Persistent Personal Names for
Globally Connected Mobile Devices

Bryan Ford, Jacob Strauss, Chris Lesniewski-Laas, Sean Rhea, Frans Kaashoek, Robert Morris
Massachusetts Institute of Technology

http://pdos.csail.mit.edu/uia
Connectivity Scenario: Stage 1

Bob & Alice meet, connect [Bonjour] – using **local names** (e.g., “Alice-PDA”)
Connectivity Scenario: Stage 2

Wish to re-connect remotely – need different, global names & more setup (e.g., “pda.alice1234.herisp.com”)
Connectivity Scenario: Stage 3

Meet again off-Internet – global names stop working!
Require **different, local names** (again)
UIA: Unmanaged Internet Architecture

Global Connectivity via Personal Names
- Like nicknames in cell phone address book
UIA: Unmanaged Internet Architecture

Global Connectivity via Personal Names

– Persistent: usable for local or remote access
UIA: Unmanaged Internet Architecture

Global Connectivity via Personal Names

- Persistent: usable for local or remote access

"Camera"

"Work-PC"

"Laptop"

"Home-PC"
Challenges

- **Intuitive setup**
  - As easy as plugging local devices together

- **Secure self-managing operation**
  - Don't make users understand key management

- **Namespace synchronization**
  - Changes on one device propagate to others

- **Partitioned Operation**
  - Remains available under limited connectivity

- **Namespace access control and revocation**
  - Handling lost or stolen devices
Contributions/Outline

Usability Concepts:
- Local Introduction, Remote Access
- Merging Devices to form Personal Groups
- Linking Groups via Personal User Names

Design Concepts:
- Secure device identities
- Optimistic state replication via change logs
- Overlay routing protocol leveraging social links
Usability Concept 1

Local Introduction, Remote Access
Local Introduction, Remote Access

1. Bob buys WiFi-enabled digital camera, "introduces" it to desktop PC at home
Local Introduction, Remote Access

1. Bob buys WiFi-enabled digital camera, "introduces" it to desktop PC at home

2. Bob takes camera on trip, stops at cyber-cafe, uploads pics to home PC for storage & sharing
Local Introduction, Remote Access

1. Bob buys WiFi-enabled digital camera, "introduces" it to desktop PC at home

2. Bob takes camera on trip, stops at cyber-cafe, uploads pics to home PC for storage & sharing

Use Device Mobility to solve Name Bootstrap Problem
Design Requirements for Local Introduction, Remote Access

Devices need:

• Stable identities
• Secure introduction procedure
• Way to route to current location
Endpoint Identifiers

Each device has an endpoint identifier (EID)
- Formed from hash of device's public key [SFS]
- Self-configured, stable, location-independent [HIP]
Device Introduction

Common case: meet in person on common LAN
- Browse network to find other device [Bonjour]
- Avoid man-in-the-middle attacks [Dohrmann/Ellison]

(screen shots from working UIA prototype)
Implementing Device Introduction

Devices exchange EIDs on introduction

– Use for finding + securely connecting in future

![Diagram showing devices exchanging EIDs](image.png)
Routing to Devices

Application

Personal Name ("laptop")

UIA Naming

Endpoint Identifier (EID)

UIA Routing

Application

IP Address Domain 1

IP Address Domain 2

UIA Routing

UIA Routing

UIA Routing
Routing to Devices

Overlay routing protocol [RON, i3, ...]

- Devices track peers in social neighborhood
  - Localized: works under partitioned operation
- Find mobile targets via limited flooding [Gnutella]

(more details in paper...)

Usability Concept 2

Merging Devices into Personal Groups
Device Names and Personal Groups

Each device has a user-controlled *personal name*
User merges devices to form *personal groups*
Personal Device Names

Short, convenient
- Like nicknames in cell phone address book
- Each device ships with manufacturer default name

"Camera"
"Tablet"
"Player"
"Laptop"
Personal Device Names

Short, convenient

- Like nicknames in cell phone address book
- Each device ships with manufacturer default name

“Coolpix”

“Nokia770”

“iPod”

“Thinkpad”
Personal Device Names

Short, convenient

- Like nicknames in cell phone address book
- Each device ships with manufacturer default name
- Binds human-readable string to device EID

"Coolpix" → EID 123

"Nokia770" → EID 234

"iPod" → EID 345

"Thinkpad" → EID 456
Personal Device Names

Short, convenient

- Like nicknames in cell phone address book
- Each device ships with manufacturer default name
- Binds human-readable string to device EID
- User can change as desired

“Coolpix” → EID 123

“Nokia770” → EID 234

“iPod” → EID 345

“Thinkpad” → EID 456
Personal Device Names

Short, convenient

- Like nicknames in cell phone address book
- Each device ships with manufacturer default name
- Binds human-readable string to device EID
- User can change as desired

“BobPix” → EID 123
“Tabloid” → EID 234
“MyPod” → EID 345
“Blinkpad” → EID 456
Merging Devices into Groups

Uses Device Introduction Procedure

**Personal Group**

- “BobPix” → EID 123
- “Tabloid” → EID 234
- “MyPod” → EID 345
- “Blinkpad” → EID 456
Design Requirements for Personal Groups

- Names Always Accessible from Any Device
- Support Partitioned Operation
- Consistency Management
- Revocation, Lost/Stolen Devices (see paper)
Implementing Names and Groups

Device keeps a series of change records

- Start with default name

Camera: EID 123

Series 123

“Coolpix” → EID 123

Laptop: EID 456

Series 456

“Thinkpad” → EID 456
Implementing Names and Groups

Device keeps a *series* of change records

- Start with default name
- To rename: cancel old, write new name record
Implementing Names and Groups

Device keeps a *series* of change records

- Start with default name
- To rename: cancel old, write new name record
- To merge:
  1. Write merge records
Implementing Names and Groups

Device keeps a series of change records

- Start with default name
- To rename: cancel old, write new name record
- To merge:
  1. Write merge records
  2. Gossip series contents

```
Camera: EID 123
Series 123
Series 456 copy

“BobPix” → EID 123
“Thinkpad” → EID 456

Laptop: EID 456
Series 456
Series 123 copy

“BobPix” → EID 123
“Thinkpad” → EID 456
```
Handling Name Conflicts

What if user merges two devices w/ same name?

⇒ merge succeeds, but *creates name conflict* (can't use name)

Resolve by renaming (on either device)
Handling Name Conflicts

What if user merges two devices w/ same name?

⇒ merge succeeds, but **creates name conflict** (can't use name)

Resolve by renaming (on either device)
Implementing Conflict Resolution

When user merges two devices w/ same name:

- Camera: EID 123
  - Series 123
  - “Coolpix” → EID 123

- Camera: EID 456
  - Series 456
  - “Coolpix” → EID 456
Implementing Conflict Resolution

When user merges two devices with same name:

- Bindings of same name to different target EIDs → conflict

```
Camera: EID 123
Series 123
Series 456 copy
```

```
“Coolpix” → EID 123
“Coolpix” → EID 456
```

```
Camera: EID 456
Series 456
Series 123 copy
```

```
“Coolpix” → EID 456
“Coolpix” → EID 123
```
Implementing Conflict Resolution

When user merges two devices w/ same name:

- Bindings of same name to different target EIDs ⇒ conflict
- On rename, write:
  - Cancel for old name
  - New name

- Camera: EID 123
  - Series 123
  - "Otherpix" → EID 123
  - "Coolpix" → EID 456
- Camera: EID 456
  - Series 456 copy
  - Series 123 copy
  - "Otherpix" → EID 123
  - "Coolpix" → EID 456
Usability Concept 3

Linking Groups via Personal User Names
Personal User Names

• Assign short personal names to friends for easy communication and sharing

Bob's Group
  - Laptop
  - Camera
  - Home-PC
  - Work-PC

Alice's Group
  - iPod
  - PowerBook

Charlie's Group
  - PC
  - Phone
Introducing Users

1. Meet, find other user's device in LAN browser
2. Click “Introduce as New Contact”
3. Enter personal name for user
User-Relative Naming

Browse tree to find desired device

Enter user-relative domain name
Implementing User Names

On introduction:

1. Exchange EIDs
2. Write User records: name → series
3. Gossip series contents

Groups remain separate, only linked via names

Implicit notion of “user” → no per-user keys
Gossip Among Multiple Devices

- Devices gossip whenever possible with
  - Other devices in personal group
  - Devices in friends' groups
Name Resolution

- Resolution starts in device's own group
- Resolve components right-to-left
- Use gossiped records – no communication
Other Design Elements

See paper on:

- Device introduction security
- Groups shared between users (“PhotoClub”)
- Group ownership
- Revocation – lost/stolen devices
- Access control using personal names
- Routing efficiency
Implementation Status

Runs on Linux, Mac OS X, Nokia 770 Tablet

UIA Control/Group Browser
- UIA Client API Library

UIA-Aware Application
- UIA Client API Library

Legacy Application
- Sockets API
- DNS Resolver

UIA Name Daemon

UIA Router

TCP/IP Protocol Stack
- Network Drivers
- tun Driver

Operating System Kernel

RPC

DNS Proxy

tun Wrapper
Legacy Application Support

- **Disguises**
  - UIA names as DNS names
  - EIDs as IP addr

- **SSH, HTTP to NATted devices via UIA names**
  - Transparent connection migration, etc.

- **Creates virtual LAN of personal devices**
  - LAN local discovery apps work remotely via UIA (e.g., Apple File Sharing using Bonjour)
Demo Video
Implementation Observations

Proof-of-concept prototype
- Many rough edges...

But demonstrates the architecture
- Logs not too big: ~40K in example
  - Small name records, infrequent changes
- Router tables, overhead not too large
  - Only track “social neighbors”, not whole world
Discussion: Global vs Local Names

Global names:
- Perfect when global usability is the whole point
- Rare, expensive, cumbersome in personal context

Persistent personal names:
- Short, convenient, unrestricted assignment
- Work anywhere, any time, on all user's devices
Future Work

- More flexible group management
- Better Access Control
  - “Allow Bob's devices to print”
  - Hide Alice from Carol
- Scalability issues
  - How far, how long to gossip name state
  - How widely router should monitor peers
- Wider deployment
Related Work

- Dynamic DNS, Mobile IP, IPSEC VPNs
- Decentralized security: SDSI/SPKI
- Host identities: SFS, HIP, JXTA, i3
- Naming/routing: DDNS, TRIAD, i3, CoDoNS
- Optimistic replication: Ficus, Coda, Ivy
- Mobile data: Rumor, P-Grid, Roma, Footloose
- Social networking: Turtle, Sprout, F2F, Tribler
 Summary

UIA simplifies global device connectivity through **persistent personal names**

Based on three key usability concepts:
- Local Introduction, Remote Access
- Merging Devices to form Personal Groups
- Linking Groups via Personal User Names

http://pdos.csail.mit.edu/uia/
Device Introduction Security

Users exchange random words out-of-band
- Short: need only resist online attacks
- Multiple-choice: ensures user participation
Routing via Opportunistic Rendezvous

- Laptops have moved, both now behind different NATs
  - Each reconnect to previous neighbors (if available)
- Routing layer must re-connect devices
  - Leverages stable hosts as rendezvous points (desktop)
  - Use desktop to forward packets or punch holes in NATs
  - Unlike Mobile IP or VPNs, no need to configure a home agent
Shared Groups

- Users can organize names in *shared groups*
  - Ex: Bob creates “PhotoClub” group, adds Alice & friends interested in photo sharing
  - Alice copies Bob's “PhotoClub” link into her group
Simulation Studies

• Key questions:
  – Can UIA's algorithms provide reliable connectivity on larger, realistic (social) networks?
  – How much overlay overhead does it impose to provide this service?
Simulation Methodology

- Uses partial crawl of Orkut tree [LiXXX]
  - 2363 users in social network
  - Highly skewed friendship degree: median 7, maximum over 1000.
  - Assume one device per person (pessimistic)

- Simulate construction of UIA routing overlay, then simulate searches for particular nodes
  - Primarily interested in searches for “friends”
  - Secondarily in 2\textsuperscript{nd} or 3\textsuperscript{rd} degree neighbors
Search Success Rate

Fraction of Connections Successful

Percent Stable Nodes

256 Tokens
64 Tokens
32 Tokens
16 Tokens
0 Tokens
Search Overhead

Mean Messages Sent Per Connection

Percent Stable Nodes

Ideal hop count-limited
256 Tokens