



The “Cracker World” is Connected

Marcellus Buchheit
President & CEO,
WIBU-SYSTEMS USA Inc

mabu@wibu.com

Overview

- **Who** are the **pirates** of the 21st century?
- **How** do they **work**?
- Which **tools** do they **use**?
- How can we **protect** “**us**” against “them”?
- Questions & Answers

Traditional Pirates (16th/17th century)



Traditional Pirates (16th/17th century)

- The **victims**:

- Hard-working **farmers**, craftsmen or artists
- Hard-working **ship owners** and the ship **crew**
- Ill-equipped **soldiers** on freight ships

Prepared for the “**market war**” but
not for real **physical** attacks

- The **attackers**:

- Flexible, independent, **small** groups
- Their power was the **physical** attack
- **Used** the loot by themselves or **sold** it for a lower price

Sometimes got **rich** with **not** much work

The dilemma of 16th/17th century

The **risk** that **pirates** will **find** a trading ship in open, unprotected water was **assumed** as **low**:

- **Protection** of a war ship was **decided** as too **expensive**
- **Ship owners** **hoped** that their ships would be **safe**
- **Access** to powerful **weapons** (canon etc) and qualified **security** staff was **limited**

Modern Pirates (21st century)



Modern Pirates (21st century)

- The **victims**:

- Hard-working **software designers** and programmers
- Hard-working software **sales** persons
- **Users** of cracked software

Prepared for the “**market war**” but
not for real “**exploiting software**” attacks

- The **attackers**:

- Flexible, independent, **small** groups
- Their power is the **knowledge** (and their cracking tools)
- **Used** the loot by themselves or **sold** it for a lower price

Sometimes get **rich** with **not** much work

Modern pirates: How they work

Professional software of today is:

- **License Management**

- User cannot use **unlicensed** software
- User cannot use **more** licenses than he/she actually **bought**.

- **Sophisticated programming**

- “Tricky” **algorithms** as core knowledge of the SW company
- Usually **not patented** (to difficult, not always possible)
- Source code is **top secret** or at least **confidential**

- **Security modules**

- Different **user rights**, usually managed by **passwords**
- **Encrypted** user **data** (text, picture, generic files)

Modern pirates: How they work (II)

Pirates attack the **core** of software:

- **License management**
 - Crack the **license control** mechanisms
 - Use/resell software, **increase** number of users
- **Sophisticated programming**
 - **Disassemble** interesting parts of software
 - **Analyze** implementation of secret algorithms
 - Use/resell **knowledge** to competitors or publish it
- **Security modules**
 - Crack the **security** mechanism
 - Use/resell **secret** user **data** or publish it

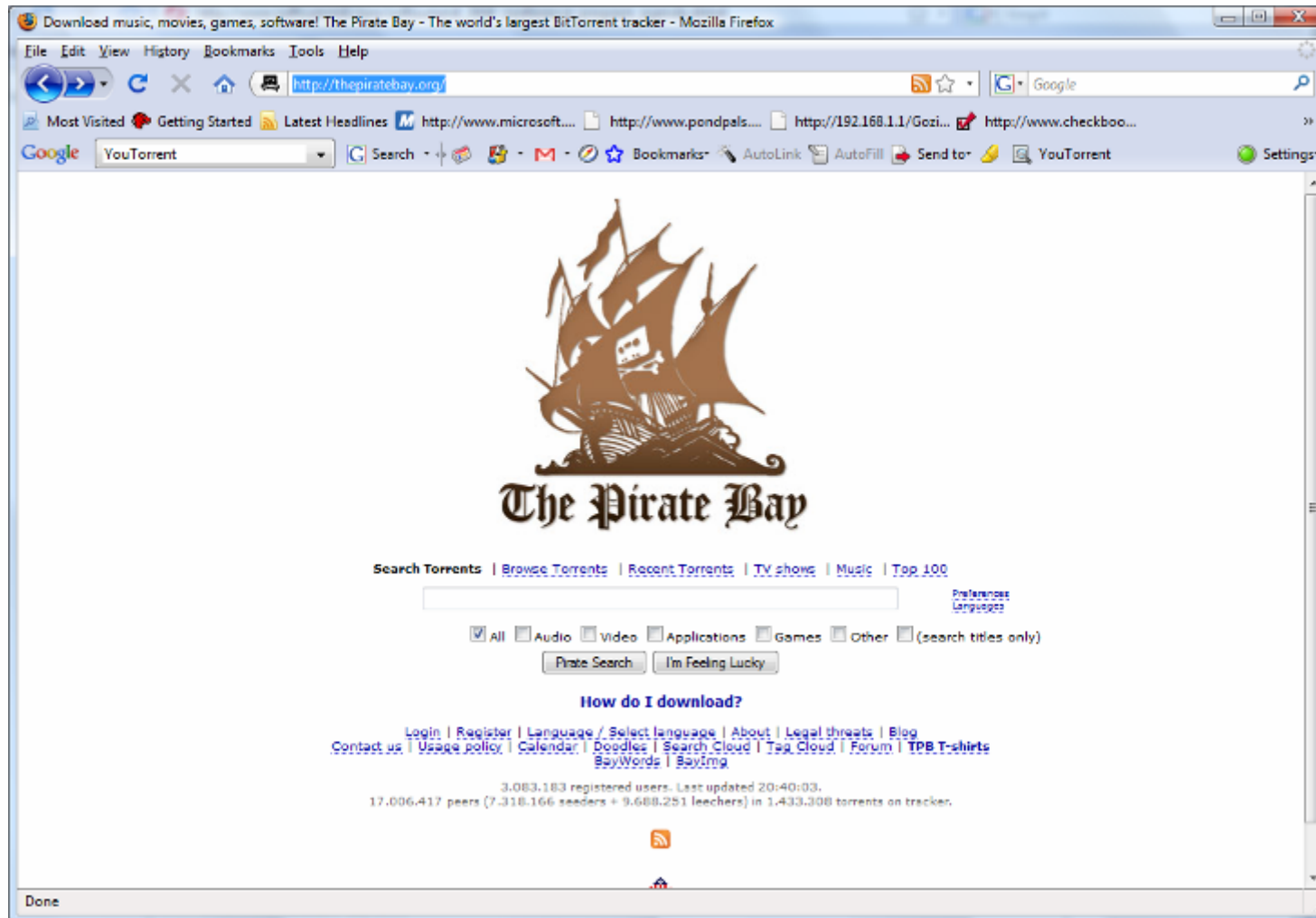
The dilemma of the 21st century

The **risk** that a software product is a target of pirates is **assumed** as **low**:

- Software has to be **finished** in high **time-pressure**
- Designer/programmer do **not know** how **cracking** works
- Access to powerful **protection tools** is **difficult**
- **Knowledge** about how to **protect** against cracking is **low**

The dilemma of the 21st century (II)

www.ThePirateBay.org



The dilemma of the 21st century (III)

“A Starbucks coffee shop with a wireless link may present an incredibly comfortable place from which to launch attacks [...]

You are fairly safe if you're drinking coffee in a Houston Starbucks while launching an attack from New Dehli over the border of China ...”

From “Exploiting Software – How to Break Code”
by Greg Hoglund and Gary McGraw, Addison Wesley

Modern Pirates: Open Teams

Exploiting software can be **difficult**:

- Every **step** needs deep **knowledge**
 - Knowledge is **complex** and changes **quickly**
- Top **crackers** are usually highly **specialized**
 - Someone has the **knowledge** only (→ researcher)
 - Someone **plans** the suitable **tools** (→ designer)
 - Someone **implements** the **tools** (→ programmer)
 - Someone **adapts** the **tools** for **real** use (→ expert)
 - Someone **uses** the **tools** (→ cracker)
 - Someone **sells** the cracked software (→ dealer)

Modern Pirates: Open Teams (II)

Internet connects the **open** teams:

- Do **not** expect a proper **website**
- Outside of **Google** or other search engine
- Usually **IP addresses** only
 - **Easy** to switch, **difficult** to follow by **legal** investigation
 - Communication usually **encrypted**
- **Entry** points via **newsgroups**
 - “Experts” invites other “experts” dependent on **knowledge**
 - Knowledge is good **protection** against **investigators**

Which tools are they using (I)?

- **Disassembler**
 - **IDA Pro** – a commercial product
- **Debugger**
 - **Olly Debug** – freeware
- Finding “**code of interest**” (Crypto, Packing etc.)
 - **PeID** – freeware
- Restruct **modified PE** files
 - **ImpRec** – freeware

Which tools are they using? (II)

Microsoft **.NET**:

- Powerful programming model
- Powerful “cracking model”

My Favorite Tools:

- Red Gate’s **.Net Reflector** (Original Lutz Roeder)
- Microsoft Disassembler **ILDASM**
- Microsoft Assembler **ILASM**

Which tools are they using? (III)

WireShark:

- Universal network end-point and network communication tool
 - <http://www.wireshark.org/>
- Analyzes all traffic via TCP, UDP, IP etc.

Which tools are they using (IV)?

- **SysInternals from Microsoft**

- www.sysinternals.com (Mark Russinovich)



Which tools are they using? (V)

Microsoft **SysInternal Tools**:

- **RegMon**

“This monitoring tool lets you **see** all **Registry** activity in real-time.”

- **FileMon**

“This monitoring tool lets you **see** all **file system** activity in real-time.”

- **DiskMon**

“This utility **captures** all hard **disk** activity [...]”

How tools are used?

Example: “License Setting via Machine Binding”

- Management **Overhead** for **Licensors**
 - Key Generator, Key Provider Website, Coding
- **Annoying** for **Licensee**
 - Change of Machine or parts of it needs **re-licensing**
- But: More or less **easy** to **crack**:
 - Machine **identification** is analyzed by **system-call tracing tools** (and these calls will then be faked)
 - **Key Storage** is file, disk or registry (what else?):
 - Microsoft’s **RegMon**, **DiskMon** and **RegMon** will tell all changes

How you can protect? (I)

C/C++ world (Assembler-based):

- Use newest **compiler**
 - Usually **better** code against buffer-overflow etc.
- Use highest **optimization** level
 - Especially **inline-code** (“forced inline”) helps a lot
 - High-optimized code is **difficult** to read, even for **experts**

How you can protect? (II)

.Net and Java World:

- Use **obfuscation** tools
 - Some are public-domain, some are “commercial grade”
 - Make it difficult to **disassemble**, **re-source** or **modify** intermediate code (**MSIL**, **Java Byte Code**)
 - **Very few** are really **effective**

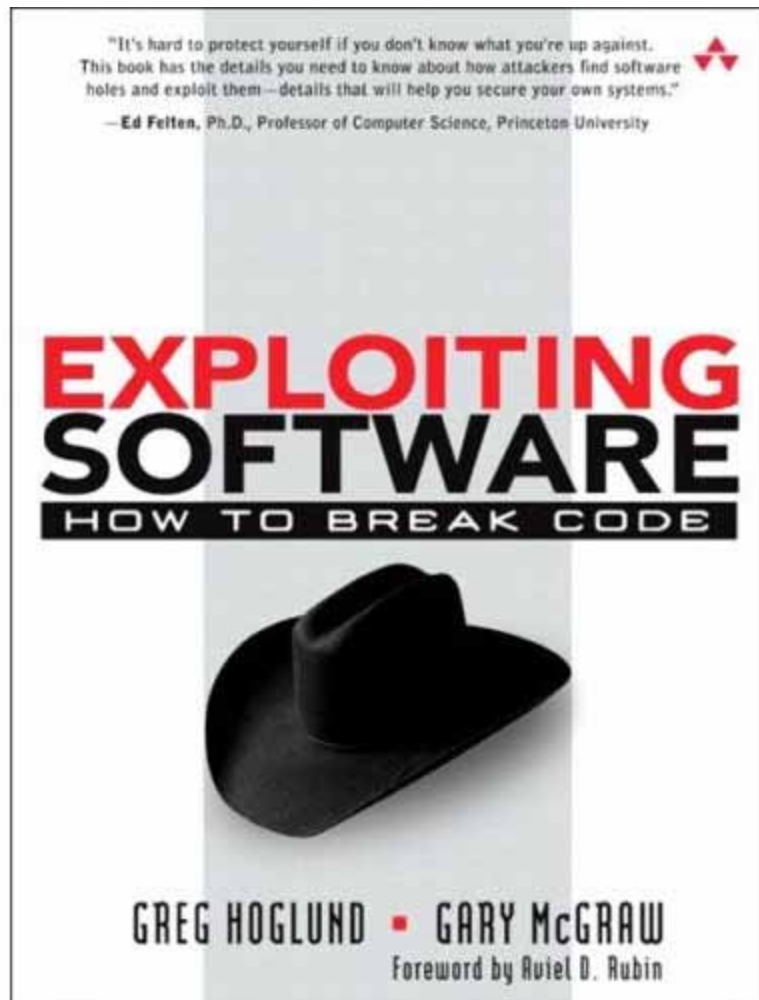
How you can protect? (II)

- Test with free **cracking** tools
 - See my **List of Favorites**
- Develop own Protection Technologies
 - Experience, Experience, Experience

The **Bottom Line**:

- Most developers **do not have** the time or knowledge to **effectively avoid exploitation** of their code

Further References (I)



Exploiting Software –

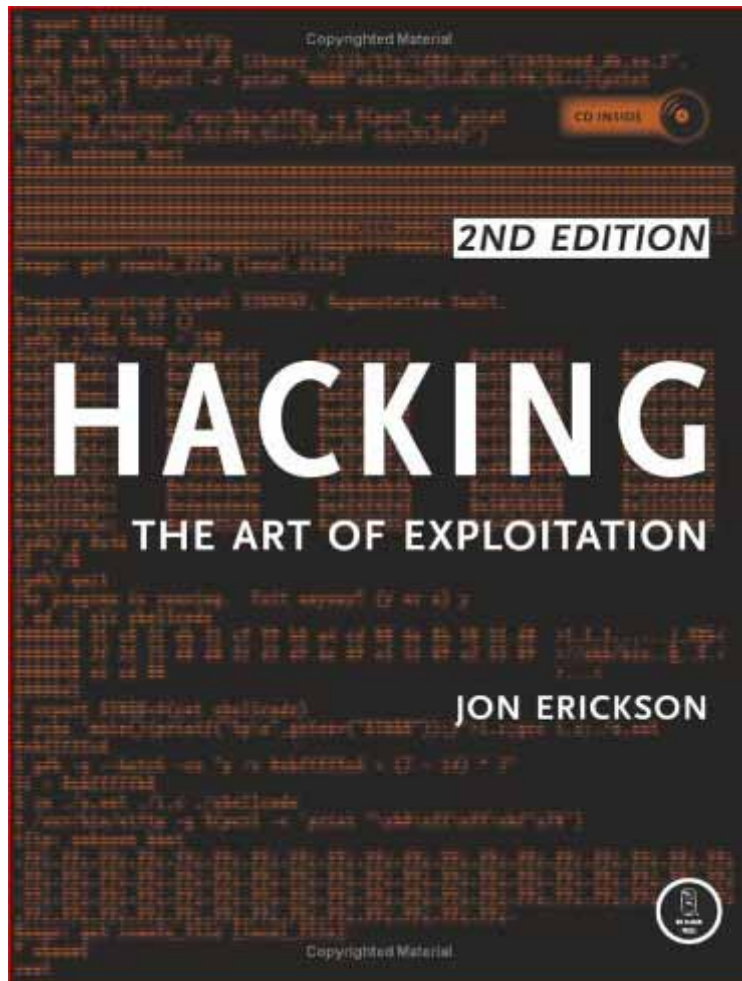
How to Break Code

Greg Hoggund &
Gary McCraw

Addison Wesley, 2004

Amazon Review: 4.5 stars
(5: 17, 4: 8, 3: 2)

Further References (II)



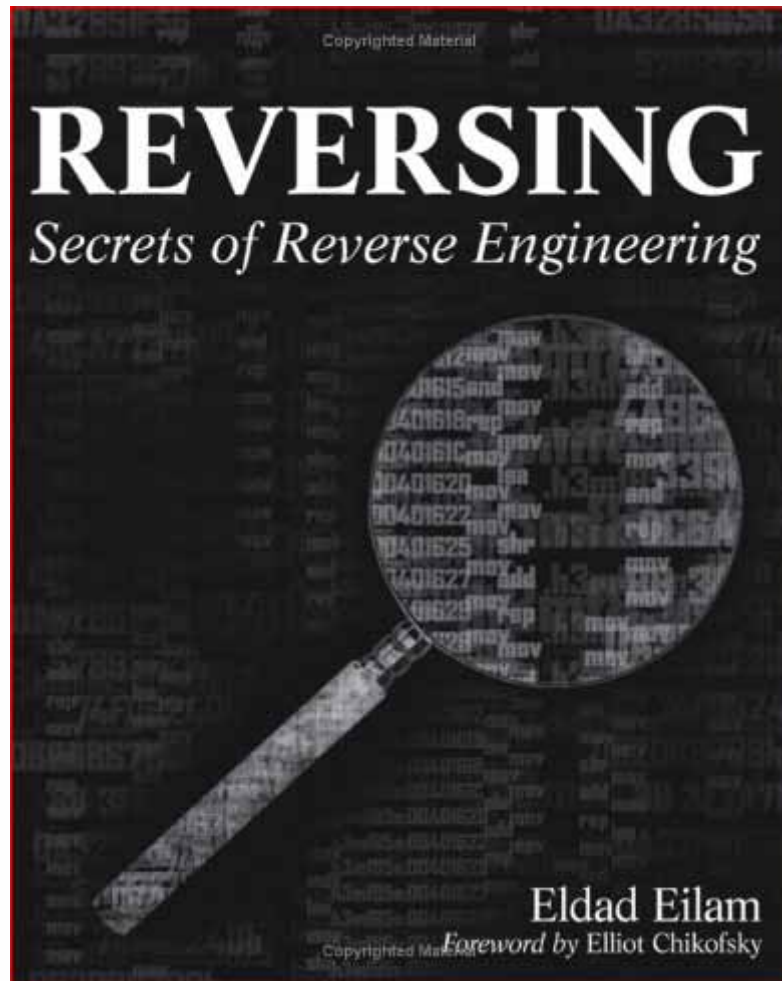
“Reversing: Secrets of Reverse Engineering”

Jon Erickson

No Starch Press, 2008

Amazon Review: 4.5 stars
(5: 29, 4: 14, 3: 8)

Further References (III)



“Reversing: Secrets of Reverse Engineering”

Eldad Eilam

Wiley, 2005

Amazon Review: 4.5 stars
(5: 11, 4: 3, 3: 1)

Further References (IV)

- “The Shellcoder’s Handbook”: Discovering and Exploiting Security Holes” –
Jack Koziol, David Litchfield etc, 2006
- “Hacker Disassembling Uncovered” –
Kris Kaspersky, 2007

Questions and Answers

