NDN: Plug and Play

John Dellaverson, Tianyuan Yu, Eric Newberry, Zhiyi Zhang, Lixia Zhang

Internet Research lab @ UCLA

ACM ICN 2020 Tutorial

September 22, 2020

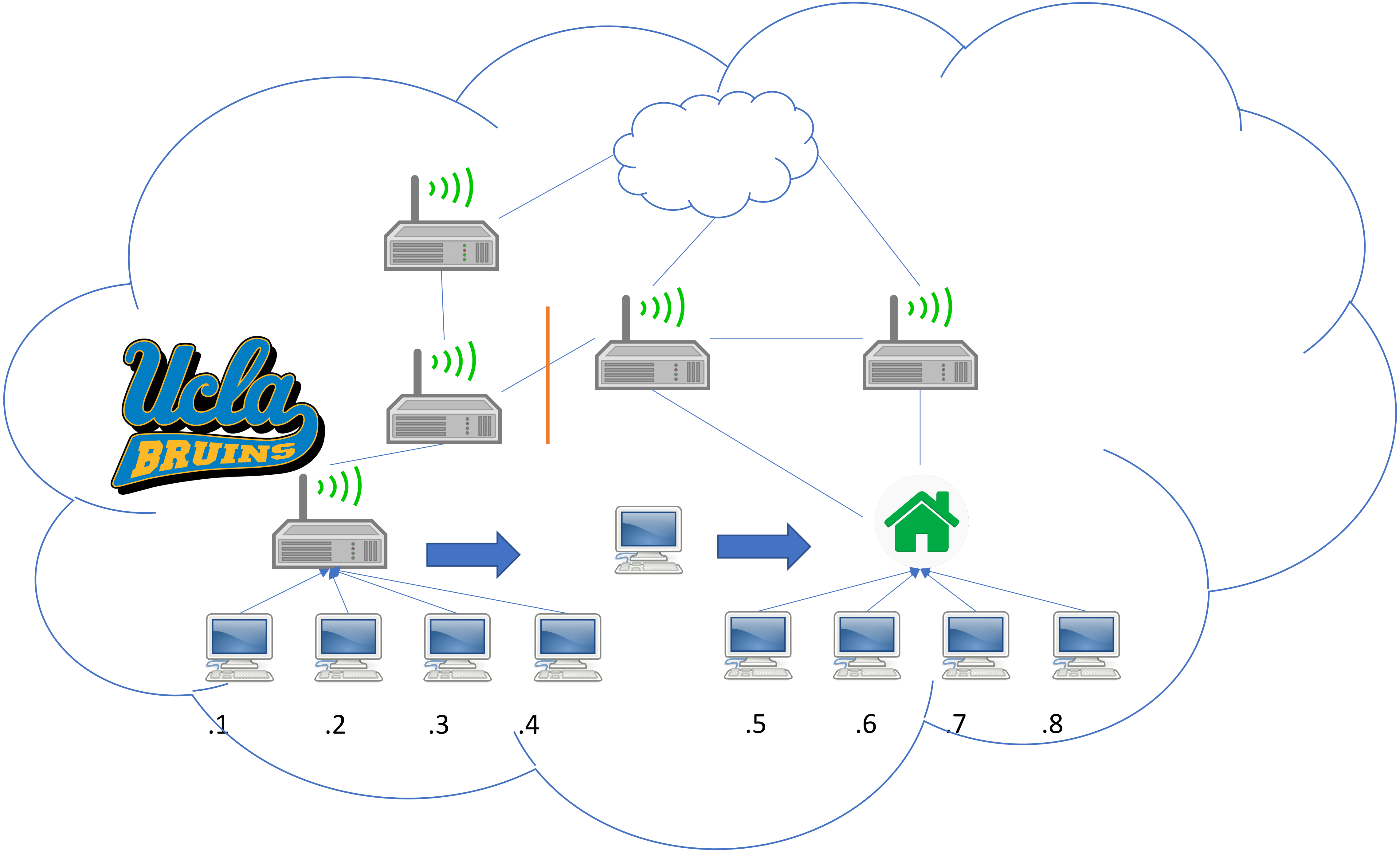
What is Plug And play

- Usually refers to the ease of use when first connected.
- Configuration (plug)
  - Configuring something into its operational networked environment
- Connectivity (play)
  - Being able to send and receive packets

- How do these steps differ between IP and NDN?

IP Configuration

- One abstraction of IP is simply pipes connecting all nodes.
- What is the goal of IP configuration? Put simply, connectivity to the global internet. This arises naturally from the IP abstraction.
    - IP configuration is to plug a node into an existing connected IP infrastructure.


Ucla
BRUINS
.1
.2
.3
.4
.5
.6
.7
.8

IP Configuration: Consequences of Simplicity

- Because IP is just about connectivity, everything involving trust happens on a higher layer
- Can’t know if talking to the right party over just IP
- Thus vulnerable on its own.
  - Source address spoofing
  - DDOS
  - Biological analogue
- Not to bash IP, just acknowledgement of change in networking

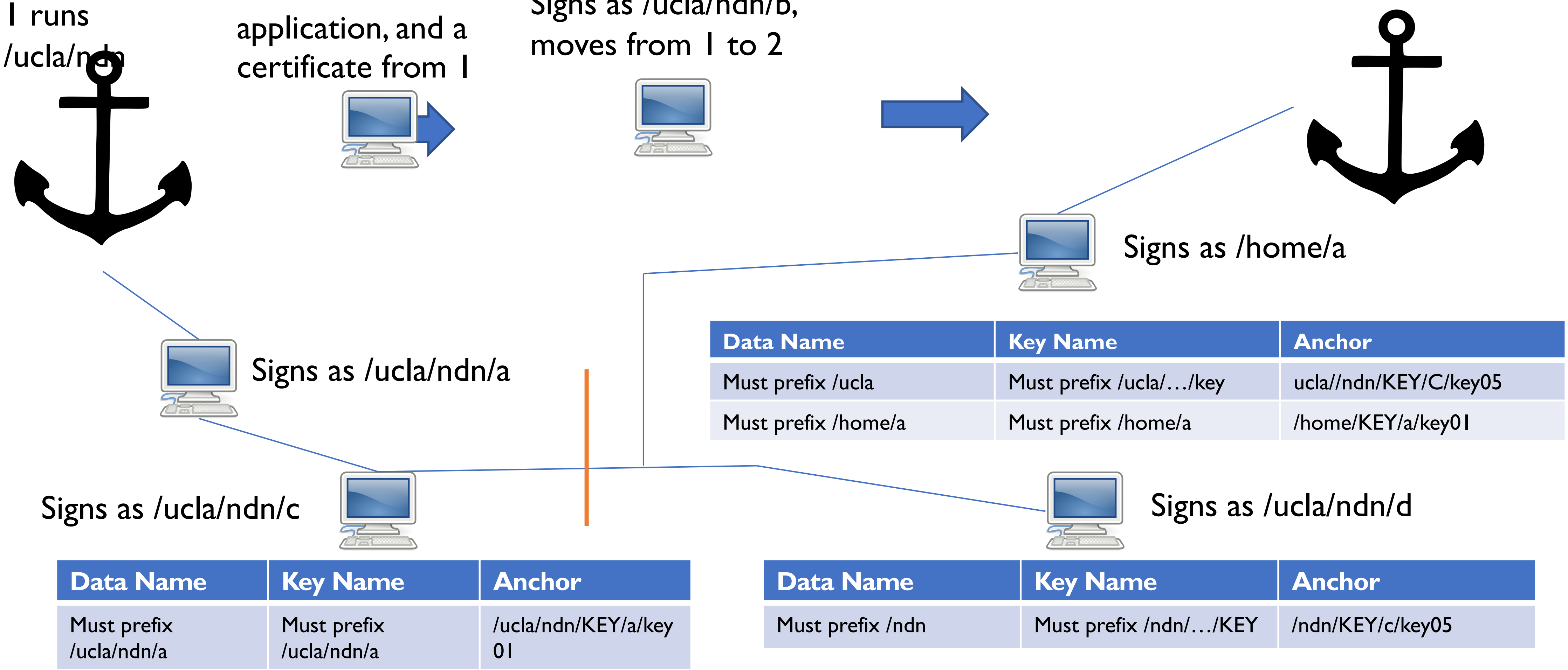
IP Reachability

- Fairly trivial, once config established: just send to either local network or to default gateway.
    - Connectivity vs Reachability
- True reachability somewhat more complex, but irrelevant for edge user.

NDN Configuration – aka ‘plug’

- What is NDN Configuration?
  - Plug a new entity into an application namespace. Because trust relations exist within this space, also inherit the trust relations of that namespace with other names.
    - Because asserting trust dynamics here, authenticity is critical.
  - All about Names and Security
  - All configuration can be encapsulated in getting a Name from a relevant application and Certificate from a trust anchor.
  - From there, can retrieve a Trust Schema, can automatically discover namespaces.
  - Does not have to be a one-time thing

NDN Config Visualized



Trust anchor 1 runs /ucla/ndn

Got the name prefix /ucla/ndn/b from an application, and a certificate from 1

Signs as /ucla/ndn/b, moves from 1 to 2

Trust anchor 2. Runs /home

Signs as /home/a

Data Name	Key Name	Anchor
Must prefix /ucla	Must prefix /ucla/…/key	ucla//ndn/KEY/C/key05
Must prefix /home/a	Must prefix /home/a	/home/KEY/a/key01

Signs as /ucla/ndn/a

Signs as /ucla/ndn/c

Signs as /ucla/ndn/d

Data Name	Key Name	Anchor
Must prefix /ucla/ndn/a	Must prefix /ucla/ndn/a	/ucla/ndn/KEY/a/key01

Data Name	Key Name	Anchor
Must prefix /ndn	Must prefix /ndn/…/KEY	/ndn/KEY/c/key05

Many Ways to Play

- Once an entity has the aforementioned essential components (Name, Certificate → Trust Schema), can get connectivity in multiple ways:
- Different ways of getting connectivity
  - NDN broadcast self-learning
  - NDND
  - NDN-autoconfig (not actually an auto-configuration system)
  - NLSR

Contrasting ip and ndn configuration

	Primary Focus	Band	Namespace	Security
NDN	Identity and Security	Out-of-band requirements on security and configuration.	Application-level name space	High – packets are signed, means that after configuration can guarantee authenticity.
IP	Connectivity	In-band: because simply setting up connectivity, DHCP suffices.	Topologically named space.	None!

Why the differences?

- IP designed 40 years ago for connectivity
- Modern applications run in DNS namespace & use TLS for security
- NDN is younger, took the opportunity to capture these needs in the design.

TLS – Config and security

- TLS not necessarily the default
  - DDOS attacks already mentioned
  - Majority of websites only using TLS in 2018
- TLS fundamentally can’t use this decentralized/local trust model.
  - NDN trust model is local and decentralized

Ways to implement plug and play

- Try to make the ‘configuration’ stage of plug and play as easy as possible by automating steps
  - Some security steps (e.g. Name, Trust Anchor, Cert) can’t be automated
- Provide easy ways to safely input the Trust Anchor + Certificate + Name.
- Automate connectivity past that

Our work

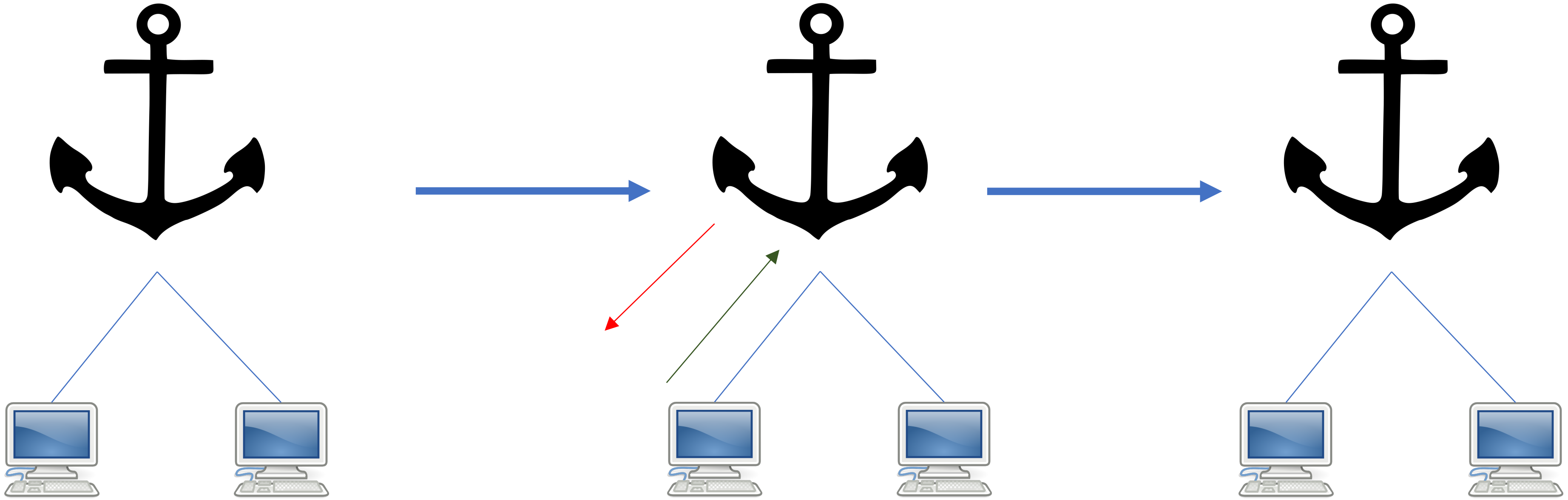
- This work is available on on Github
- Relatively simple
- Connects to other machines in one-hop WiFi or Ethernet range
- One machine designated as anchor.
- Other machines request a certificate, receive one (with the name specified by the server).
- Can easily imagine doing this while an application is already running.

Interest:

/ndn/config/<required-info>/<nonce>

Identity:
ndn/config

Certificate for
My-ndn/a

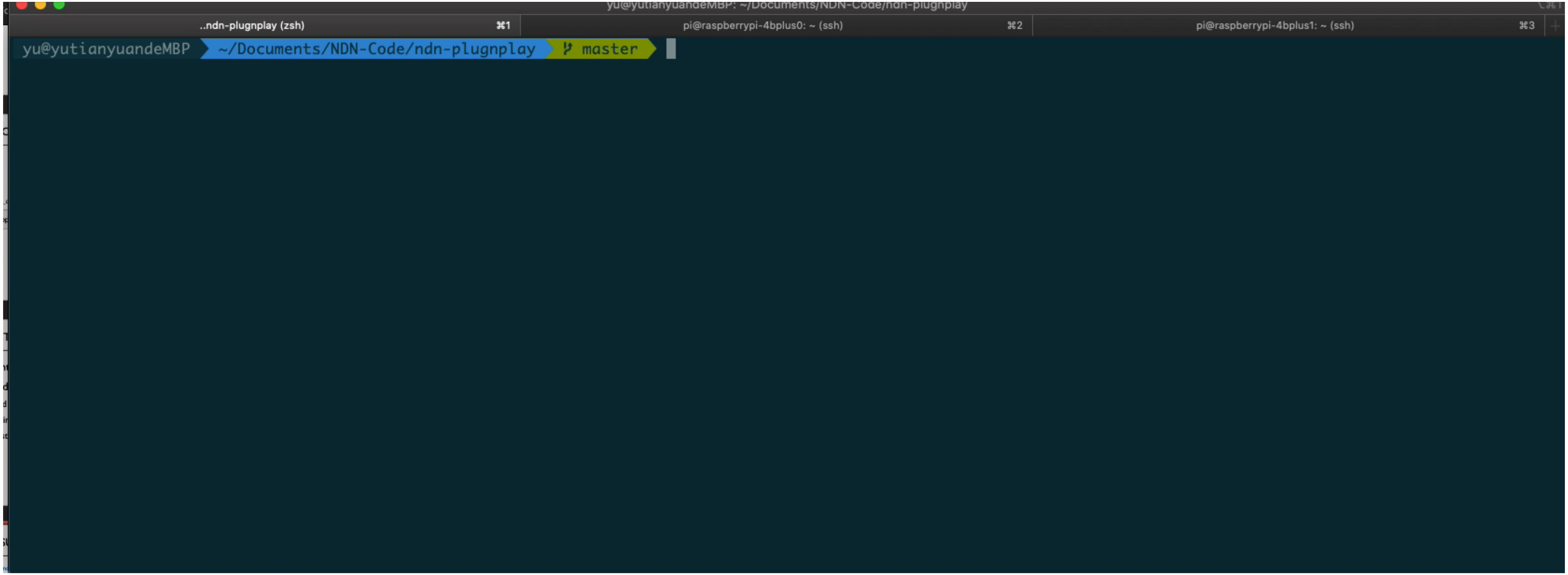


Identities:
/localhost/operator

My-ndn/a

My-ndn/a

My-ndn/b


yu@yutianyuandeMBP: ~/Documents/NDN-Code/ndn-plugnplay
..ndn-plugnplay (zsh) ⌘1
pi@raspberrypi-4bplus0: ~ (ssh) ⌘2
pi@raspberrypi-4bplus1: ~ (ssh) ⌘3

```
yu@yutianyuandeMBP  ~/Documents/NDN-Code/ndn-plugnplay   master 
```

Enabling applications

- NDNSD
  - Service discovery and publishing
  - Sync over namespaces (e.g.'/discovery/printers')
  - Assumes trust relations already configured
- npChat
  - Decentralized multimedia sharing app
  - Works off of application level pub-sub model
  - Assumes existing certificates

Future Directions

- Further clarify the difference between IP and NDN Configuration.
- Develop our tools such that they can support more complex cases.
    - E.g. pure consumers, not immediately near one another, etc.
- Develop the automation tools for connectivity.
- Integrate these two sets of tools into a ‘plug and play’ software that users and developers can use.

Conclusion

- As NDN developers, we should try to understand what the fundamental requirements for configuration are, and pare away extraneous pieces.
- Make ‘playing’ with NDN as easy as possible.
- Setup and more on this topic in the next talk!