SCMS-20

Software for LaGrande Technology: Impact to the Software Development Process

Joseph Cihula

Software Security Architect

Intel Corp.

September 17, 2003

Safer Computing Track – Fall IDF

Tuesday

LT Overview
SCMS-16

TCG & TPM v1.2
SCMS-17

LT Architecture
SCMS-18

Tech Showcase
Every Day
Birds of a Feather Lunches
Tuesday & Wednesday

Wednesday

Privacy Method for Assuring Trust
SCMS-19

Opt-In Strategy
SCMS-156

Trusted Mobile KB Controller
SCMS-24

Software for LT
SCMS-20

Fundamentals of NGSCB
SCMS-21

Migrating Apps to NGSCB
SCMS-22

Thursday

TPM Recovery
SCMS-25

TCG Credentials
SCMS-157

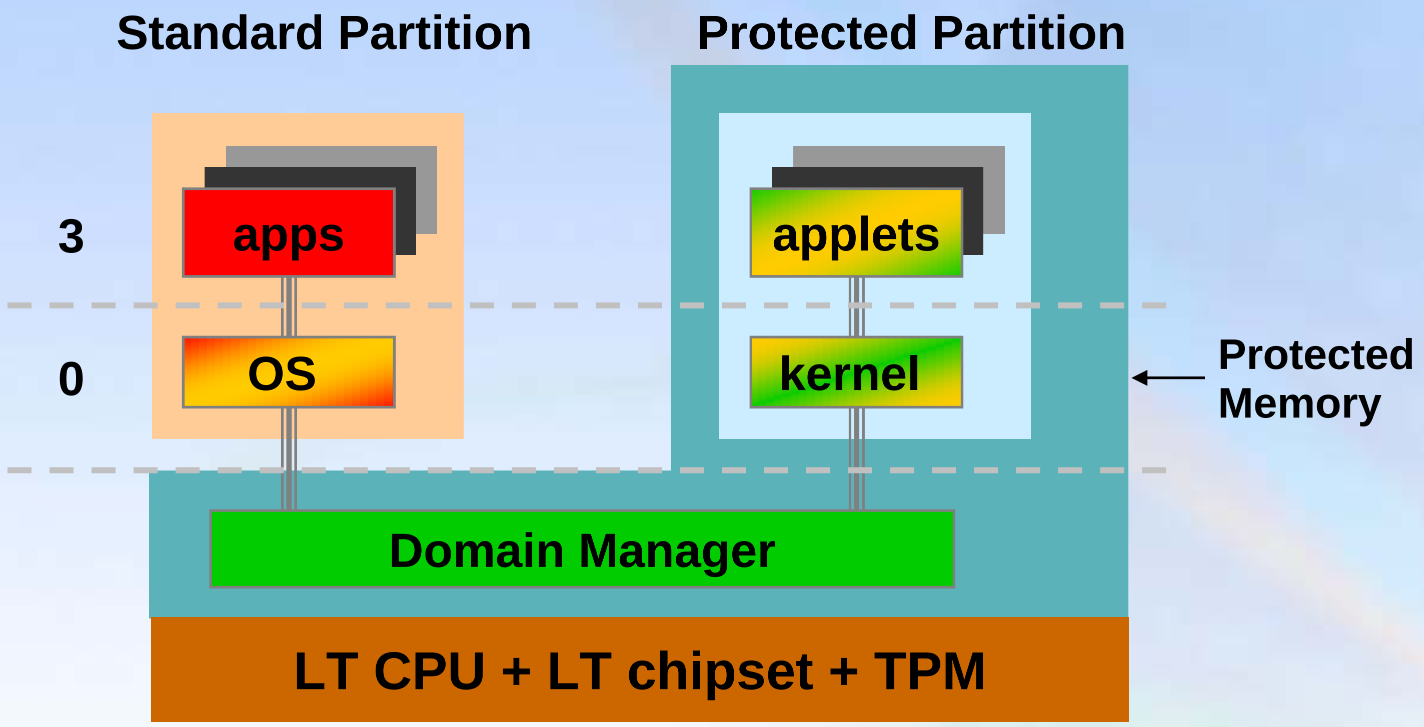
TPM Mfg & Testing
SCMS-180

🟨 = Overview
🟦 = Medium Technical
🟩 = Highly Technical

Outline

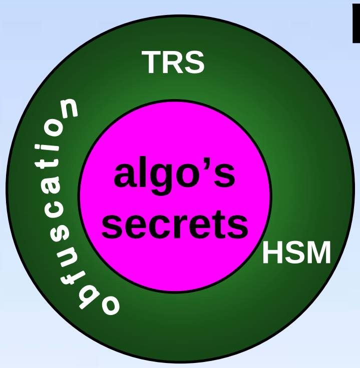
- LaGrande Technology (LT) Overview
- Why Design for LT?
- The LT Software Development Process
  - Security Analysis
  - Design
  - Development
  - Testing
  - Maintenance
- Example: Order Entry

LaGrande Technology Overview



- LT is a general-purpose security foundation
  - LT is application and OS agnostic

Why Design for LT?



- Robust security is easier and more maintainable
  - Today's methods: tamper resistant software (TRS), obfuscation, hardware security modules (HSMs), etc.
    - Requires specialized knowledge, proprietary, very complex, difficult to maintain, expensive
  - LT allows standard, straightforward designs and implementations
    - Standard algorithms, re-use existing code, simple and easy to maintain
    - Hardware protection—not obscurity—provides security

Why Design for LT?

- New security functionality
  - Protected execution (aka domain separation)
    - Program operations and data cannot be observed nor interfered with
  - Protected input and graphics
    - Keystrokes and mouse input are protected from software attacks
    - Displayed information can’t be captured by software
  - Sealed storage
    - Data can be sealed to specific software environment
    - Once sealed, can be persisted anywhere
  - Attestation
    - Remote verifiers can be assured of software and platform they are talking to
      - Protected kernel may extend this to applets ➡ code identity

The LT Software Development Process

- Security Analysis
- Design
- Development
- Testing
- Maintenance

Security Analysis

Security Analysis
Threat List

- High-privilege attacks difficult to prevent today
  - Bypass many OS security mechanisms
  - Very powerful for attacker
  - Difficult for applications to secure against OS compromises
  - E.g. executing code as root/administrator to install a device driver
- LT can maintain security in face of OS compromise
  - Expands threat list to include OS compromises
  - May require new mindset for finding threats

Security Analysis
Mitigation Strategy

- LT enables mitigations not possible before
- Mitigations should be fine-grained and include partial solutions
  - To permit incremental value add over time
- Need to understand entire system design
  - Some LT mitigations may be incomplete
    - Third party code dependencies
    - Functionality not present in protected partition
    - Data needs to be available outside of protected partition
    - Data is available on un-securable systems
  - Moving the attack vector may be valuable
    - E.g. from client to server, etc.

Security Analysis
Solution Prioritization

- Prioritization is about balancing the cost of a mitigation against the risk its threat represents
  - Many risk factors to consider
    - Severity, frequency, business, etc.
  - Many cost factors as well
- LT reduces some mitigation costs
  - Reduces need for costly alternatives (TRS, HSM, etc.)
  - Permit use of common algorithms / existing code
  - Often mitigates multiple threats with single solution
- … but may increase others
  - Effort to write / move code to protected partition
  - May have to re-create or move libraries or infrastructure
  - Potentially multiple code bases for non-LT platforms

Security Analysis
Examples

- Scanning a process' memory for data
  - LT Process sensitive data in protected applet (protected execution)
- Altering stored security policy
  - LT Seal policy before storing (sealed storage)
- Capture user password
  - LT Collect password from protected applet (protected input)
- Capture data by scraping screen
  - LT Display sensitive data from protected applet (protected output)

Design

Software Design for LT

- Move minimum necessary code to protected partition
  - Functionality may be limited
  - Easier to secure and trust less code
  - Easier to develop and maintain
- Avoid redundant UI
  - Will condition the user, defeating purpose of protected graphics
- Don’t over-secure
  - Adds complexity without security value
- Understand global data flows
  - Important for knowing what to protect where
- Separate code and data
  - Don’t hardcode private or shared keys or passwords
    - Code is protected when executing, but not when stored on disk

Managing the Code Base(s)

- Basically, partitioning for LT is just distributed computing
- Easier in managed code
  - Becomes responsibility of managed runtime provider to support protected partition
  - Same basic interfaces, so mostly transparent where code is running
- Otherwise can use abstraction layer

Development

Software Development for LT

- Limited in-the-field debugging and performance tuning
  - Will depend on protected kernel
  - Likely that ‘release’ protected kernels:
    - Will not be debuggable from standard partition
    - Disable event-based monitoring, debug registers, etc.
      - Time-based sampling is still supported
- Protect data sent to un-protected I/O devices
  - Only keyboard, mouse, graphics have hardware protection
  - Protected kernel could protect additional devices
  - Protect data before it leaves protected partition

Software Development for LT

- Make security-related configuration part of code identity
  - E.g. trace level, data sealing, backward compatibility, etc.
- Code reviews are important for security
  - Need to be conscious of data movement between partitions
  - Also check for common security mistakes
    - Buffer overrun, array indexing, canonicalization, access control, least privilege, etc.
  - Security vulnerabilities in protected applet can compromise protected data and operations

Testing

Testing LT Software

- Security testing may require special expertise
- Workload generation for sealed data
  - Also applies to developer unit testing
  - May want to turn off sealing for intermediate builds
  - Otherwise need to re-generate or migrate for each new build/patch
- Validating on legacy platforms and OSes is not an issue
  - Since LT functionality isn't supported by legacy
- LT does not require software to be certified
  - LT does not make value judgments about software in the protected partition

Maintenance

LT Software Maintenance

- Upgradeability must be built in
  - Needed to migrate existing sealed data
  - Exact process depends on protected kernel support
- Changes impact attestation verifiers
  - Publish new code identity
  - Need to update any verifiers

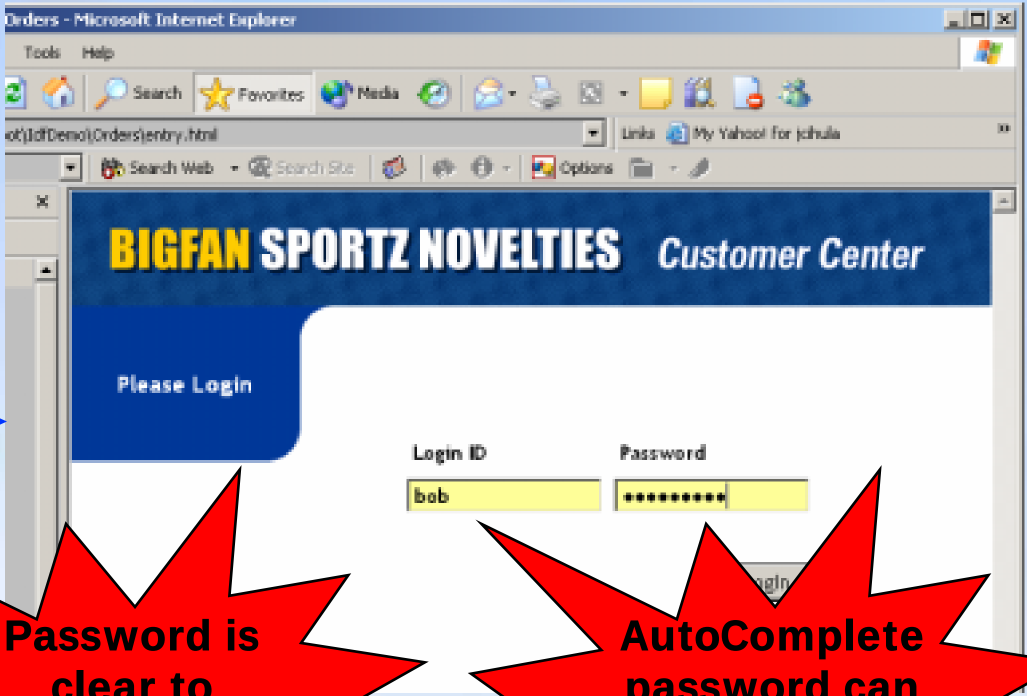
Example:

Order Entry

Security Analysis
Threat List

Paul

User logs into Account


Orders - Microsoft Internet Explorer
Tools Help
Search Favorites Media
ot\IdfDemo\Orders\entry.html
Links My Yahoo! for jchula
Search Web Search Site Options

BIGFAN SPORTZ NOVELTIES Customer Center

Please Login

Login ID: bob

Password: *********

Login

Password can be captured by keystroke sniffer

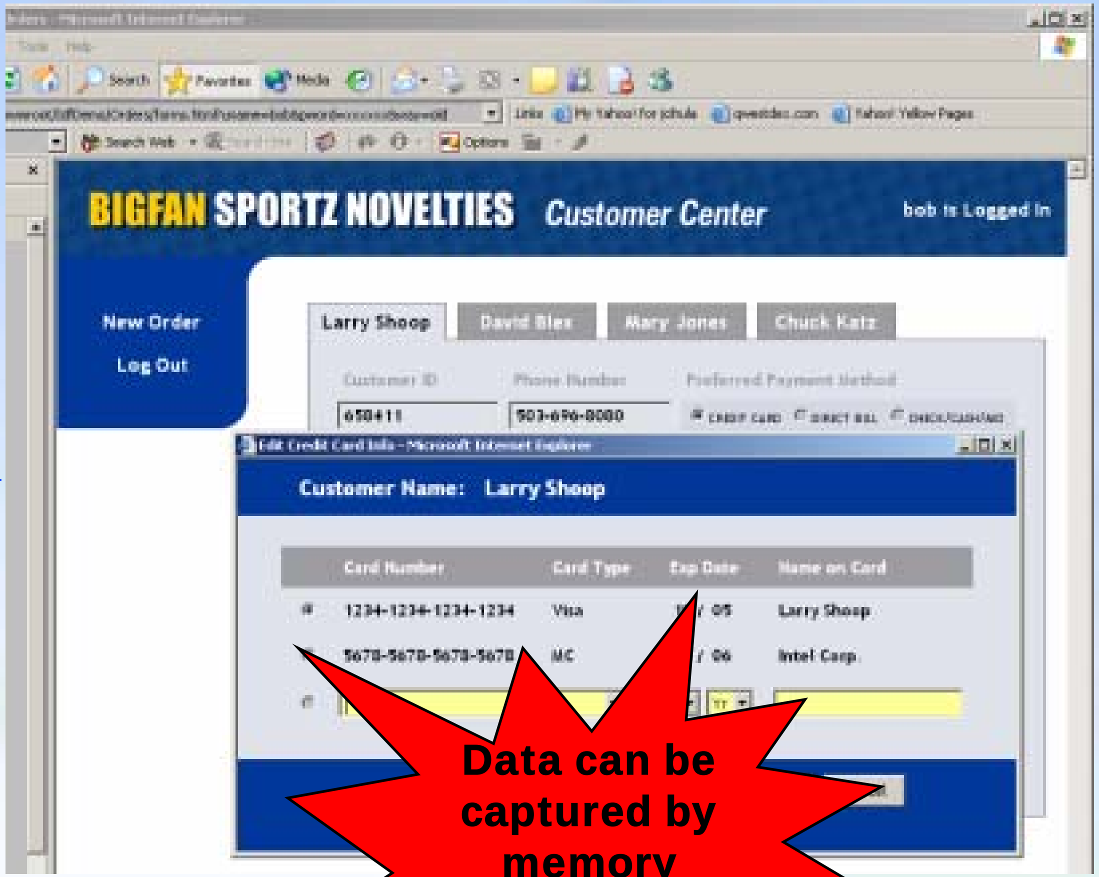
Password is clear to browser and script

AutoComplete password can be copied from HD

Security Analysis
Threat List

Paul

User places order


BIGFAN SPORTZ NOVELTIES Customer Center

bob is Logged In

New Order
Log Out

Larry Shoop | David Bles | Mary Jones | Chuck Katz

Customer ID	Phone Number	Preferred Payment Method
650411	503-696-8080	CREDIT CARD / DIRECT BILL / CHECK/CASH/MO

Edit Credit Card Info - Microsoft Internet Explorer

Customer Name: Larry Shoop

	Card Number	Card Type	Exp Date	Name on Card
●	1234-1234-1234-1234	Visa	[illegible] / 05	Larry Shoop
○	5678-5678-5678-5678	MC	[illegible] / 06	Intel Corp

Data can be captured by screen scraper

Data can be captured by memory scanner

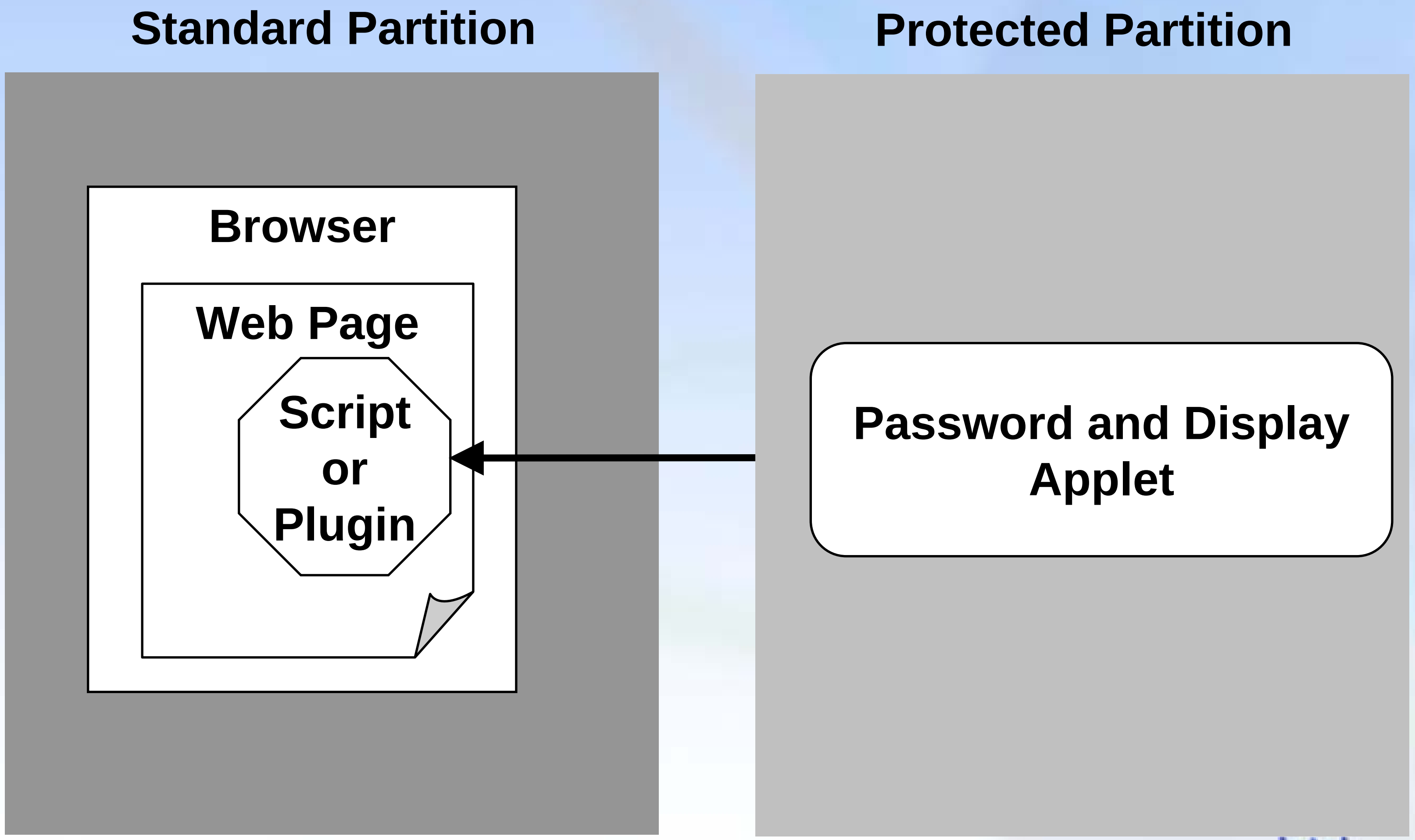
Security Analysis
Mitigation Strategy

1. Capture password with keyboard sniffer
   - ✓ Use multi-factor or non-password authentication
   - ✓ Collect password from protected applet
2. Password is extracted from browser memory
   - ~ Ensure browser is secure from attacks
   - ~ Use non-password authentication
   - ✓ Collect password from protected applet
3. AutoComplete saves password on harddrive
   - ✓ Disable AutoComplete
4. Data can be captured by screenscraper
   - ✓ Display data from protected applet
5. Data is extracted from browser memory
   - ~ Ensure browser is secure from attacks
   - ~ Use separate application for display
   - ✓ Display data from protected applet

Security Analysis
Solution Prioritization

1. Capture password with keyboard sniffer
   - ~ Use multi-factor or non-password authentication
   - ✓ Collect password from protected applet
2. Password is extracted from browser memory
   - ~ Ensure browser is secure from attacks
   - ~ Use non-password authentication
   - ✓ Collect password from protected applet
3. AutoComplete saves password on harddrive
   - ✓ Disable AutoComplete
4. Data can be captured by screenscraper
   - ✓ Display data from protected applet
5. Data is extracted from browser memory
   - ~ Ensure browser is secure from attacks
   - ~ Use separate application for display
   - ✓ Display data from protected applet

Design


Standard Partition
Protected Partition
Browser
Web Page
Script or Plugin
Password and Display Applet

Summary / Next Steps

- LT provides security enhancements to applications
  - Begin internal discussions on how your applications can leverage LT
- LT's impact to development process can be successfully managed
  - Begin planning for impact of LT on your product roadmaps
- Early availability of LT Software Development Platforms
  - Contact your Intel representative for information on the Intel Early Access Program (EAP)

Fall ’03 U.S. IDF session presentations are available to IDF attendees only.
To download, go to:

`http://www.intel.com/idf/attendee`

```
Username:  attendee
Password:  fall2003
```

Thank you for attending.

Please fill out the

Session Evaluation Form.