

# Minding the App Store - Protecting Software and Device Features



**Benjamin Jun**  
VP & CTO  
Cryptography Research Inc., a division of Rambus



Session ID: ASEC-202  
Session Classification: Intermediate

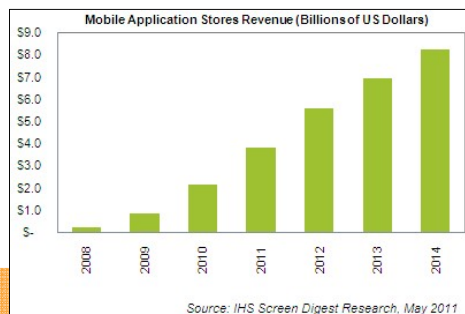
**RSACONFERENCE2012**

## The "app store" concept

1. User device has latent capabilities
  - Hardware platform
  - Pre-installed digital assets, configurations
  - Downloaded data
2. Pay \$, capability activated

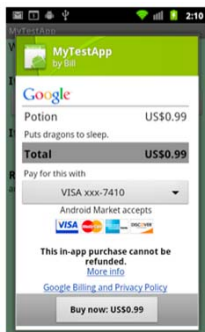
**"Revenue for Major Mobile App Stores to Rise 77.7% in 2011"**

- IHS iSuppli 5/2011



## What are we protecting?

- **Downstream (in-app) purchase**
  - Purchase, then download
  - Download, then purchase



In-app billing, developer.android.com



Word Lens upgrade



## What are we protecting?

- Downstream (in-app) purchase
- **Subscription revenue**

CHANNELS		NEW CUSTOMER PRICE	
80+ <a href="#">view lineup</a>		SHOWTIME® included for 6 months	Online Exclusive <b>\$29<sup>99</sup>/mo</b> for 12 months <a href="#">START SHOPPING →</a>
160+ <a href="#">view lineup</a>		SHOWTIME® included for 6 months	Online Exclusive <b>\$39<sup>99</sup>/mo</b> for 12 months <a href="#">START SHOPPING →</a>
200+ <a href="#">view lineup</a>		Plus our Sports Entertainment Package	<b>\$84<sup>99</sup>/mo</b> for 12 months <a href="#">START SHOPPING →</a>



XFINITY website Feb 2011



## What are we protecting?

- Downstream (in-app) purchase
- Subscription revenue
- **Product differentiation**



Price Average retail price (price may vary)	\$199.99	\$299.99	\$299.99
ENERGY STAR® qualified			
<b>Print and copy speed (ppm=pages per minute)</b>	B&W: Up to 32 ppm Color: Up to 31 ppm	B&W: Up to 33 ppm Color: Up to 32 ppm	B&W: Up to 35 ppm Color: Up to 34 ppm

## What are we protecting?

- Downstream (in-app) purchase
- Subscription revenue
- Product differentiation
- **Device subsidies**

**“AT&T’s subsidy rate has risen 178% since 2008, to \$172 — the fastest rate of increase in the industry... If AT&T is feeling pressured... it can choose to subsidize more heavily, hurting its margins but maintaining market share.”**

— AT&T’s flexible device subsidies could be a nail in Sprint’s coffin, M. Maisto, Connected Planet, 7/6/2011

**FREE phones**  
are in full swing

Online only. Free shipping!  
Select phones. 2-yr voice agreement with qualifying monthly data plan required.



Palm Pre™  
Pre3c™  
Pre3c Plus™

## What are we protecting?

- Downstream (in-app) purchase
- Subscription revenue
- Product differentiation
- Device subsidies
- **Trusted user environment**

**Path**

Address book usage

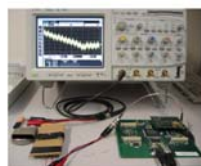


Fig. 2. Equipment used in our experiments. At top is a 4 GHz oscilloscope. At bottom from left to right: our non-invasive reader on ECU, and

Pacemaker hack (Halperin et al)



## When features are valuable, attacks follow

- Attacker motivations
  - Gain functionality / access
  - Obtain control of locked device
  - Sell an interoperable component



ECU re-flash

(Hondata)

- This talk...
  - Defining capabilities
  - Methods for policy enforcement
  - Security building blocks





## Describing Capabilities (who, what, where, when)

RSA CONFERENCE 2012

### Know your rights (1/2)

- Define controlled capabilities across all lifecycle phases
  - List should map to business rules
  - Completeness is important
  - Also important: understanding non-controlled capabilities

#### Examples:

- |                                    |                              |
|------------------------------------|------------------------------|
| ▪ Permit execution of signed code  | ▪ Access to online resource  |
| ▪ Ability to run unsigned code     | ▪ Use of peripheral ports    |
| ▪ Access to digital assets         | ▪ Use of on-chip component   |
| ▪ Developer / debug port           | ▪ # of content streams       |
| ▪ Performance / quality adjustment | ▪ Access to storage resource |

## Know your rights (2/2)

- Identify control points
  - UI actions associated with functionality
  - Capabilities enabled by software, hardware, server connectivity, local data, OS, etc.
- Recognize special constraints
  - Developer, manufacturing, debug, field repair, ...
  - Activation and de-activation cycles

Lifecycle phases
Development
Manufacturing
Identity provisioning
Regionalization
Normal operation
End of support
Dagnostic / field return



## What's in a name?

- User/device IDs...
  - "Fixed" numbers (system/OS, MAC)
  - Assigned identifiers (registry key/value, cookie)

```
[[UIDevice currentDevice] uniqueIdentifier];
```

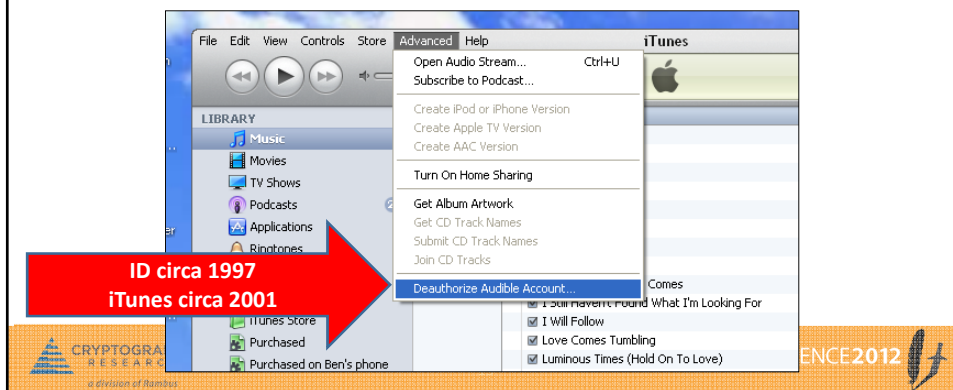
Retrieve device identifier (iOS)

- Some identifiers will change over time
  - Groups: Sales region, service contract ID, developer device pool
  - May be fluid: Geographic region, field return status



## Choose your IDs wisely (or, choose your legacy issue)

- Who owns the identity?
- What happens if ID changes or user migrates?
- Sometimes a single identity value works best
- Sometimes matching M-of-N identities is better



## Identities and keys

- Device, user, and group keys
- Certificates

**Android.security.KeyChain class**

Public Methods	
static void	choosePrivateKeyAlias (Activity activity, KeyChainAliasCallback response, String[] keyTypes, Principal[] principals) Launches an <code>Activity</code> for the user to select the alias for a private key and certificate pair for authentication.
static Intent	createInstallIntent () Returns an <code>Intent</code> that can be used for credential installation.
static X509Certificate[]	getCertificateChain (Context context, String alias) Returns the <code>X509Certificate</code> chain for the requested alias, or null if no there is no result.
static PrivateKey	getPrivateKey (Context context, String alias) Returns the <code>PrivateKey</code> for the requested alias, or null if no there is no result.

## Key management

- Choosing symmetric vs. asymmetric keys...
  - Assignment vs. in-field enrollment
  - Permission logic vs. encryption-based access control
- If keys applied properly, risk **shifts** to key management
- Before creating or importing a key
  - Who owns the key?
  - What requirements does (should) the key owner have?
  - Who may touch the key and how is access controlled?
  - How is the key assigned/provisioned?
  - What are the tamper resistance needs?

## Communicating rights (1/2)

- “Capability manifest” conveys client privileges
  - List specific privileges / capabilities
  - Identity ID, group bindings for recipient
  - May include contingencies: prerequisite authorization, local purchase log, authorized dates, etc.

Microsoft office product key

FlexNet license file

```
SERVER snoopy1 B37A6EF02758 22000
DAEMON cds1md /nfs/tools/example/tools.lnx86/bin/cds1md
USE_SERVER
FEATURE xxxxxxxx cds1md 10.1 20-aug-2012 675bc64e51ac548fc91e \
  VENDOR_STRING=:PERM DUP_GROUP=NONE vendor_info=27-jan-2011 \
  ISSUED=20-jan-2010 SN=2011-01-27T14:09:06:667 SIGN2=0084 A999 \
  9c54 7644 2402 7061 9000 0cfd 4974 c908 26f8 a9b5 6e21 016e \
  4d4f d8cc 2375 d0d8 7fe2 da29 cef0 c072 1c1a e1fc cabd 9461 \
  0e3c 1391 ebf3 abef v7.1_k=0d9ef10813cd4e2850eb
FEATURE xxxxxxxx cds1md 10.1 20-aug-2012 1 \
  4d4f1d9df69a86027fe2 VENDOR_STRING=:PERM DUP_GROUP=NONE \
  vendor_info=20-jan-2010 ISSUED=20-jan-2010 \
  SN=2011-01-27T14:09:06:667 SIGN2=048e 722e d01f 7d79 6743 \
  4a76 3421 8a05 0ec7 f1dc f860 a785 7500 0348 ac15 1232 0248 \
  e497 308d f08a 34e2 81a9 12b3 e1fc 1fa7 8d70 90a8 6b96 e88d \
  cff1 v7.1_k=1d0a5f29c030d2623e4a
FEATURE xxxx snsmgrd 2012.03 31-dec-2012 3 ac8a88af80ec6c362a88 \
  VENDOR_STRING=e5-ed4f817e996nfe00e081409798 ck=223 \
  START=31-mar-2010 SIGN=04fd0354212c
FEATURE xxxxx snsmgrd 2012.03 31-dec-2012 5 d8c7885ef35a08e07a56 \
  VENDOR_STRING=e5-fa3788a996h1dhk00e081409798 ck=187 \
  START=31-mar-2010 SIGN=849413610d46
```



## Communicating rights (2/2)

- File format: structured (bitmap) or freeform (XML)
  - Degree of parsing intelligence
  - Reverse compatibility requirements
  - Handling inputs from multiple sources (sales channels)
  
- Digitally signed, updated with in-field purchase
  - May include encrypted rights keys
  
- Message stored locally
  - ...ideally with minimal storage security requirements
  - Must address offline synchronization, revocation, updates, etc.

## Policy Enforcement



## Policy engine

- Ideally run on each access / invocation / boot
- Interpret, authenticate, and apply policy
  - Adjust software capabilities
  - Decrypt protected content

## Approach #1: "Active whitelist"

- **Positive** permission control
  - Activate features listed in the capability manifest
  - Most intuitive approach: Rules & business logic closely aligned
- Authorization checks embedded within application
  - Rights logic controls access to asset
  - Gating code is run at various points within application
- Security challenges
  - Security logic may not be sufficiently isolated from other apps
  - Does little to prevent copy of modified executable/content
  - Complex platforms may provide several ways to bypass checks

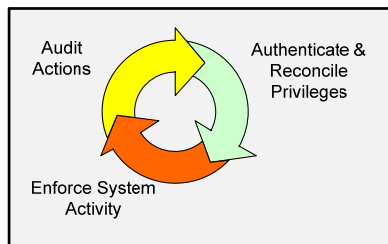
## Approach #2: “Behaviorally blacklist”

- **Negative** permission control
  - Detect when system is in policy violation, react accordingly
  - In many platforms, audit + react is less difficult to implement, offers more coverage
  - Foundation for reactive security
  
- Uses a (relatively) independent monitor
  - Ability to observe, compare against negative threshold criteria
  - Adequate control over device capabilities



## Approach #2: “Behaviorally blacklist”

- Secure monitor provides secondary control
  - Augments positive controls



- Think negative!
  - Negative control logic offers more points of enforcement
  - Design requires solid knowledge of system interactions, lifecycle





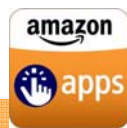
# Building Blocks for Authorization Management

23

RSACONFERENCE2012

## Off the shelf solutions

- Threats well understood
  - Encrypted pay TV delivery is 30 years old
  - ...yet **implementation robustness** and susceptibility vary
- Commercial, open source, and consortium offerings
  - Mostly for digital content, software, streaming data
  - Limited offerings for platform control, offline authorization, and high \$ value protection
  - Security threats require solutions to get close to platform, OS, hardware



E2012



Images provided to refer to example systems only

## Protected data containers (1/2)

- Package for secured installation
- **Read-only data**, authenticated by signature, possibly encrypted
  - Code, configuration, installation settings
  - Rights metadata
  - Multimedia container formats
- **Read-write data**, secured by platform or policy engine
  - Private application storage
  - Usage / purchase commitment history

## Protected data containers (2/2)

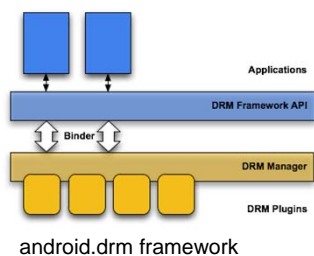
- Leverage crypto
  - Digital signing for authenticity, encryption for access control
  - Example: TPM provides attestation, seal/unseal operations
- “Roll-your-own” challenge: filesystem + databases
  - Easiest if policy engine (and not much else) is root
  - Ideally, assets can be re-authenticated on every reboot
  - Configuration manifest database
  - Leverage windows registry, linux /etc/, Mac OS property list

## Reasons to sign code

1. **Code authentication:** accept code from trusted sources } Important for closed platforms
2. **Code privileges:** privilege metadata specifies platform, process capabilities } Important for supervised app store
3. **Code review:** code reviewed by an authority who attests to its safety }
4. **Code responsibility:** code can be traced to a registered entity } Important for less controlled environments
5. **Code revocation:** code (or signers) that are discovered to be malicious can be revoked }

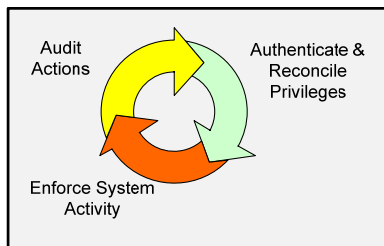
## Communication protocols

- Server – client messages
  - Authorization
  - Keep-alive / date update
  - Upstream report and audit
- Connection types
  - Live socket
  - Live-but-intermittent
  - Store-and-forward, offline
- Message generation infrastructure
  - Container creation and signing capabilities
  - Links with billing, manufacturing, and audit systems



## Hardening the policy engine

- Your policy engine must do several things right
- Carefully consider single points of failure





Negative enforcement

## Implementation strength is crucial

Policy engine property	Platform tools & capabilities
<b>Correctness of engine execution</b>	Authenticated bootloader Process separation State management on sleep/resume Glitch protection
<b>Preserve secrets (opacity)</b>	Process separation Side channel resistance Hardware key management
<b>Correctness of identity information</b>	Public keyring management Native platform access, hardware IDs
<b>Application control</b>	OS privilege Sandboxing
<b>Examine state of system</b>	OS privilege Process monitoring
<b>Hardware control</b>	Native code interfaces, hardware security partitions

## Leverage existing hardware and OS protection

- Principle of least privilege (security kernel , app )
- OS
  - Identity management
  - Process partitioning, protected storage
  - I/O infrastructure
- Hardware resources
  - Secured key oracles (TPM, payment API)
  - Partitioned cores (Intel ME)
  - Privilege management (ARM TrustZone)
  - Security features in application specific cores
  - Hardware key management

## Conclusions





## Protect your application platform

- Map the business requirements
  - Configuration manifest: identities, capability list
  - Understand how platform protections (OS / native code) map to control points
  - Mind system lifecycle
- Implementation
  - Consider positive and negative controls
  - Use existing building blocks in system, device, OS
- Client tamper resistance is important!



## Apply Slide

- In the next few months, you should:
  - Generate a feature management policy for all phases of product lifecycle
  - Evaluate datastructures for personality management and rights conveyance
- Within six months, you should:
  - Become familiar with general DRM offerings
  - Understand platform security building blocks for code authentication, execution control, and configuration management
  - Understand your tamper resistance requirements



## Contact Information

Benjamin Jun  
Cryptography Research, Inc.  
ben@cryptography.com  
415.397.0123  
[www.cryptography.com](http://www.cryptography.com)

