Agentúra
Ministerstva školstva, vedy, výskumu a športu SR
pre štrukturálne fondy EÚ

Európska únia
Európsky sociálny fond

UNIVERZITA KOMENSKÉHO V BRATISLAVE
FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY

PRÍPRAVA ŠTÚDIA MATEMATIKY A INFORMATIKY NA FMFI UK V ANGLICKOM JAZYKU

ITMS: 26140230008

DOPYTOVO – ORIENTOVANÝ PROJEKT

Moderné vzdelávanie pre vedomostnú spoločnosť/Projekt je spolufinancovaný zo zdrojov EÚ

Security incidents, weaknesses and vulnerabilities

Martin Stanek

Department of Computer Science
Comenius University
stanek@dcs.fmph.uniba.sk

Security of IT infrastructure (2013/14)

Content

General discussion

Examples – security failures

- Randomness of cryptographic keys
- Timing attacks on comparisons
- Adobe password encryption
- PKCS #11 and cryptographic tokens
- WPS (WiFi Protected Setup)
- Encrypted USB drives
- Hash tables collisions

Other incidents

Introduction

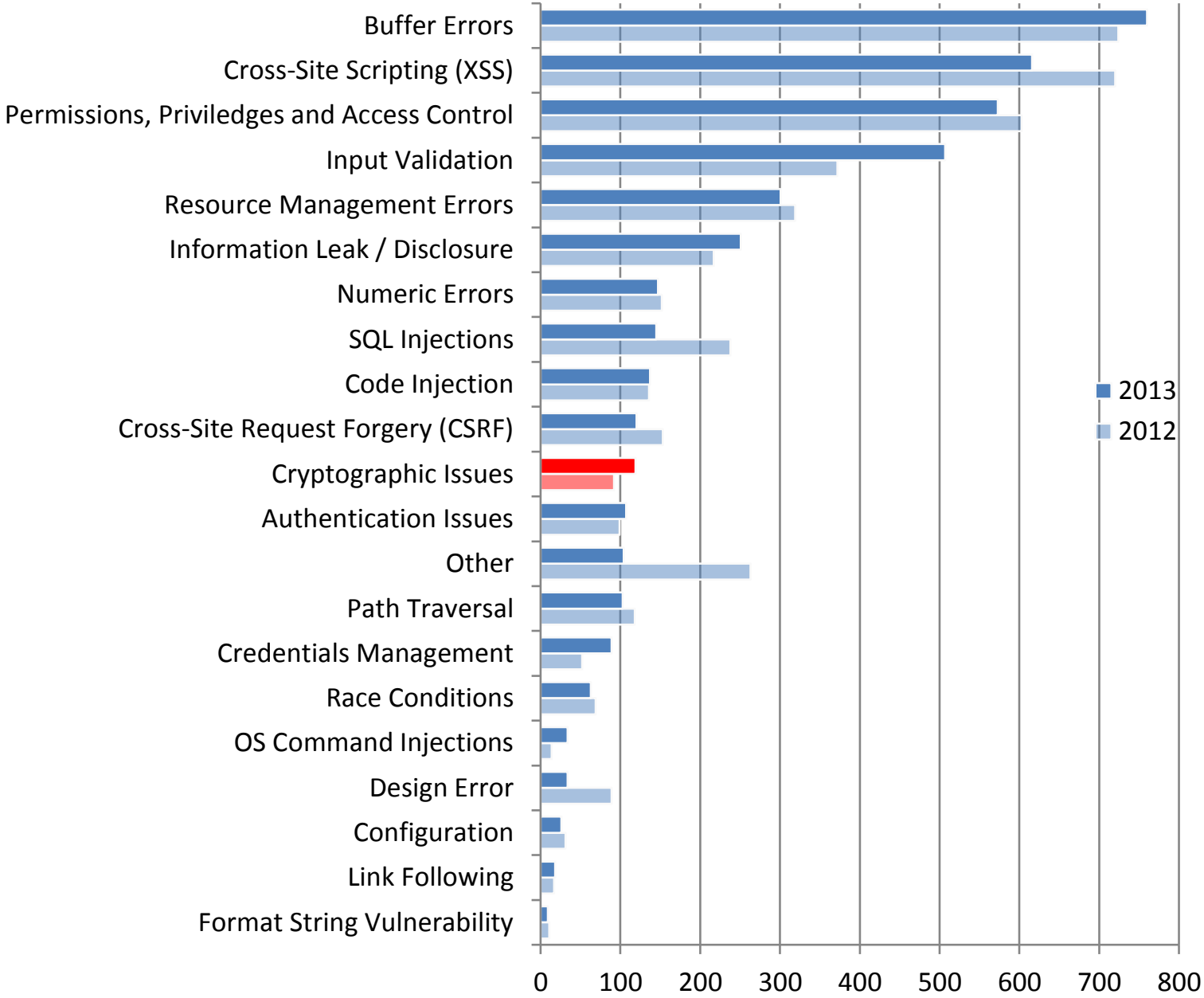
Security incidents and failures

- various causes (or their combination): human factor, criminal activities, technical vulnerabilities etc.
- impact: “nothing” happened, loss of reputation, cost of repair/replacement of data and systems, direct financial loss, bankruptcy etc.

mostly technical failures/vulnerabilities in this lecture

- just examples . . . reality is worse
- National Vulnerability Database (nvd.nist.gov)
- vulnerabilities published: 5289 (2012), 5186 (2013)
- classification (categories, severity etc.)
- “Insufficient information” not shown on the following chart

NVD – vulnerabilities published in 2012 and 2013



Other classifications of vulnerabilities

- Common Weaknesses Enumeration (cwe.mitre.org)
  - Common Attack Pattern Enumeration and Classification (capec.mitre.org)
- Open Web Application Security Project (OWASP, www.owasp.org)
  - primarily for web applications – vulnerabilities, attacks, risks
  - Testing Guide
- more detailed classifications, description, examples, additional information

Real world – surveys and analyses

[1] IBM X-Force 2013 Mid-Year Trend and Risk Report (September 2013)
[2] EY's Global Information Security Survey 2013 (October 2013)
[3] Websense Security Predictions for 2014 (2013)
[4] DataLossDB Statistics (datalossdb.org)
[5] Verizon 2013 Data Breach Investigations Report (2013)

...etc.

Data breaches

- 2013 – the most frequent types of data breaches [4]:
  1. Hack – Computer-based intrusion, data may or may not be publicly exposed (48%)
  2. Fraud or scam (usually insider-related), social engineering (8%)
  3. Stolen Laptop (generally specified as a laptop in media reports) (7%)
  4. Web – Data typically available to the general public via search engines, public pages, etc. (6%)
  5. Unknown or unreported breach type (4%)
  6. Discovery of documents not disposed properly (4%)
  7. Email communication exposed to unintended third party (4%)

  ...other causes

Data breaches – examples (1)

1. University of Veterinary Medicine and Pharmacy in Košice
   - January 2014
   - leaked personal data of 1500 students
   - addresses, ID card numbers, personal identification numbers, ...
   - published PDF in Central register of contracts
   - "blacked out" personal data ...still there, can be copied (selected)
2. Korea Credit Bureau, South Korea
   - February–December 2013
   - more then 100 million accounts and credit cards information
   - South Korea: more than 4 credit cards per person in average
   - 40% population affected
   - names, home addresses, phone numbers, bank account numbers, credit card details, identification numbers, income, passport numbers, ...
   - contract technician copied the data to portable hard drive (and sold)

Data breaches – examples (2)

3. Target, USA
   - December 2013
   - 40 millions credit and debit cards information
     + additional 70 millions personal information
   - card information
     + names, mailing addresses, phone numbers, email addresses
   - malware on POS devices
4. Orange, France
   - January 2014
   - 800 000 customers' data
   - names, mailing addresses, e-mails, phone numbers, customer account IDs (masked)
   - hack (details unknown)

Top priorities [2]

- global survey, more than 1900 organizations
- top priorities in 2014:
  1. Business continuity/disaster recovery
  2. Cyber risks/cyber threats
  3. Data leakage/data loss prevention
  4. Information security transformation (fundamental redesign)
  5. Compliance monitoring

Security failures/vulnerabilities ...

examples

Randomness of cryptographic keys

- 2008 – Debian
- modification of openssl source code
  - the use of uninitialized memory
  - broken initialization of pseudorandom generator . . . initialized by PID only
  - at most 98301 unique initialization values overall (depending on particular platform)
- impact:
  - predictable keys for SSH, OpenVPN, DNSSEC, X.509 certificates, session keys in SSL/TLS, . . .
  - using library just for a single DSA signing . . . compromised private key
  - similar problem with randomness in Sony Playstation 3 (ECDSA signatures, 2010)

Later . . . in openssl 1.0 source code

```
/* DO NOT REMOVE THE FOLLOWING CALL TO MD_Update()! */
MD_Update(&m,buf,j);
/* We know that line may cause programs such as
   purify and valgrind to complain about use of
   uninitialized data.  The problem is not, it's
   with the caller.  Removing that line will make
   sure you get really bad randomness and thereby
   other problems such as very insecure keys. */
```

- Correct and secure implementation of cryptography is not easy
  - 7 vulnerabilities in openssl (NVD, published in 2010–2013) with severity High

Timing attacks on comparisons (Google, Sun, …)

- 2009 – Keyczar (Google), Java (Sun), …
- common scenario: server compares received HMAC with calculated one
- attacker's goal: to get correct HMAC for his own message ("authentic")
- What is wrong with this code (Python)?

```
return self.Sign(msg) == sig_bytes
```

What is wrong with this code (Java)?

```
public static boolean
isEqual(byte digesta[], byte digestb[]) {
  if (digesta.length != digestb.length)
    return false;

  for (int i = 0; i < digesta.length; i++) {
    if (digesta[i] != digestb[i])
      return false;
  }
  return true;
}
```

HMAC reconstruction

How long does it take for server to answer/react to incorrect HMAC

- if the first 0, 1, 8 or 15 bytes are correct?
- HMAC reconstruction based on time-variance of responses
- 4th byte:

```
71 A0 89 00 00 . . . 00
71 A0 89 01 00 . . . 00
         . . .
71 A0 89 4A 00 . . . 00      longer time to process?
         . . .
71 A0 89 FF 00 . . . 00
```

- usually multiple measures required for a single value (noise)
- statistical evaluation of measurements

Constant-time comparison (Java)

```
public static boolean
isEqual(byte[] digesta, byte[] digestb) {
  if (digesta.length != digestb.length)
    return false;

  int result = 0;
  for (int i = 0; i < digesta.length; i++) {
    result |= digesta[i] ^ digestb[i];
  }
  return result == 0;
}
```

Adobe password encryption

- 2013, Adobe
- data breach, 38 million *active* users account information exposed
- 150 million user accounts overall
- passwords are encrypted (the key was not leaked)
  …using 3DES (block cipher with 8 B block) in ECB mode
- result:
  - equal password substring [1-8], [9-16] easily identifiable
  - guess using password hits (part of account information), e.g.
    “numbers 123456”, “c’est 123456”
    “1*6”, “sixones”
    “q w e r t y”, “6 long qwert”

The most frequent passwords from Adobe's database

1.	123456	(≈ 1,9 million)
2.	123456789	(≈ 446 thousand)
3.	password	(≈ 345 thousand)
4.	adobe123	(≈ 211 thousand)
5.	12345678	
6.	qwerty	
7.	1234567	
8.	111111	
9.	photoshop	
10.	123123	
11.	1234567890	
12.	000000	
13.	abc123	
14.	1234	
15.	adobe1	
16.	macromedia	
17.	azerty	
18.	iloveyou	
19.	aaaaaa	
20.	654321	

The most frequent passwords

source: Splashdata 2013, based on leaked passwords, comparison with 2012

1.	123456	(+ 1)
2.	password	(− 1)
3.	12345678	
4.	qwerty	(+ 1)
5.	abc123	(− 1)
6.	123456789	(new)
7.	111111	(+ 2)
8.	1234567	(+ 5)
9.	iloveyou	(+ 2)
10.	adobe123	(new)
11.	123123	(+ 5)
12.	admin	(new)
13.	1234567890	(new)
14.	letmein	(− 7)
15.	photoshop	(new)
16.	1234	(new)
17.	monkey	(− 11)
18.	shadow	
19.	sunshine	(− 5)
20.	12345	(new)
21.	password1	(+ 4)
22.	princess	(new)
23.	azerty	(new)
24.	trustno1	(− 12)
25.	000000	(new)

PKCS #11 and cryptographic tokens

- PKCS #11 – Cryptographic Token Interface Standard
  - defines interface for using tokens
  - tokens: HSM, smart cards (various types), software tokens
  - generating keys, encryption and decryption, signing and verification of digital signatures, …
- private and secret keys can have additional “security” attributes
  - *sensitive* – cannot be read or exported from token in clear
  - *unextractable* – cannot be read or exported from token at all (even encrypted)
  - *wrap* – can be used to encrypt (wrap) other keys
  - *decrypt* – can be used to decrypt data

  …+ rules on setting and combining attributes
- standard is not explicit about all situations
- implementations differ

Examples of PKCS #11 vulnerabilities

- some token allowed export of sensitive or unextractable key in clear
- wrap/decrypt attack:
  - keys: k_1 (sensitive), k_2 (wrap, decrypt)
    1. wrap ... $\{k_1\}_{k_2}$
    2. decrypt ... k_1
  - variant with importing a new key k_2 into the token (then a single "wrap" suffices)
  - variant with creating two copies of k_2, while the first one is "wrap" a the second one is "decrypt"
- other problems related to change attributes of keys
- 2010, 18 commercially available tokens tested
  ... 10 tokens "broken" (attack found, often multiple attacks)
  e.g. 7 tokes happily exported sensitive or unextractable keys in clear
- for detail, see
  `secgroup.dais.unive.it/projects/security-apis/tookan/`

WPS (WiFi Protected Setup)

- 2011
- goal: easy (and secure) method to add an device to network
- implementation:
  - 8 digit PIN code authentication (printed on a sticker)
  - theoretically 10^8 possibilities
  - practically: response to incorrect PIN leaks an information whether the first half of the PIN is wrong
    last digit is a checksum
  - $10^4 + 10^3$ possibilities
- WPS can't be turned off in some WiFi routers

Encrypted USB drives

- 2010; Kingstone, SanDisk, Verbatim
- FIPS 140-2 Level 2 certification; AES-256 encryption
- reality:
  - encryption key does not depend on user's password
  - USB key unlocks if some expected string (fixed, password- and device-independent) is received

Hash tables collisions

- 2011; Oracle, Microsoft, PHP, Apache Tomcat, ...
- analogous problem found originally in 2003; Perl, Squid
- hash table – data structure for storing (key/data) pairs
  - average complexity $O(n)$ for inserting/deleting/finding n elements
  - worst case complexity $O(n^2)$ for n elements (when keys collide)
- problem: colliding keys can be generated easily
- parameters of HTTP POST requests are parsed into hash table automatically
- DoS attack on web server:
  ~70-100kbits/s $\Rightarrow$ one i7 core busy (2011, PHP)

Hashing for hash tables

- Java 6 (java.lang.String, method public int hashCode())
  - 32-bit arithmetic (int), s_i denotes an i-th character of an $(s_{1,\ldots,n})$:

$$\sum_{i=1}^{n} 31^{n-i} \cdot s_i$$

- PHP 5 (algorithm DJBX33A, 32-bit arithmetic), s_0 is constant 5381

$$\sum_{i=0}^{n} 33^{n-i} \cdot s_i$$

- ASP.NET (algorithm DJBX33X), s_0 is constant 5381

$$\bigoplus_{i=0}^{n} 33^{n-i} \cdot s_i$$

- easy to find large multicollisions

Solutions

- limit the size of POST requests, limit CPU for single request, etc.
- better hash function
  - for example randomized hashing – the function dependent on randomly chosen parameter (when process starts)

Other incidents (1)

- NSA
  - 2013; approx. 1.7 million files
  - Showden (contractor)
  - gradual publication of documents and files, global surveillance programs
    - tools and methods, e.g. see Tailored Access Operations (TAO) catalog
    - identities of cooperating companies and governments
    - identities of ISPs and platforms that NSA has penetrated or attempted to penetrate
    - foreign officials and systems that NSA has targeted
- Associated Press
  - April 2013
  - AP Twitter account hacked:
    Breaking: Two Explosions in the White House and Barack Obama is Injured.
  - 136 billion USD from the S&P's 500 Index in two minutes

Other incidents (2)

- Network Time Protocol – DoS attacks
  - NTP amplification attack (amplification factor 19)
  - single 234-byte request . . . 10 packets response (total 4 460 bytes).
  - MONLIST command (IP addresses of the last 600 machines interacting with an NTP server)
  - February 2014 . . . reported DDoS attack with 400 Gbps traffic
- State-sponsored malware
  - high sophistication, multiple modules, targeted
  - Stuxnet – attacking Iranian nuclear program (discovered in 2010)
  - Duqu (similar to Stuxnet) – information capture (e.g. keystrokes), (discovered in 2011)
  - Flame (similar to Stuxnet) – information capture (discovered in 2012)
  - Mask/Careto – information capture (discovered in 2013)

Other incidents (3)

- RSA (1)
  - 2011; targeted attack on RSA (part of EMC)
  - SecurID tokens (one-time passwords, two-factor authentication), market leader
  - replacing 40 million tokens
- NIST, RSA
  - 2013
  - NIST standard includes Dual Elliptic Curve Deterministic Random Bit Generator (Dual EC DRBG)
  - Dual EC DRBG proposed by NSA
  - RSA: Dual EC DRBR default in BSAFE toolkit
  - RSA: 10 million USD deal with NSA (Reuters)
  - possible trapdoor in Dual EC DRBR (strong suggestion)