Kerberos + Android

A Tale of Opportunity
Platform Decisions

The Statistics
Why Go Mobile?

80% of the world's population now has a mobile phone.

( 5 Billion Phones )
Why Go Mobile?

Of those 80%, 1.08 Billion are smartphones.
Why Go Mobile?

In the US:

- 60%
- 40%

the ratio is even higher, with smartphones making up 40% of all mobile phones.
OK, well why **Android**?
Android?

Reason 1: US Market Dominance

- U.S. Smartphones (40%)
- Android 40%
- iPhone 28%
- Blackberry 19%
- Windows Mobile, 7%
- Windows Phone 7, 1%
- Other, 5%
Android?
Reason 2: Consumer Popularity

- **100 million** activated Android devices (now 400,000 / day)
- **200,000 apps** in Android Market (4.5 billion activations to date)
- **310 devices** available to consumers (112 countries)
Android?

Reason 3: Developer Popularity

- 450,000 developers building for the platform!
Android.
Meaning?

- **Opportunity** for increased Kerberos visibility
- **Useful** for Android and Kerberos developers
- **Fun** to see where the community takes it
Our Plan

What we wanted to do.
Goals
We wanted to fill a missing gap.

1. Port Kerberos **libraries** to Android

2. Port some C-based Kerberos **client apps** to Android

   - kinit
   - klist
   - kvno
   - kdestroy
Goals

We wanted to spark community involvement.

3. Build a sample **Android NDK App** (with a simple **GUI**)

4. Give changes back to **community**
Action!

What we did.
1. Crypto Implementation
Crypto
Added new CyaSSL crypto implementation

- Kerberos crypto options: 
  CyaSSL, OpenSSL, NSS, built-in
Crypto

Added new CyaSSL crypto implementation

- CyaSSL is very portable
2. Porting
Android Port
Kerberos Libraries + CyaSSL ➔ Android.

- Cross-compiled libraries for Android
- Created **shell script** for easy reproduction by developers
3. Android Application
Android App
Simple sample NDK project

Home Screen

- Single screen
- Uses JNI
- Wrapper around native client apps
Android App
Simple sample NDK project

kinit

- Gets a ticket using specified principal
Android App
Simple sample NDK project

**klist**

- Lists our tickets
Android App
Simple sample NDK project

kvno

- Gets a service ticket for the entered principal
Android App
Simple sample NDK project

klist after kvno

- Verify that we got a ticket
Android App
Simple sample NDK project

kdestroy

- Clear our ticket cache
Android App

Notes

• Uses a keytab instead of passwords

• Storage locations have been chosen for convenience
  
  Can be easily modified to what the developer needs
  
  Currently at /data/local/kerberos
Android App

License Type

• Application code will remain under the MIT license
4. GSS-API Wrapper
GSS-API
Java Wrapper

• Provide Java bindings for developers to use

• Uses SWIG framework

• Wrapper around native Kerberos GSS-API library

(Contains functionality found in gssapi.h)
2 example clients:

- Android client functionality
- Stand-alone Java app for desktop use
Example Client

- Est. context with example server
- Send wrapped message, verify returned sig. block (gss_wrap, gss_verify_mic)
- Repeat #2, but with gss_seal, gss_verify
- Misc. API tests and exit.
Example Server

- Est. context with client
- Receive and unwrap a message from the client
- Generate & send signature block for received message
The Future

What's happening next?
The Future
Look to the Community.

Availability

- Code will be linked from both MIT and yaSSL websites
The Future
Look to the Community.

PR Activity / Visibility

- Blog posts
- Forum posts
- Press releases
- GitHub
- Mailing lists
- etc...
The Future

Other ideas or thoughts?
References

Statistics

- [http://www.go-gulf.com/blog/smartphone](http://www.go-gulf.com/blog/smartphone)

Project Locations


- GSS-API Java Wrapper: [https://github.com/cconlon/kerberos-java-gssapi](https://github.com/cconlon/kerberos-java-gssapi)
Thanks!
www.yassl.com