

## Hacking Web Sites for Fun and Profit



Barry Dorrans

Application Development Consultant

Charteris plc.

[barry.dorrans@charteris.com](mailto:barry.dorrans@charteris.com)

[barryd@idunno.org](mailto:barryd@idunno.org)

CHARTERIS

## Hacking Web Sites for Fun and Profit

- ◆ Parameter Manipulation
- ◆ Cross Site Scripting
- ◆ Character Encoding
- ◆ SQL Injection
- ◆ Leaked Information

## Parameter Manipulation

- ◆ URL / Query Strings
- ◆ Form Variables
- ◆ Unexpected values
- ◆ Cookies
- ◆ HTTP Headers

## URL / Query Strings

- ◆ Easy to view, easy to change
- ◆ URLs should not trigger destructive actions
- ◆ Mitigation Techniques
  - ◆ Don't place important parameters in URLs
  - ◆ If you must encrypt it or provide a hash checksum



CHARTERIS

## Form Variables

- ◆ Slightly harder to view, easy to change
- ◆ URLs should not trigger destructive actions
- ◆ Mitigation Techniques
  - ◆ Do not pass data around in hidden fields
  - ◆ Use session state
  - ◆ If not available encrypt or hash checksum



CHARTERIS

## Unexpected Values

- ◆ Do not rely on client side validation
- ◆ Duplicate validation at all layers



CHARTERIS

## Cookies

---

- ◆ Simple Text Files, easy to edit
- ◆ Cookie: lang=en-gb; ADMIN=no; y=1;  
time=10:30GMT;

## Cookies

- ◆ Simple Text Files, easy to edit
- ◆ Cookie: lang=en-gb; ADMIN=yes; y=1; time=10:30GMT;
- ◆ Mitigation techniques
  - ◆ Validate cookie data
  - ◆ Encrypt cookie contents

CHARTERIS

## HTTP Headers

- ◆ REFERER used by spammers
- ◆ Fake user agents
- ◆ Web sites can act upon language strings, opening SQL injection problems
- ◆ Mitigation
  - ◆ Validate
  - ◆ Don't believe them

CHARTERIS

## Cross Site Scripting

- ◆ < and > are evil
- ◆ Never display raw user input back to the user
- ◆ asp.net input validation only goes so far and limits users too much
- ◆ HTML tags can have events
- ◆ Can be used for
  - ◆ Session hijacking
  - ◆ Cookie sniffing
  - ◆ Browser hijacks



CHARTERIS

## Character encoding

- ◆ The forms authentication hack
- ◆ Change a / in a url to a \
- ◆ Bug in Context.Request.Path
- ◆ Quick fix was URL rewriting of \ to /
- ◆ However
  - ◆ %5c = \
  - ◆ %255c = \
  - ◆ %%%35%63 = \
- ◆ <http://ha.ckers.org/xss.html>

CHARTERIS

## SQL Injection

- ◆ Manipulation of “raw” SQL
- ◆ System Tables
- ◆ Union and FOR XML AUTO
- ◆ SQL login and rights
- ◆ Stored Procedures
- ◆ Parameterised queries

## Manipulation of “raw” SQL

- ◆ Needs
  - ◆ Ability to embed commands using --
  - ◆ Ability to string multiple commands together in a batch
  - ◆ Ability to query meta data from system tables
- ◆ The more powerful the SQL dialect the more susceptible it is to attacks

## Manipulation of “raw” SQL

- ◆ `select count(*) from users where username='barryd' and password='pigeon'`
- ◆ Now add `' or 1=1; --` as the password
- ◆ `select count(*) from users where username='barryd' and "' or 1=1;-- password='pigeon'`



CHARTERIS

## Manipulation of “raw” SQL

- ◆ `select count(*) from users where username='barryd' and password='pigeon'`
- ◆ Now add `' or 1=1; --` as the password
- ◆ `select count(*) from users where username='barryd' and "' or 1=1;-- password='pigeon'`



CHARTERIS

## System Tables

- ◆ `select * from sysobjects`
  - ◆ `where xtype='U'` : user defined tables
  - ◆ `where xtype='P'` : stored procedures
- ◆ `select * from INFORMATION_SCHEMA.tables`  
`where table_name = 'table'`
- ◆ `select TABLE_CATALOG, TABLE_SCHEMA,`  
`TABLE_NAME, COLUMN_NAME, DATA_TYPE,`  
`CHARACTER_MAXIMUM_LENGTH`  
`from information_schema.columns` `where`  
`table_name = 'table'`

CHARTERIS

## The fun part

- ◆ Break the query with bad syntax
- ◆ Use group by to hone into column names
- ◆ Use union to inset information from other tables
- ◆ [http://localhost:2600/hacked/sqlInjection/badQuotes.aspx?id=999;](http://localhost:2600/hacked/sqlInjection/badQuotes.aspx?id=999)  
insert into quotations (quotation,author) values ('Actually I use XP', 'Steve Jobs') --
- ◆ What about xp\_exec?



## Hash your stored passwords

- ◆ One way “encryption”
- ◆ Dictionary attacks (NT SAM)
- ◆ Salting

- ◆ Plain Text

pigeon

- ◆ MD5

b9e6ed3f047270d365d1d4c0d4a118c9

CHARTERIS

## Mitigation

---

- ◆ No more sa / dbo connections
- ◆ No data reader / writer rights
- ◆ Beware of Sql Express user instances
- ◆ Stored procedures or parameterised commands
- ◆ Fail gracefully

## Leaked Information

- ◆ Error Pages
- ◆ Search Engines
- ◆ View State

## Search Engines

- ◆ Hack using [google](#)
  - ◆ ["# -FrontPage-" inurl:service.pwd](#)
  - ◆ [allinurl: admin mdb](#)
  - ◆ <http://johnny.ihackstuff.com/>
- ◆ What about your own web site search?
  - ◆ Specific inclusion is preferable to exclusion



CHARTERIS

## asp.net ViewState

- ◆ View state is not encrypted by default

- ◆ Machine Access Control locked

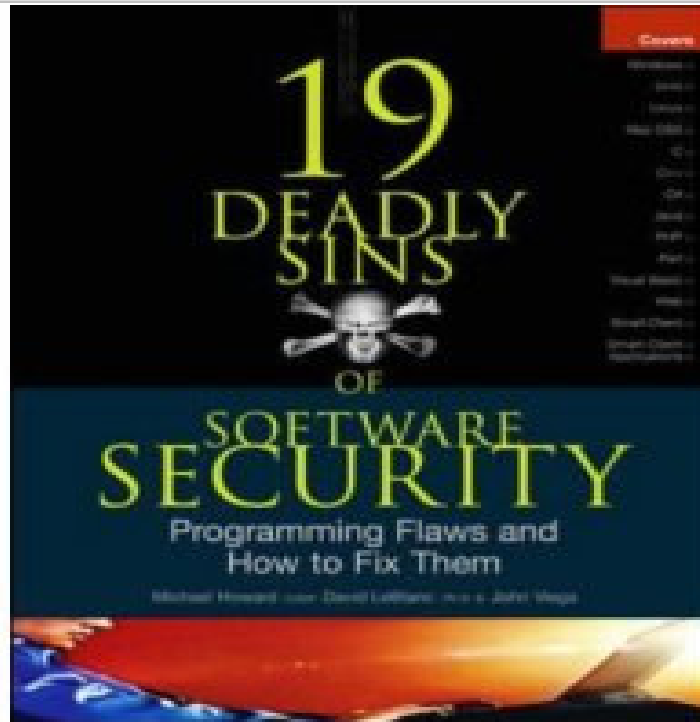
- ◆ `<%@Page EnableViewState='true'  
EnableViewStateMac="true" %>`

```
<system.web>  
  <machineKey  
    decryptionKey="AutoGenerate, IsolateApps  
  " decryption='3DES' ... />  
</system.web/>
```

# The 19 Deadly Sins of Software Security

Michael  
Howard  
David  
LeBlanc  
John Viega

McGraw Hill



CHARTERIS

NOTES

## The 19 Deadly Sins of Software Security

- ◆ Ninety-five percent of software bugs are caused by the same 19 programming flaws.

Amit Yoran,

Former Director of The Department of  
Homeland Security's National Cyber  
Security Division

CHARTERIS

## The 19 Deadly Sins of Software Security

- ◆ Buffer Overflows
- ◆ Format String problems
- ◆ SQL injection
- ◆ Command injection
- ◆ Failure to handle errors

## The 19 Deadly Sins of Software Security

- ◆ Cross-site scripting
- ◆ Failing to protect network traffic
- ◆ Use of "magic" URLs and hidden forms
- ◆ Improper use of SSL
- ◆ Use of weak password-based systems

CHARTERIS

## The 19 Deadly Sins of Software Security

- ◆ Failing to store and protect data
- ◆ Information leakage
- ◆ Improper file access
- ◆ Integer range errors
- ◆ Trusting network address information

## The 19 Deadly Sins of Software Security

- ◆ Signal race conditions
- ◆ Unauthenticated key exchange
- ◆ Failing to use cryptographically strong random numbers
- ◆ Poor usability