Wallet
Secure Data Distribution and Management

Russ Allbery
June 6, 2006
Contents

• The Starting Point
• Overview and Goal
• remctl
• remctl v2
• Wallet Structure
• Classes of Data
• Authorization Methods
• Data Storage Methods
• Implementation Details
The Starting Point

- *leland_srvtab* to distribute *srvtabs* and *keytabs*
- keytab support underdocumented and strange
- Supports cached keytabs, questionable implementation
- Based on IBM sysctl, K4 authentication only
- Only supports ACL files on disk
Overview and Goal

- Distribute keytabs with better ACL management than kadmin
- Use the same method for certificates, passwords, etc.
- Store or automatically generate
- Allow upload of data into the wallet
- Flexible authorization system
- Self-contained distributable client
remctl

• Simple replacement for sysctl

• Already in extensive use

• Check an ACL and run a command, nothing more

• A few problems:
  – 32KB output size limit
  – Cannot stream output
  – No library interface
  – No persistant connections
  – Weird and underdocumented protocol
  – Unfortunate choice of ports
remctl v2

- Redesigned and *documented* protocol
- Down-negotiates to the old protocol when needed
- No output size limits, streaming output
- Persistent connections
- Client library with simple API
- Extensive test suite
Wallet Structure

- Server runs under remctl for authentication
- Stores predefined classes of secure data
- Split into three layers:
  - Data storage and generation layer
  - Authorization layer
  - Metadata layer that links the two
- Support arbitrary authorization and data methods
Classes of Data

- keytabs regenerated on each request
- keytabs pulled from the KDC database
- SSL certificate private keys stored by user
- Stanford-signed certificates generated on the fly
- Arbitrary files (database passwords) stored by user
- Almost certainly others to come...
Authorization Methods

- `remctl` provides authentication but only authorization for administrative functions
- Basic authorization: ACL files on disk or in MySQL
- PTS group membership
- LDAP group membership or attribute presence
- NetDB roles (particularly for host keytabs)
Data Storage Methods

- File stored on disk
- Data generated on the fly (kadmin for keytabs)
- Data retrieved from elsewhere (pre-existing keytabs)
Implementation Details

- Client linked statically with Kerberos, remctl
- Some Kerberos-specific knowledge in client: generating srvtabs, merging keytabs
- Server written in Perl
- MySQL for metadata store
- Runs on secure host, but not on KDC
- Backend remctl call to retrieve existing keys from KDC