An Attack Surface Metric

Pratyusa K. Manadhata
Carnegie Mellon University

Context: Security Metrics

Software vendors are spending big on security.

How secure is our software?

Are we better-off than before?

Which is more secure: XP or Vista? Ubuntu or Fedora?

Gauging progress is critical for secure software development. We need measurements and metrics.

We Need Metrics Now!

- A long standing research challenge [ACSAC 01, CRA 03, DIMACS 03, CSTB 07]

Toward a Safer and More Secure Cyberspace [CSTB 2007]:

"..though many benefits would flow from the invention of good metrics, the challenge in this cybersecurity research area is particularly great, and some very new ideas will be needed if cybersecurity metricians are to make more progress."

Our Approach: Attack Surface Measurement (ASM)

How can we quantify a software system's security?

Measure the system's attack surface

Motivation: ASM is Useful to both Industry and Consumers

A guide in consumers’ decision making process

A tool in the software development lifecycle to improve security

- design, implementation, testing, deployment, and maintenance

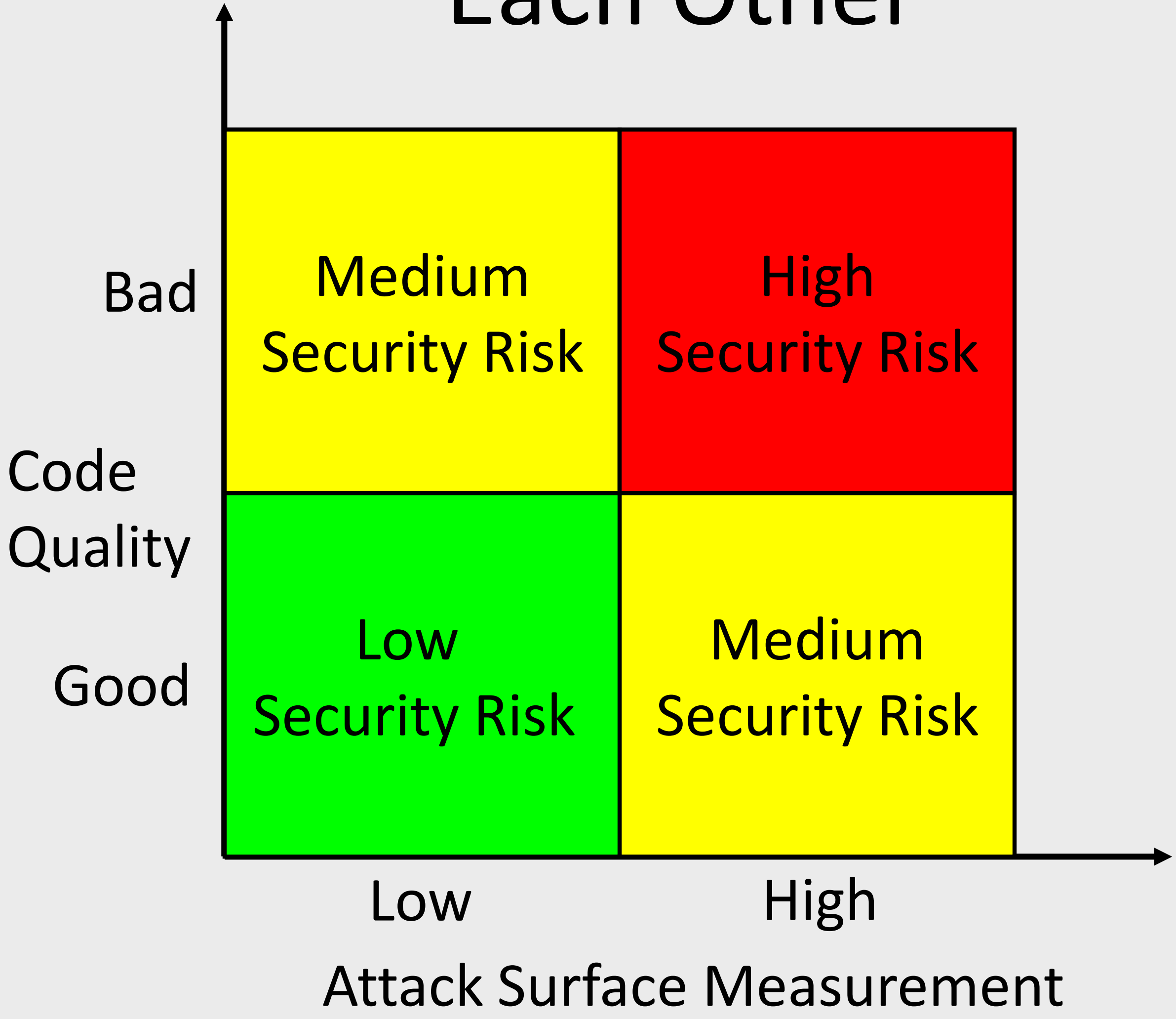
Attack Surface Reduction (ASR) Mitigates Risk

Traditional industry approach: code quality improvement

Software will ship with known and future vulnerabilities

Reduce attack surface to increase the difficulty and decrease the impact of future exploitation

Code Quality and ASR Complement Each Other

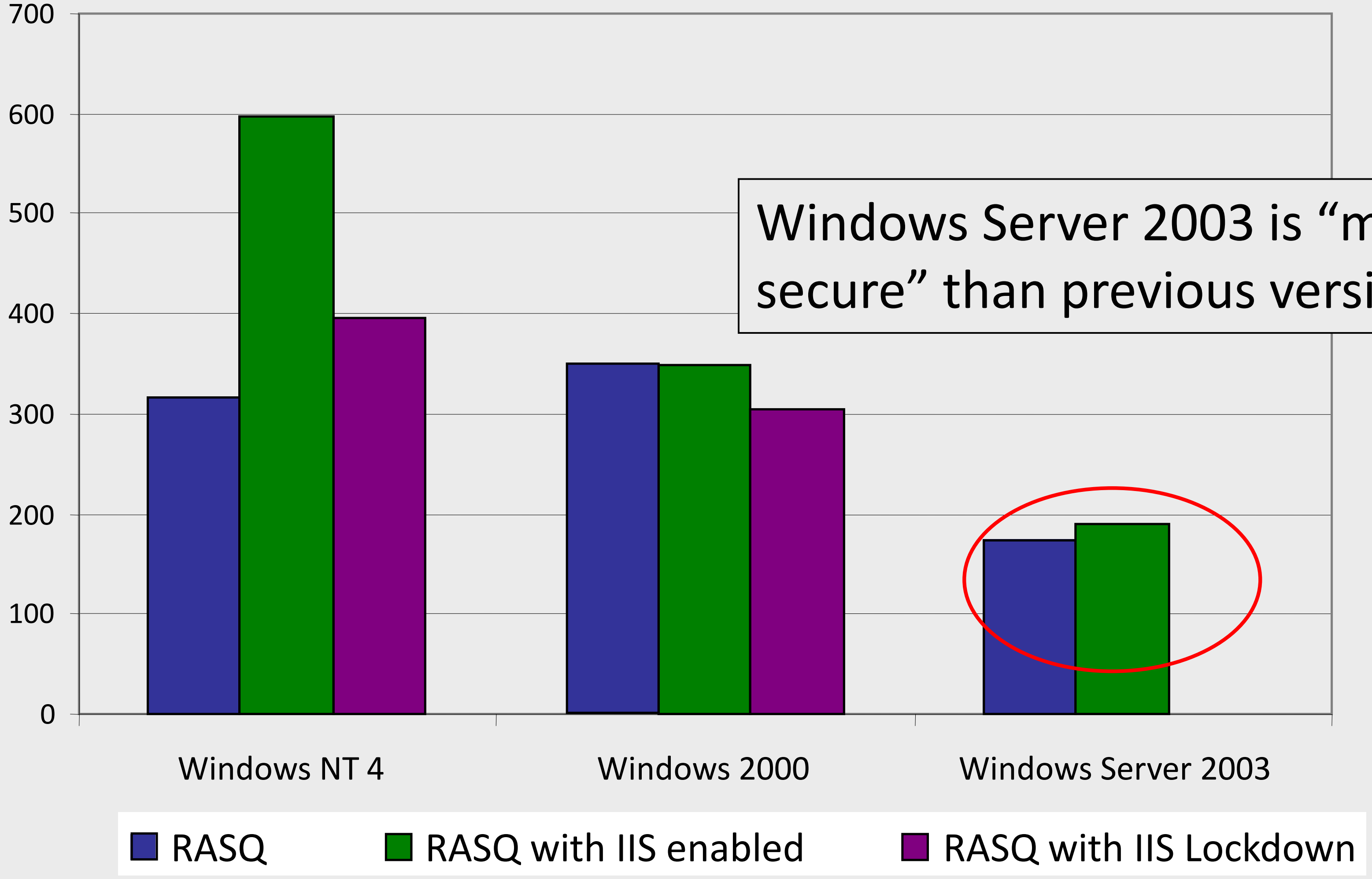


Code Quality

	Low	High
Bad	Medium Security Risk	High Security Risk
Good	Low Security Risk	Medium Security Risk

Attack Surface Measurement

Inspiration: Relative Attack Surface Quotient for 7 Versions of Windows [HPW03]



Windows Server 2003 is “more secure” than previous versions.

Linux Attack Surface Measurements

Attack Vector	Debian	RH Default	RH Facilities	RH Used
Open socket	15	12	40	41
Open RPC endpoint	3	3	3	3
Services running as root	21	26	29	30
Services running as nonroot	3	6	8	8
Setuid root programs	54	54	72	72
Local user accounts	21	25	33	34
User id = root accounts	0	4	3	3
Unpassworded accounts	0	0	2	2
Nobody account	1	1	1	1
Weak file permission	7	7	21	37
Scripts enabled	1	2	2	2

Confirms perception that Debian is more secure than RedHat

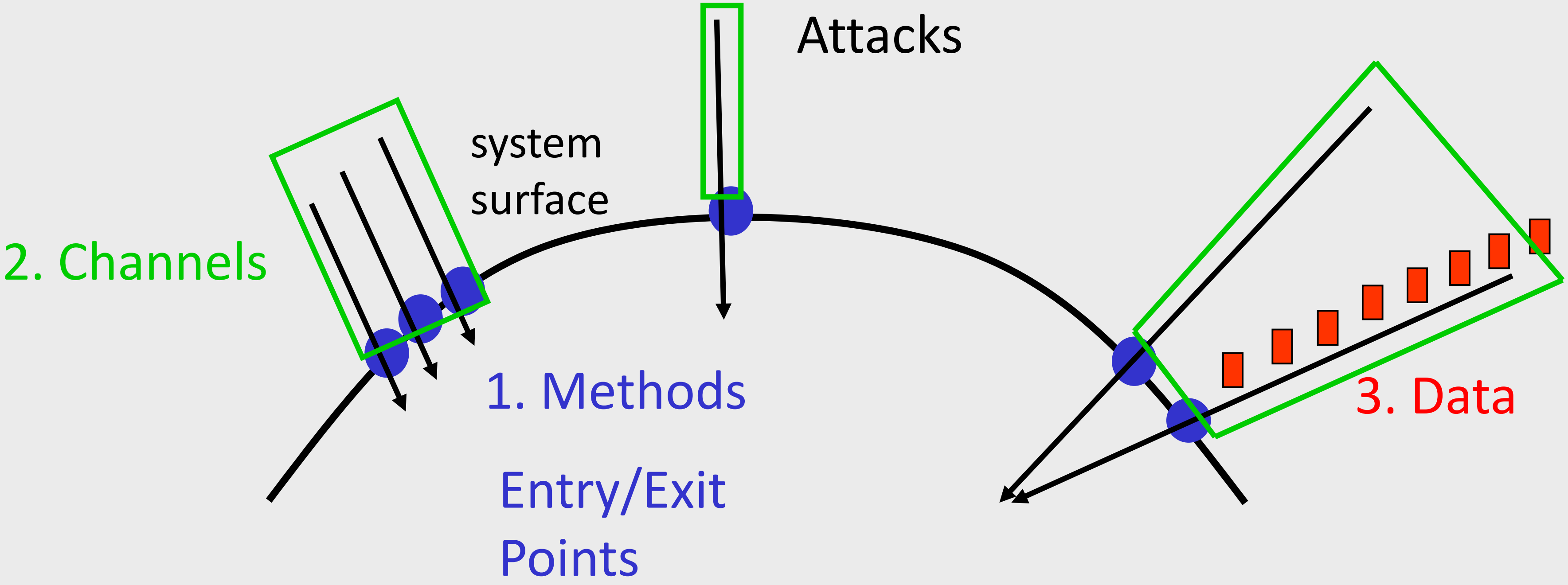
Lessons Learned from Windows and Linux Measurements

- Measurement method is ad-hoc
- Requires a security expert
- Focus is on measuring the attack surfaces of operating systems

Research Goals

- Formalize the notion of attack surface
- Introduce a systematic attack surface measurement method
  - Anyone, anywhere, anything
- Validate the method
- Demonstrate the uses of the method

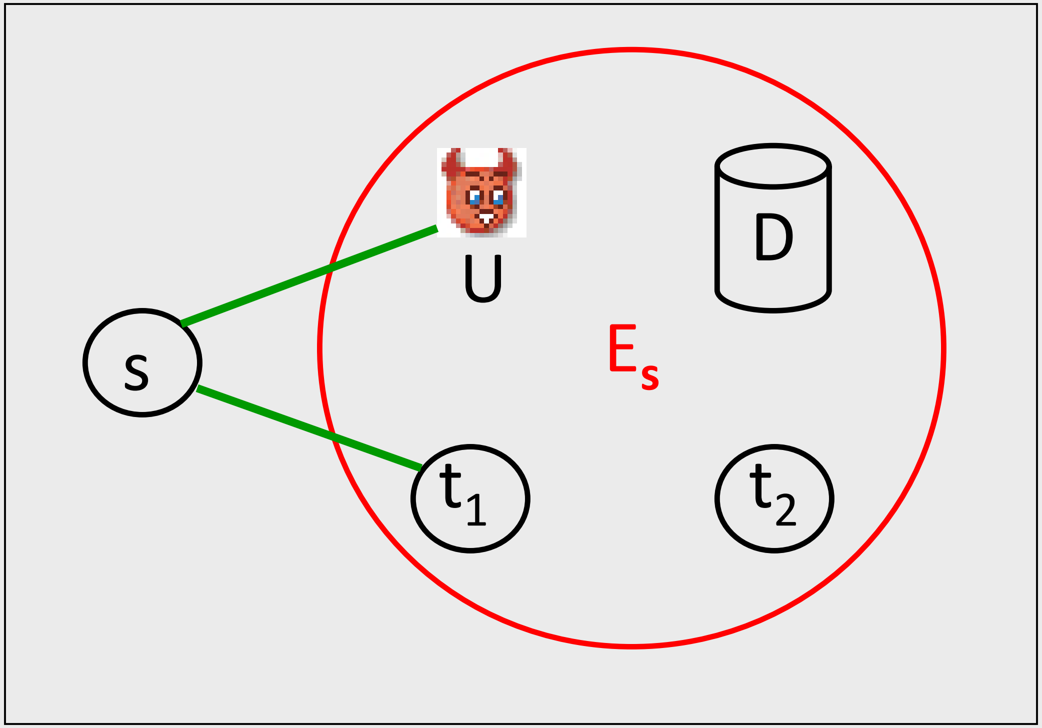
Intuition Behind Attack Surfaces



Hence we define a system’s attack surface in terms of the system’s resources (i.e., methods, channels, and data items).

Model of a System and its Environment

A system, s, and its environment, $E_s = \langle U, D, T=\{t_1, t_2\}\rangle$.



Formal model uses I/O automata [LT89] .

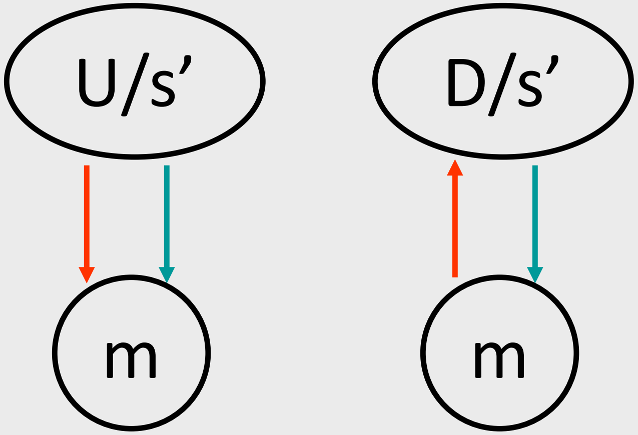
Not All Resources Are Part of the Attack Surface

- Only those resources that the attacker can use to send data into or receive data from the system are relevant.

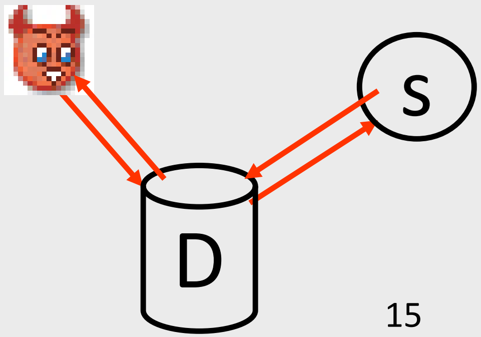
- We introduce the formal entry point and exit point framework to identify the relevant resources.

Entry Point and Exit Point Framework

- Entry Points/Exit Points
  - Direct (input/output action)
  - Indirect (internal action)



- Channels (e.g., sockets and pipes)
  - c ϵ Res(m.pre)
- Untrusted Data Items (e.g., files)
  - d ϵ Res(m.post), d ϵ Res(m.pre)



Attack Surface Definition

- Definition
  - M: set of entry points and exit points
  - C: set of channels
  - I: set of untrusted data items.

attack surface = $\langle M, C, I \rangle$

Theorem: Given an environment, E, if $AS(A) \geq AS(B)$, then $Attacks(A||E) \supseteq Attacks(B||E)$.

Not All Resources Contribute Equally to the Attack Surface

- Contribution $\propto$ Damage Potential

  Contribution $\propto$ (Attacker Effort) $^{-1}$

- Contribution = $\dfrac{\text{Damage Potential}}{\text{Attacker Effort}}$

Higher Damage Potential $\Rightarrow$ Stronger m.post

$\Rightarrow$ more methods can follow m

Lower Attacker Effort $\Rightarrow$ Weaker m.pre

$\Rightarrow$ m can follow more methods

Attack Surface Measurement (ASM)

- ASM(A) ≥ ASM(B) if there exists a nonempty set, R, of resources s.t.

  $\forall r \in R.\ \text{contribution}(r, A) \geq \text{contribution}(r, B).$

Theorem: Given an environment, E, if ASM(A) ≥ ASM(B), then Attacks(A||E) $\supseteq$ Attacks(B||E).

Quantitative Attack Surface Measurement

- Assume der: method → Q.
  - Similarly, for channel and data.

$$\text{ASM} = \left\langle \sum_{m \in M} der(m), \ \sum_{c \in C} der(c), \ \sum_{d \in I} der(d) \right\rangle$$

- Analogous to risk modeling

$$\sum_{m \in M} \underbrace{p(m)}_{\text{probability = 1}} \underbrace{der(m)}_{\text{consequence}}$$

Abstract Measurement Method

1. Identify a set, M, of entry points and exit points, a set, C, of channels, and a set, I, of untrusted data items.
2. Estimate each relevant resource’s damage potential-effort ratio, der.
3. Compute Attack Surface Measurement =

$$\left\langle \sum_{m \,\square\, M} \mathrm{der}(m),\ \sum_{c \,\square\, C} \mathrm{der}(c),\ \sum_{d \,\square\, I} \mathrm{der}(d) \right\rangle .$$

C Measurement Method and Examples

- FTP Servers
  - ProFTP 1.2.10 , Wu-FTP 2.6.2
- IMAP Servers
  - Courier 4.0.1, Cyrus 2.2.10

Step 1: Identify Relevant Resources

- Entry Points and Exit Points
  - Static analysis
  - C library methods (e.g., read) for data exchange
  - Call graph

- Channels and Untrusted Data Items
  - Run time monitoring
  - Open channels
  - Data read and written

Step 2: Damage Potential-Effort Ratio

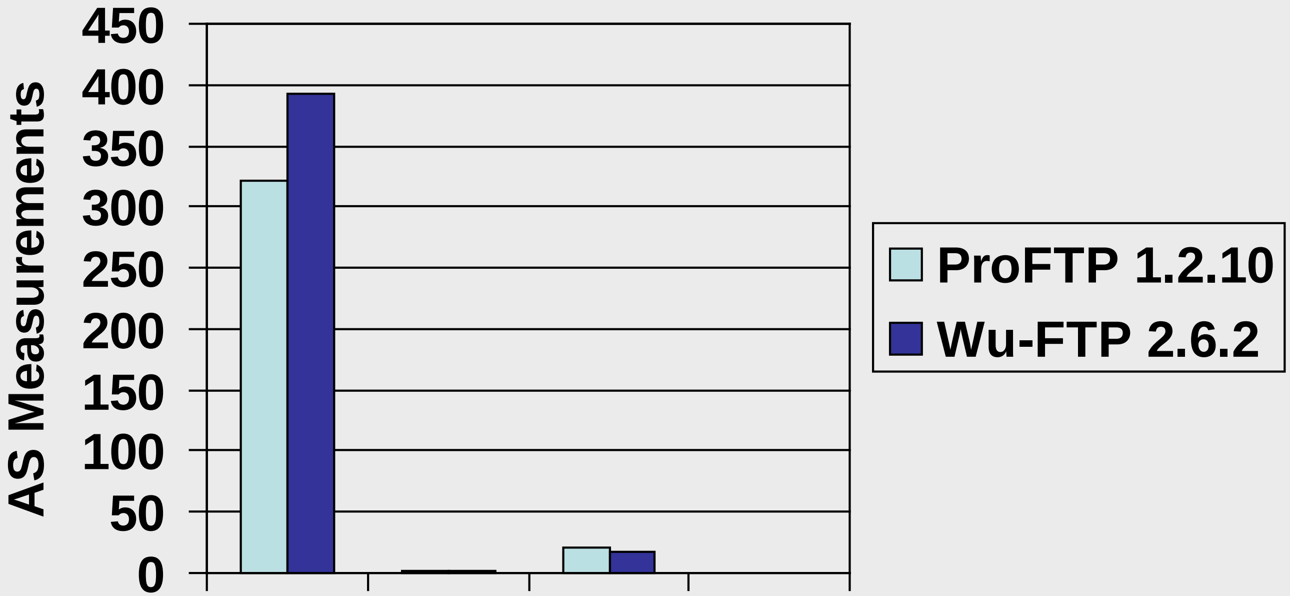
Resource	Damage Potential	Attacker Effort
Method	Privilege	Access Rights
Channel	Protocol	Access Rights
Data Items	Type	Access Rights

Impose a total ordering among the values of the attributes and assign numeric values accordingly, e.g.,

root = 5 and auth = 3.

FTP Measurement Results

ProFTP = ⟨312.9, 1.0, 18.9⟩, Wu-FTP = ⟨392.3, 1.0, 17.6⟩



Use domain knowledge to decide which dimension presents more risk and choose accordingly.

Validation

- Validating a software measure is hard [KPF97,….]
  - security metric is even harder

Software measure	Attack surface	MS Bulletins, Expert Survey
Prediction System	Security Risk	IO Automata Model, Patch Analysis, Anecdotal Evidence

Liu and Traore independently validated our metric [LT07].

Validating the Measurement Method

Key Assumptions

- Three dimensions of the attack surface
- Damage potential-effort ratio
- Six attributes
  - method privilege, method access rights, channel protocol, channel access rights, data item type, and data item access rights

Statistical Analysis of Microsoft Security Bulletins (MSB)

- An MSB mentions a vulnerability and resources needed for exploitation
- Are methods, channels, and data used in the exploitation?
- Analyzed MSBs from 2004-2006

Methods	✓
Channels	✓
Data	✓

Results: The Attributes are Indicators of Damage Potential and Effort

Attribute	Significance	Correlation
Privilege	✓	✓
Method Access Rights	✓	✓
Protocol	✓	?
Channel Access Rights	✓	✓
Type	✓	?
Data Access Rights	✓	✓

Expert Linux System Administrator Survey

- MSB has no data relevant to a resource’s attackability
  - Could not validate damage potential-effort ratio
- Surveys are widely used to collect a wide range of data
  - Prior work uses surveys to validate measures [K87, ….]
  - Feedback from one target user group (Industrial collaboration for other target user group)
  - W.r.t. Linux (MSB w.r.t. Windows)

Results: A Majority of the Subjects Agree With Our Measurement Method

Methods	✓
Channels	✓
Data	✓

Damage Potential-Effort Ratio	✓

Privilege	✓
Method Access Rights	✓
Protocol	?
Channel Access Rights	✓
Type	?
Data Access Rights	✓

Validating the Prediction System

- Show that if system A is more secure than system B, then ASM(A) < ASM(B)
- Assumption: Vulnerability patches improve software security
  - ASM(After Patch) < ASM(Before Patch)

Patches reduce attack surface measurement

Results: A Majority of the Patches Reduce ASM

Software	Percentage of Patches that reduce ASM	Significance ($p< 0.05$)
Firefox 2.0	67%	✓
ProFTP (all)	70%	✓
All NVD Bulletins	76.9%	✓

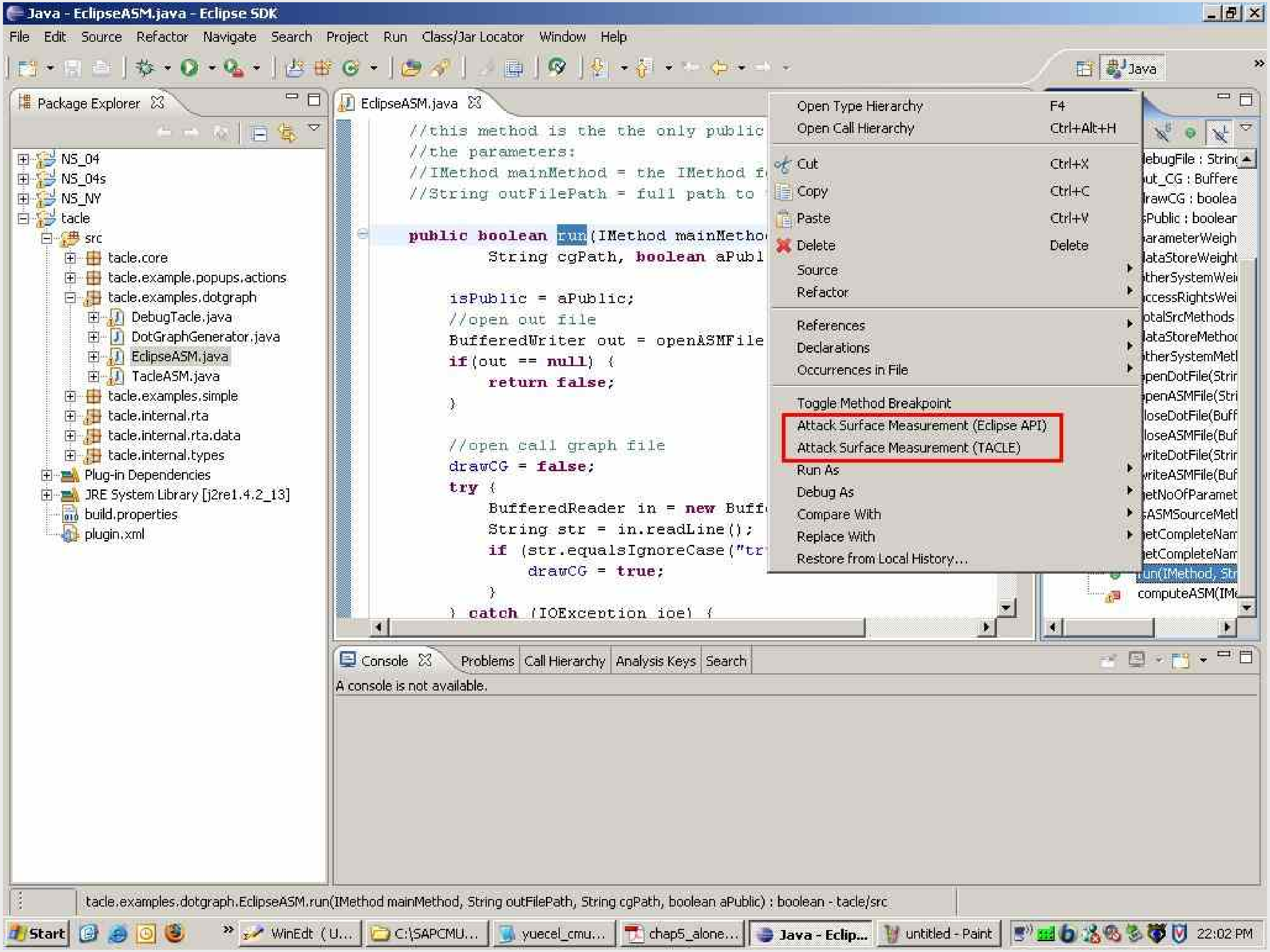
Anecdotal Evidence from Industry

- Microsoft
  - Sasser Worm
  - Nachi Worm
  - Zotob Worm
- Firefox 2.0
  - SSL buffer overflow

Collaboration with SAP

- SAP is world's largest provider of enterprise-scale software
  - Complex technology platforms and business applications
- Demonstrate that the measurement method scales to enterprise-scale software
- Receive feedback from software architects and developers

Java Measurement Tool Screenshot



```
//this method is the the only public
//the parameters:
//IMethod mainMethod = the IMethod fo
//String outFilePath = full path to

public boolean run(IMethod mainMethoc
        String cgPath, boolean aPubl

    isPublic = aPublic;
    //open out file
    BufferedWriter out = openASMFile
    if(out == null) {
        return false;
    }

    //open call graph file
    drawCG = false;
    try {
        BufferedReader in = new Buffe
        String str = in.readLine();
        if (str.equalsIgnoreCase("tr
            drawCG = true;
        }
    } catch (IOException ioe) {
```

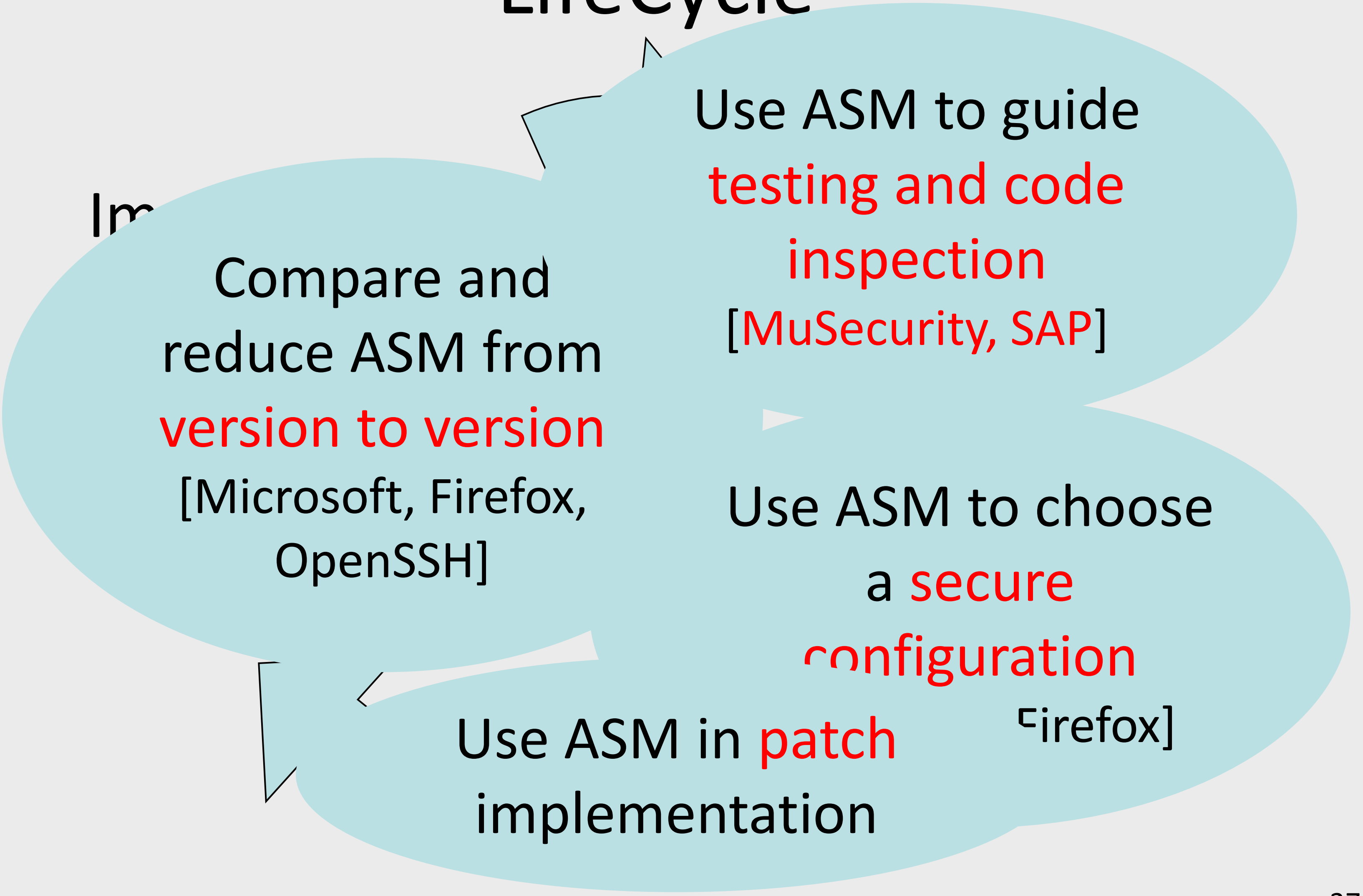
```
A console is not available.
```

tacle.examples.dotgraph.EclipseASM.run(IMethod mainMethod, String outFilePath, String cgPath, boolean aPublic) : boolean - tacle/src

Results

- Measured the attack surface of a key component of SAP component
  - Measurement results conform to expectation
  - Detailed tool output, incremental analysis, and what-if scenarios are useful for attack surface reduction
  - Lessons learned

ASM in Software Development LifeCycle



Im

Use ASM to guide testing and code inspection [MuSecurity, SAP]

Compare and reduce ASM from version to version [Microsoft, Firefox, OpenSSH]

Use ASM to choose a secure configuration

Firefox]

Use ASM in patch implementation

Future Work: Software Development

- Range analysis

- Other uses
  - ``Safe’’ software composition
  - Testing, deployment, maintenance

Future Work: Software Consumers

- Attack surface measurement in the absence of source code
  - Components as Entry/Exit points
  - Channels and Data as before

- Multiple metrics are needed for decision support

How do we combine multiple measures?

Related Work-1

- Prior work assumes the knowledge of vulnerabilities [AB95, VGMCM96, ODM99…]

- ASM is based on a system’s inherent properties
  - Formal framework encompasses past, present, and future vulnerabilities
  - Complementary to prior work

Related Work-2

- Prior work takes an attacker-centric approach [S99, MBFB05, LB08,..]

- ASM takes a system-centric approach
  - Depends on a system's design
  - No assumptions about the attacker
  - Can be used as a tool in software development

Related Work-3

- Prior work is conceptual in nature and haven’t been applied to real systems [AB95, MGVT02, S04,..]

- We measured the attack surfaces of real-world software
  - FTP servers, IMAP servers
  - SAP business applications

Summary

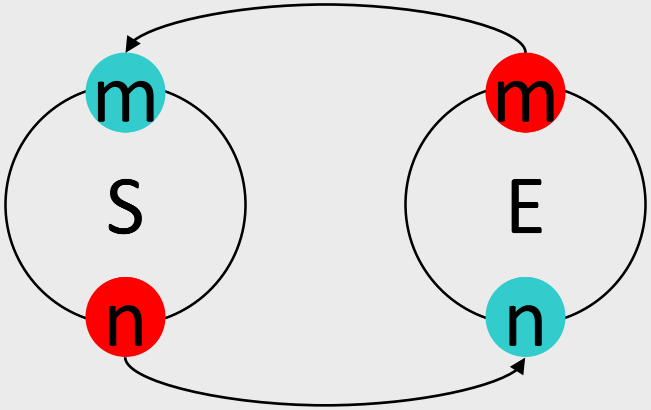
- Introduced a pragmatic approach for security measurement
  - Software industry found it useful [Microsoft, Firefox, OpenSSH, MuSecurity, SAP, ..]
- First step in the grander challenge of security metrics
  - Understanding over time will lead to more meaningful metrics

Acknowledgements: Jeannette Wing, Roy Maxion, Virgil Gligor, Mike Reiter, Yuecel Karabulut, Effrat Keren, Dilsun Kaynar, Kymie Tan, Gourav Kataria, Miles McQueen, Mark Flynn, Michael Howard, Paul Hoffman, Mary Shaw, and Survey Participants.

Backups

I/O Automata [LT89]

- Action Signature
  - Input, Output, Internal actions
  - Pre and Post conditions m.pre and m.post



- Composition
  - $E_s = (U_{io} \,||\, D_{io} \,||\, (\,||\, t_{io} \,))$
  - $P = s_{io} \,||\, E_s$

t_{io}□T_{io}

Validation of the Attributes

- An MSB has a severity rating and mentions the six resources attributes

Impact	Damage Potential
Difficulty	Attacker Effort

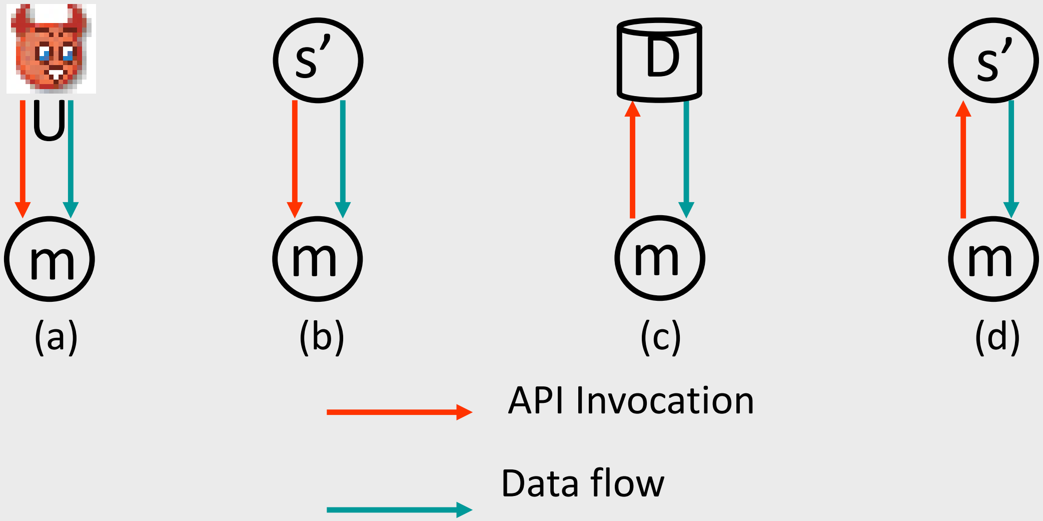
Significant Predictor	Two sided Z-test ($p < 0.05$)
Correlation	Sign of Coefficient in Ordered Logistic Regression

Inspiration: Howard’s Relative Attack Surface Quotient (RASQ)[H03]

- Howard’s informal RASQ Measurement Method
  - Identify a system’s attack vectors
  - Assign weights to the attack vectors to reflect their attackability
  - RASQ = sum of the weighted counts of the attack vectors

Direct Entry Points

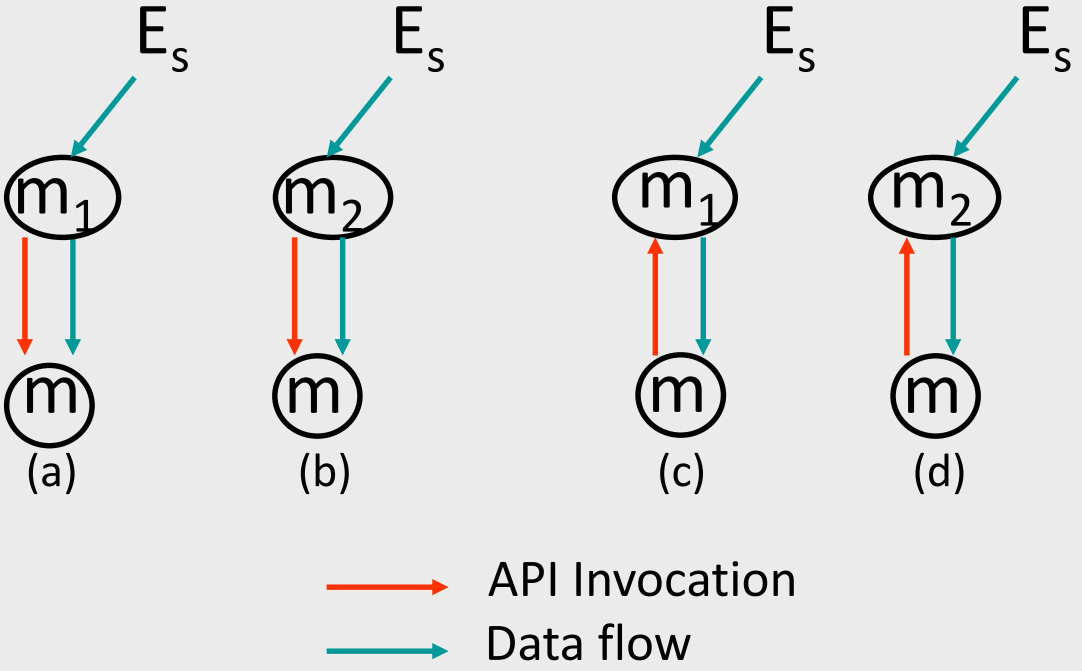
Methods that directly receive data.



direct entry point: an input action with a matching output action

Indirect Entry Points

Methods that indirectly receive data.



indirect entry point: internal action

(m1.post => m.post) Λ

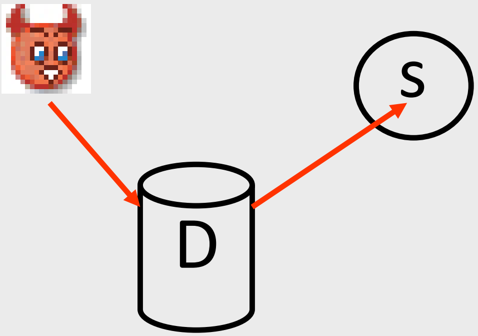
(d ϵ Res(m1.post) Λ d ϵ Res(m.pre))

Channels and Data

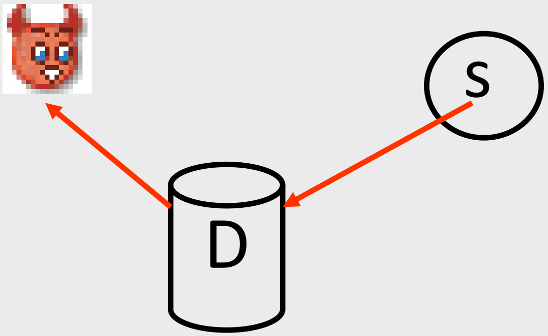
Channels (e.g., sockets and pipes)

- c є Res(m.pre)

Untrusted Data Items (e.g., files)



d є Res(m.post)



d є Res(m.pre)

Definition of An Attack

Attacks (s_{io}) = Set of executions of (s_{io} || E_s) that contain either an input action or output action of s_{io}.

Not All Resources Contribute Equally to the Attack Surface

- contribution α damage potential
  α 1/attacker effort
- r1 ≥ r2 if higher damage potential and/or lower attacker effort

m(MA, CA, DA, MB, CB, DB)

pre: $P_{pre} \wedge (MA \geq m.ef) \wedge (CA \geq c.ef) \wedge (DA \geq d.ef)$

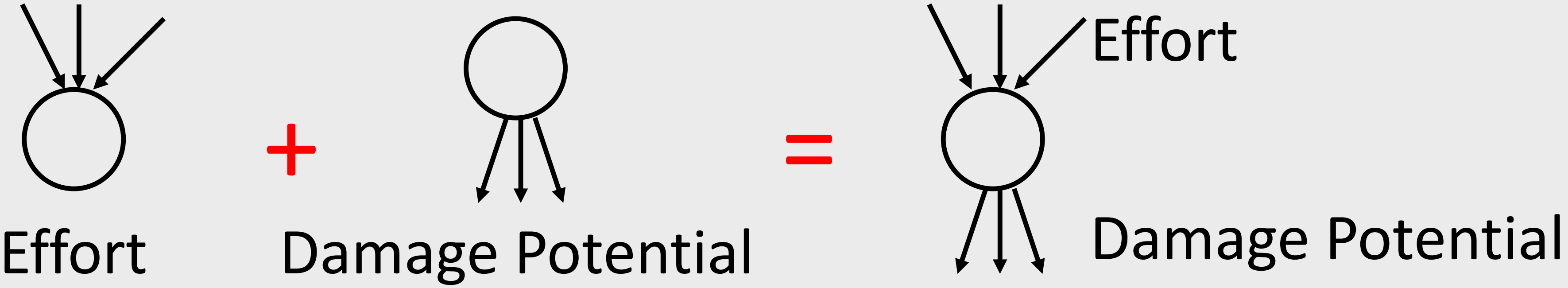
post: $P_{post} \wedge (MB \geq m.dp) \wedge (CB \geq c.dp) \wedge (DB \geq d.dp)$

Damage Potential-Effort Ratio

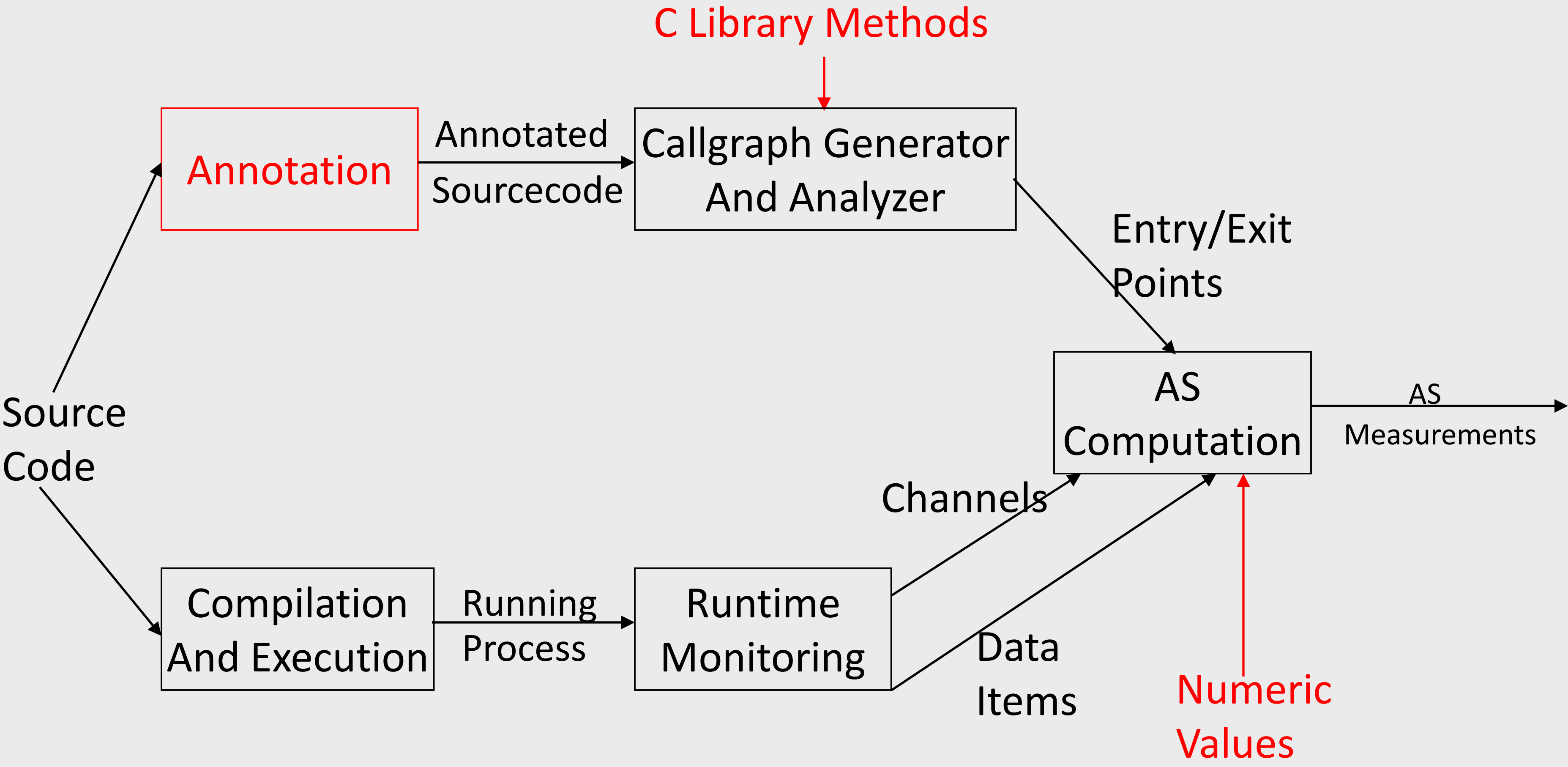
- Contribution $\propto$ Damage Potential

  Contribution $\propto$ (Attacker Effort)$^{-1}$

- Contribution = $\dfrac{\text{Damage Potential}}{\text{Attacker Effort}}$



C Measurement Method



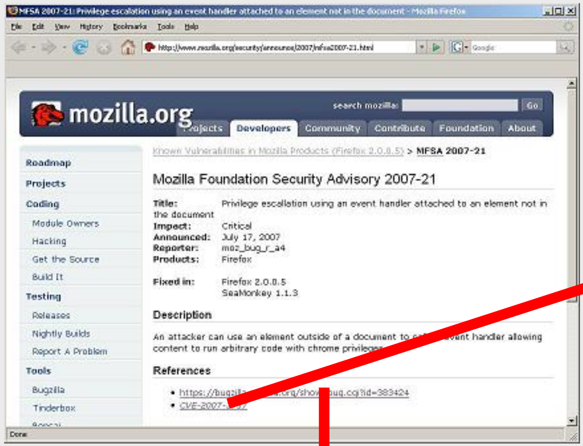
Survey Methodology

- Email survey of experienced Linux system administrators
  - Diverse background and geographic location
- Questions on a five point Likert scale [L32]
  - Pretesting and interviewing to avoid bias
  - Self-selection bias
- Descriptive analysis techniques
  - Central tendency bias
  - %age of agreement, disagreement, and neither
  - t-test ($p < 0.05$)

Not All Patches Are Relevant

- Heuristics: vulnerability type determines patch relevance
  - Use National Vulnerability Database (NVD) type information
  - Infer type if missing
- Not all relevant patches reduce the attack surface
- Consider local effect of a patch

Data Collection for Firefox 2.0



MFSA 2007-21: Privilege escalation using an event handler attached to an element not in the document - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://www.mozilla.org/security/announce/2007/mfsa2007-21.html
mozilla.org
search mozilla: Go
Projects Developers Community Contribute Foundation About
Roadmap
Projects
Coding
Module Owners
Hacking
Get the Source
Build It
Testing
Releases
Nightly Builds
Report A Problem
Tools
Bugzilla
Tinderbox
Done

Known Vulnerabilities in Mozilla Products (Firefox 2.0.0.5) > MFSA 2007-21

Mozilla Foundation Security Advisory 2007-21

Title: Privilege escallation using an event handler attached to an element not in the document
Impact: Critical
Announced: July 17, 2007
Reporter: moz_bug_r_a4
Products: Firefox

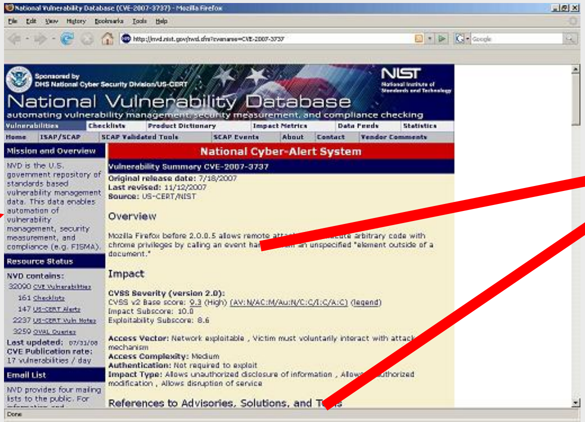
Fixed in: Firefox 2.0.0.5
SeaMonkey 1.1.3

Description

An attacker can use an element outside of a document to c event handler allowing content to run arbitrary code with chrome privileges.

References

- https://bugzilla.mozilla.org/show_bug.cgi?id=383424
- CVE-2007-3737



National Vulnerability Database (CVE-2007-3737) - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://nvd.nist.gov/nvd.cfm?cvename=CVE-2007-3737
Sponsored by DHS National Cyber Security Division/US-CERT
NIST National Institute of Standards and Technology
National Vulnerability Database
automating vulnerability management, security measurement, and compliance checking
Vulnerabilities Checklists Product Dictionary Impact Metrics Data Feeds Statistics
Home ISAP/SCAP SCAP Validated Tools SCAP Events About Contact Vendor Comments
Done

Mission and Overview

NVD is the U.S. government repository of standards based vulnerability management data. This data enables automation of vulnerability management, security measurement, and compliance (e.g. FISMA).

Resource Status

NVD contains:
32090 CVE Vulnerabilities
161 Checklists
147 US-CERT Alerts
2237 US-CERT Vuln Notes
3259 OVAL Queries
Last updated: 07/31/08
CVE Publication rate: 17 vulnerabilities / day

Email List

NVD provides four mailing lists to the public. For

National Cyber-Alert System

Vulnerability Summary CVE-2007-3737

Original release date: 7/18/2007
Last revised: 11/12/2007
Source: US-CERT/NIST

Overview

Mozilla Firefox before 2.0.0.5 allows remote att ecute arbitrary code with chrome privileges by calling an event ha an unspecified "element outside of a document."

Impact

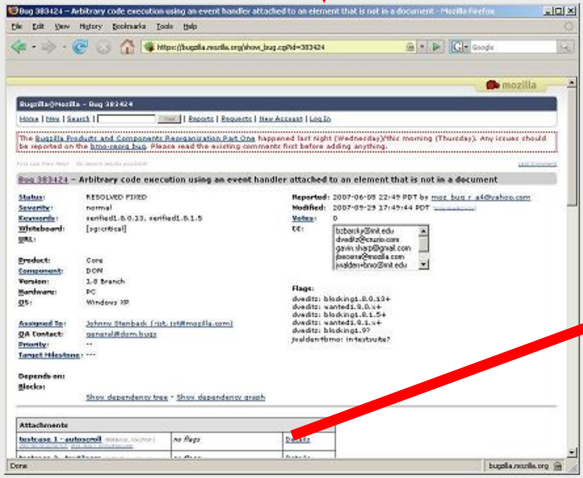
CVSS Severity (version 2.0):
CVSS v2 Base score: 9.3 (High) (AV:N/AC:M/Au:N/C:C/I:C/A:C) (legend)
Impact Subscore: 10.0
Exploitability Subscore: 8.6

Access Vector: Network exploitable , Victim must voluntarily interact with attack mechanism
Access Complexity: Medium
Authentication: Not required to exploit
Impact Type: Allows unauthorized disclosure of information , Allows uthorized modification , Allows disruption of service

References to Advisories, Solutions, and T s

Relevant?

MITRE



Bug 383424 – Arbitrary code execution using an event handler attached to an element that is not in a document - Mozilla Firefox
File Edit View History Bookmarks Tools Help
https://bugzilla.mozilla.org/show_bug.cgi?id=383424
mozilla
Home | New | Search | Find | Reports | Requests | New Account | Log In
Done
bugzilla.mozilla.org

Bugzilla@Mozilla – Bug 383424

The Bugzilla Products and Components Reorganization Part One happened last night (Wednesday)/this morning (Thursday). Any issues should be reported on the bmo-reorg bug. Please read the existing comments first before adding anything.

Bug 383424 – Arbitrary code execution using an event handler attached to an element that is not in a document

Status: RESOLVED FIXED
Severity: normal
Keywords: verified1.8.0.13, verified1.8.1.5
Whiteboard: [sg:critical]
URL:

Product: Core
Component: DOM
Version: 1.8 Branch
Hardware: PC
OS: Windows XP

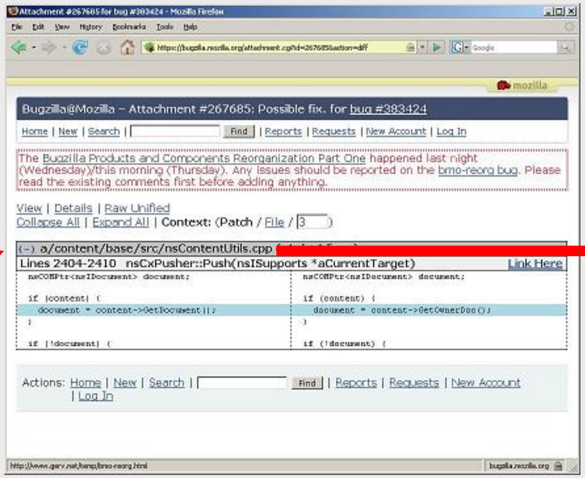
Assigned To: Johnny Stenback (:jst, jst@mozilla.com)
QA Contact: general@dom.bugs
Priority: --
Target Milestone: ---

Depends on:
Blocks:
Show dependency tree - Show dependency graph

Reported: 2007-06-05 22:49 PDT by moz_bug_r_a4@yahoo.com
Modified: 2007-09-29 17:49:44 PDT
Votes: 0
CC: bzbarsky@mit.edu
dveditz@cruzio.com
gavin.sharp@gmail.com
jbecerra@mozilla.com
jwalden+bmo@mit.edu

Flags:
dveditz: blocking1.8.0.13+
dveditz: wanted1.8.0.x+
dveditz: blocking1.8.1.5+
dveditz: wanted1.8.1.x+
dveditz: blocking1.9?
jwalden+bmo: in-testsuite?

Attachments		
testcase 1 - autoscroll	no flags	Details



Attachment #267685 for bug #383424 - Mozilla Firefox
File Edit View History Bookmarks Tools Help
https://bugzilla.mozilla.org/attachment.cgi?id=267685&action=diff
mozilla
http://www.gerv.net/temp/bmo-reorg.html
bugzilla.mozilla.org

Bugzilla@Mozilla – Attachment #267685: Possible fix. for bug #383424

Home | New | Search | Find | Reports | Requests | New Account | Log In

The Bugzilla Products and Components Reorganization Part One happened last night (Wednesday)/this morning (Thursday). Any issues should be reported on the bmo-reorg bug. Please read the existing comments first before adding anything.

View | Details | Raw Unified
Collapse All | Expand All | Context: (Patch / File / 3)

(-) a/content/base/src/nsContentUtils.cpp
Lines 2404-2410 nsCxPusher::Push(nsISupports *aCurrentTarget) Link Here

```
  nsCOMPtr<nsIDocument> document;              nsCOMPtr<nsIDocument> document;

  if (content) {                               if (content) {
    document = content->GetDocument();           document = content->GetOwnerDoc();
  }                                            }

  if (!document) {                             if (!document) {
```

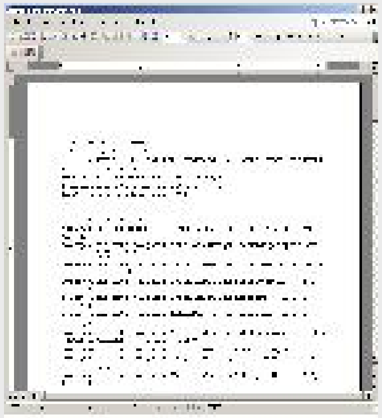
Actions: Home | New | Search | Find | Reports | Requests | New Account | Log In

Reduces ASM?

Java Measurement Method

- Focus on method dimension
- Entry Points and Exit Points
  - Call graph
  - Interface methods, methods invoking other systems' interfaces and Java I/O library methods
- Damage Potential-Effort Ratio
  - Use SAP's threat modeling process to assign numbers

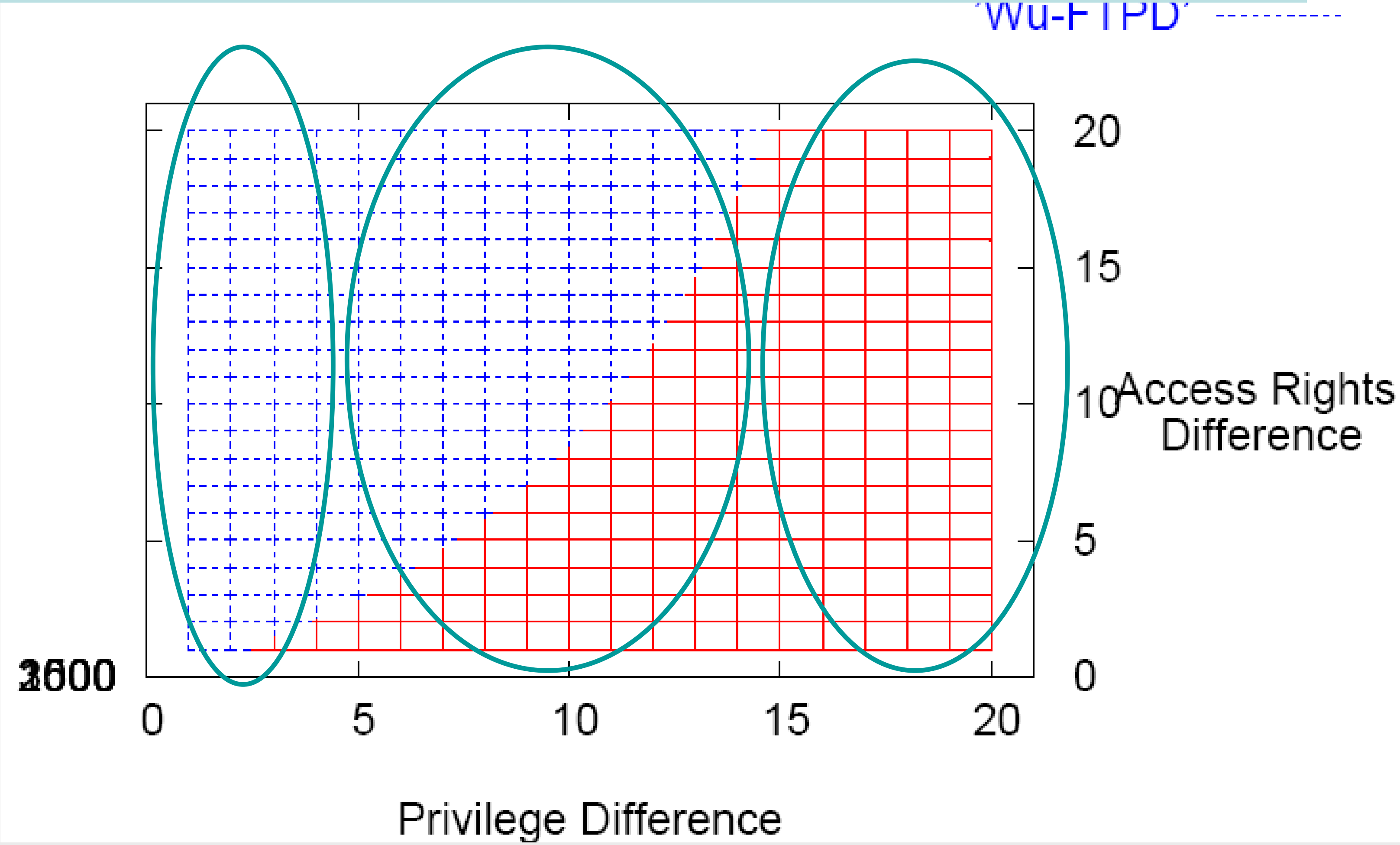
Tool Usage in Software Development



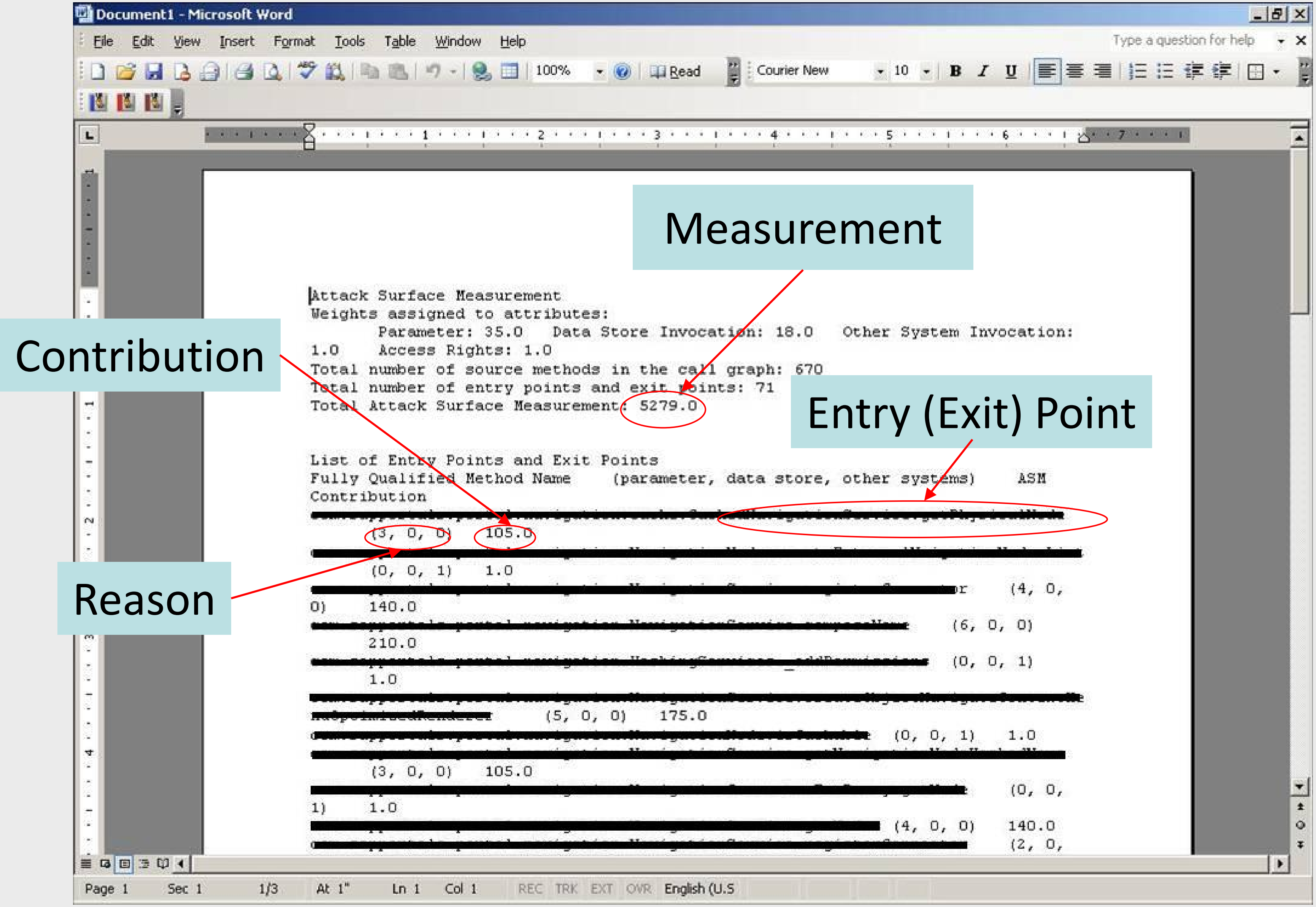
- Tool produces detailed output
  - Guides attack surface reduction
- Incremental analysis
- What-If scenarios
  - Addition of a new feature
  - Removal of a feature

FTP Daemons (method)

1. Access rights don’t matter.
2. proftpd privilege level contributes more than auth.'s.



Tool Output



Measurement

Contribution

Entry (Exit) Point

Reason

```
Attack Surface Measurement
Weights assigned to attributes:
       Parameter: 35.0   Data Store Invocation: 18.0   Other System Invocation:
1.0    Access Rights: 1.0
Total number of source methods in the call graph: 670
Total number of entry points and exit points: 71
Total Attack Surface Measurement: 5279.0

List of Entry Points and Exit Points
Fully Qualified Method Name    (parameter, data store, other systems)    ASM
Contribution
[redacted]
      (3, 0, 0)    105.0
[redacted]
      (0, 0, 1)    1.0
[redacted]                                                     (4, 0,
0)    140.0
[redacted]                                              (6, 0, 0)
      210.0
[redacted]                                              (0, 0, 1)
      1.0
[redacted]
                              (5, 0, 0)    175.0
[redacted]                                         (0, 0, 1)    1.0
[redacted]
      (3, 0, 0)    105.0
[redacted]                                                     (0, 0,
1)    1.0
[redacted]                                         (4, 0, 0)    140.0
[redacted]                                                     (2, 0,
```

References- 1

[ACSAC 01] Rayford B. Vaughn, Ronda R. Henning, and Ambareen Siraj. Information assurance measures and metrics - state of practice and proposed taxonomy. In Proc. of Hawaii International Conference on System Sciences, 2003.

[CRA 03] Computing Research Association (CRA). Four grand challenges in trustworthy computing. http://www.cra.org/reports/trustworthy. computing.pdf, November 2003.

[DIMACS 03] Gary McGraw. From the ground up: The DIMACS software security workshop IEEE Security and Privacy, 1(2):59–66, 2003.

[CSTB 07] Seymour E. Goodman and Herbert S. Lin, editors. Toward a Safer and More Secure Cyberspace. The National Academics Press, 2007.

[HPW03] M. Howard, J. Pincus, and J.M. Wing. Measuring relative attack surfaces. In Proc. of Workshop on Advanced Developments in Software and Systems Security, 2003.

References- 2

[LT89] N. Lynch and M. Tuttle. An introduction to input/output automata. CWI-Quarterly, 2(3):219–246, September 1989.

[KPF97] Barbara Kitchenham, Shari Lawrence Pfleeger, and Norman Fenton. Towards a framework for software measurement validation. IEEE Trans. Softw. Eng.,21(12):929–944, 1995

[LT07] M. Y. Liu and I. Traore. Properties for security measures of software products. Applied Mathematics and Information Science (AMIS) Journal, 1(2):129–156, May 2007.

[K87] Chris F Kemerer. An empirical validation of software cost estimation models. Commun.ACM, 30(5):416–429, 1987.

[L32] R. Likert. A technique for the measurement of attitudes. Archives of Psychology, 22(140):5–55, June 1932.

References- 3

[AB95] J. Alves-Foss and S. Barbosa. Assessing computer security vulnerability. ACM SIGOPS Operating Systems Review, 29(3):3–13, 1995.

[VGMCM96] J. Voas, A. Ghosh, G. McGraw, F. Charron, and K. Miller. Defining an adaptive software security metric from a dynamic software failure tolerance measure. In Proc. of Annual Conference on Computer Assurance, 1996.

[ODM99] R. Ortalo, Y. Deswarte, and M. Kaˆaniche. Experimenting with quantitative evaluation tools for monitoring operational security. IEEE Transactions on Software Engineering, 25(5):633–650, 1999.

[S99] Bruce Schneier. Attack trees: Modeling security threats. Dr. Dobb’s Journal, 1999.

[MBFB05] Miles A. McQueen, Wayne F. Boyer, Mark A. Flynn, and George A. Beitel. Timeto-compromise model for cyber risk reduction estimation. In ACM CCS Workshopon Quality of Protection, September 2005.

References- 4

[LB08] David John Leversage and Eric James Byres. Estimating a system’s mean time-tocompromise. IEEE Security and Privacy, 6(1):52–60, 2008.

[MGVT02] Bharat B. Madan, Katerina Goseva-Popstojanova, Kalyanaraman Vaidyanathan, and Kishor S. Trivedi. Modeling and quantification of security attributes of software systems. In DSN, pages 505–514, 2002.

[S04] Stuart Edward Schechter. Computer Security Strength & Risk: A Quantitative Approach. PhD thesis, Harvard University, 2004.