROVING?

- Python 2.7 -> sqlite3.dll

Risk Level:	
Tile:	Windows NXCOMPAT (DEP) Detection
Desc:	NO NXCOMPAT (DEP) Detected

Risk Level:	
Tile:	Windows ASLR Detection
Desc:	NO ASLR Detected

- Python 3.3 -> sqlite3.dll

Risk Level:	
Tile:	Windows ASLR Detection
Desc:	NO ASLR Detected

3. CASE STUDY III: A DLL INSIDE A WELL-KNOWN SOFTWARE

Risk Level:	
Tile:	Windows ASLR Detection
Desc:	NO ASLR Detected

Risk Level:	
Tile:	Windows NXCOMPAT (DEP) Detection
Desc:	NO NXCOMPAT (DEP) Detected

Risk Level:	
Tile:	Windows Unsafe API
Desc:	Unsafe API Detected
Potential Unsafe API (14):	• 0x00407170: strcpy • 0x004071bc: strncpy • 0x00407168: strcat • 0x00407164: strncat • 0x00407244: wsprintfA • 0x004071d8: sprintf • 0x004071c8: _vsnprintf • 0x004071b8: _snprintf • 0x004071c8: _vsnprintf • 0x004071bc: strncpy • 0x00407164: strncat • 0x004071cc: sscanf • 0x0040716c: strlen • 0x0040715c: memcpy

VULNEX

3. CASE STUDY III: THE MOST COMMON WORD INSIDE A MICROSOFT BINARY?

Total N-Grams

	2	3	4	5	6	7	8	9	10
Total	1208	2150	2464	2535	2560	2557	2516	2452	2376

Top 10 N-Grams

2-gram	Frequency	3-gram	Frequency	4-gram	Frequency	5-gram	Frequency	6-gram	Frequency	7-gram	Frequency	8-gram	Frequency	9-gram	Frequency	10-gram	Frequency
on	119	ion	68	tion	58	croso	44	crosof	44	crosoft	44	icrosoft	44	Microsoft	25	Microsoft	21
ti	111	tio	58	soft	44	osoft	44	rosoft	44	icrosof	44	Microsof	25	icrosoft	21	Mitigation	14
et	78	oft	44	cros	44	icros	44	icroso	44	Microso	25	crosoft	21	microsoft	19	crosoft Co	14
ic	69	et_	44	icro	44	rosof	44	Micros	25	rosoft	21	microsof	19	Attribute	15	icrosoft C	14
io	68	cro	44	roso	44	ation	38	Config	22	microso	19	ttribute	15	itigation	14	t Corporat	11
at	66	get	44	osof	44	Micro	25	osoft	21	ttribut	15	Attribut	15	Mitigatio	14	rosoft Cor	11
in	64	ros	44	get_	43	Confi	22	micros	19	tribute	15	itigatio	14	crosoft C	14	oft Corpor	11
Co	64	oso	44	atio	38	onfig	22	ration	15	Attribu	15	tigation	14	rosoft Co	14	orporation	11
ro	62	sof	44	Micr	25	soft	21	ribute	15	Mitigat	14	Mitigati	14	ft Corpor	11	ft Corpora	11

VULNEX

4. Conclusions

4. VERIFYING SOFTWARE SECURITY POSTURE MATTERS!

- Binaries contain a lot of information!
- The security posture of the software developed by you is important:
  - Security improves Quality
  - Branding (show you care about security)
- How is the security posture of software vendors you use?

4. Q&A

- Thanks!
-  @BinSecSweeper
-  @simonroses | @vulnexsl
- www.vulnex.com

VULNEX