# APACHECON 2014

# Leveraging Linux platform for Identity Management in Web Applications

**Dmitri Pal**
**Sr. Engineering Manager Red Hat, Inc.**

# Once upon a time...

- There has been a cool idea!

# And the ball started rolling...

- Development started
- Days... nights... weekends...
- One person... several... community
- Core functionality emerged
- Time to show someone!

# Realization

- Application is not just core functionality, it needs to work with/for different users
  - Ordinary users, admins of different levels
- No time! Let us create a superuser who can do everything and deal with different users later
  - We just want to show how cool the application is and what problems it would solve
- What about the password?
  - Let us hard code or stick into a config file

# First POC Deployment

- OMG! We forgot that we need to add support for multiple users...
- Let us do it quickly...
- Local SQL database will do...
- We can fix it over the weekend...
- It is good enough for now!

# First Real Production Deployment

- Customer has some kind of LDAP directory, who knows what it is...
- Framework has support for LDAP...
- I should be able to figure things out quickly...
- It is just last step that need to do and I am done forever with the user management.
  - Well... NO! This is where the real complexity begins.

# Who Are My Users?

- End users:
  - Users coming from the internet
  - Users that are a part of the enterprise
  - Contractors, partners, providers, suppliers...
- Power users:
  - Enterprise sysadmins
  - Service provider
  - IT services subcontractor

# Enterprise Applications

- On Premise:
  - Users/admins come from:
    - Domain controllers (LDAP + Kerberos + more):
      - Active Directory
      - IdM (FreeIPA)
    - LDAP directories:
      - 389, OpenLDAP, ApacheDS, SunDS, Oracle OID...
- Managed service:
  - End users and some power users - from customer sources
  - Power users (those who manage tenants) - from managed service provider

# Consumer Applications

- End users can be stored in a directory of choice
- Power users usually come from the service provider namespace
- Who manages the platform?
- Who manages the cloud infrastructure?
- What is the relation of multiple layers in the stack identity wise?

# The Point

- Identity management does not end with just LDAP...
- Complexity only starts there...
- One needs to think about:
  - What identity my application uses connecting to other resources, for example LDAP, to fetch my users. How is it authenticated? What is the security of this connection? How do I avoid common security pitfalls: cleartext passwords, lack of encryption, sensitive data in configuration files

# More To The Point

- Security of connections
  - Identities, passwords, keys, certs
- Multiplexing identity sources
  - Different LDAPs, Domains, Forests
- Multitenancy
  - User compartmentalization
- Failover
  - Each directory consists of multiple servers
- Offline situation (connection to all LDAPs lost)

# Application Modes

- Every application needs to work in different modes:
  - Development – simple, no central LDAP
  - Demo – emulated users with roles, single box
  - POC – using a dummy directory
  - Production – all sorts of different identity sources as mentioned above

# Complexity Matrix

- Evolution of the application leaves a lot of cruft that is hard to maintain or clean
- Different modes of operation dictate different identity sources
- Different production use cases and requirements create a lot of complexity
- Add compliance requirements and audits...

**Welcome to the identity management nightmare!**

# What if...

- Can we offload all this complexity somewhere?

# What if...

- Can we offload all this complexity somewhere?
- How about Linux operating system?
  - Platform needs to deal with all this complexity
  - It already supports different sources of identity
    - LDAPs, AD domains, IdM (FreeIPA), trusts...
  - Has offline caching
  - Has secure connections
  - Undergoes audit

# Couple Words About SSSD

- SSSD = System Security Services Daemon
- A group of services that connects a machine to identity sources of your choice
- Supports multiple identity sources
  - AD, IdM (FreeIPA), LDAP
  - Direct connection or trusts
- Provides authentication and identity data
- Secure connection using host identity and key

# SSSD (continued)

- Failover, DNS discovery, sites
- Caches information offline
- Can have a local domain/source
- A part of all known Linux distros

# Gap Analysis

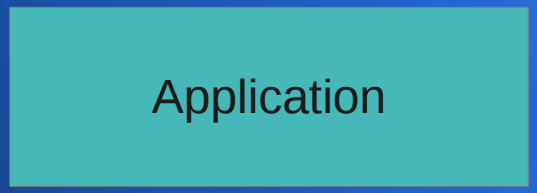
- SSSD was focused on the identities needed to access a system = POSIX
- Applications do not require POSIX
  - Can be POSIX but might not be
- Applications need an object oriented way to request extra data
- Applications might require extended attributes
  - Email, avatar, locale, custom properties, group membership

# Recent SSSD Enhancements

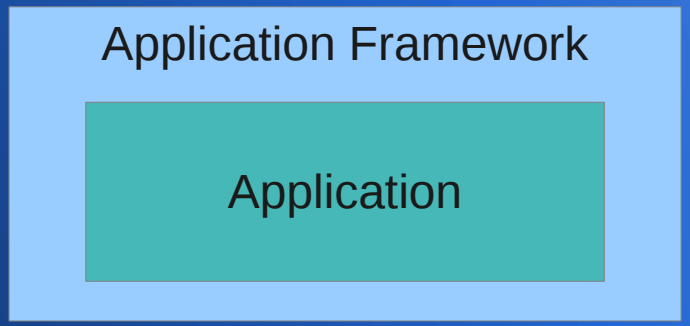
- D-BUS interface to fetch identity information
- Ability to define which attributes need to be fetched from the central source besides POSIX attributes
- Ability to serve different identity data to different consumers*.

**- in works*

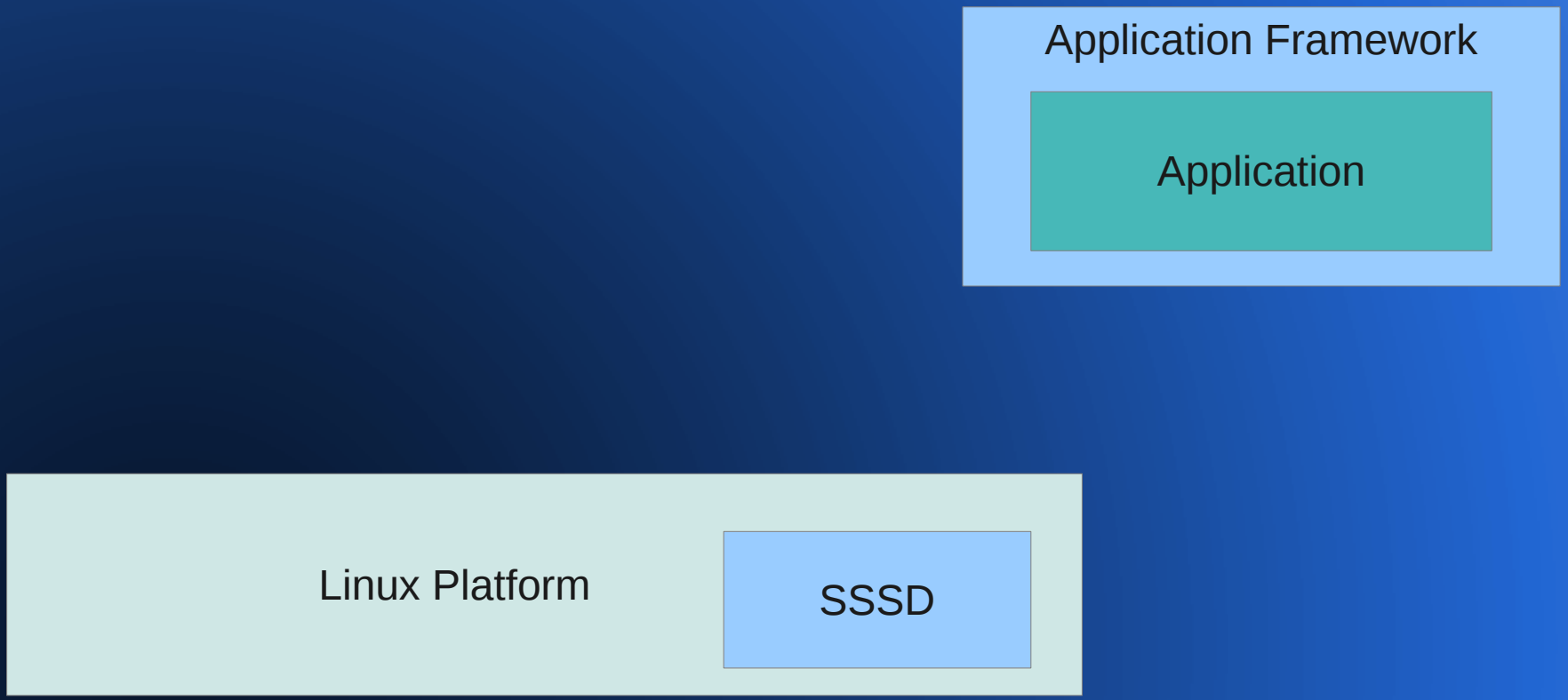
# Architecture



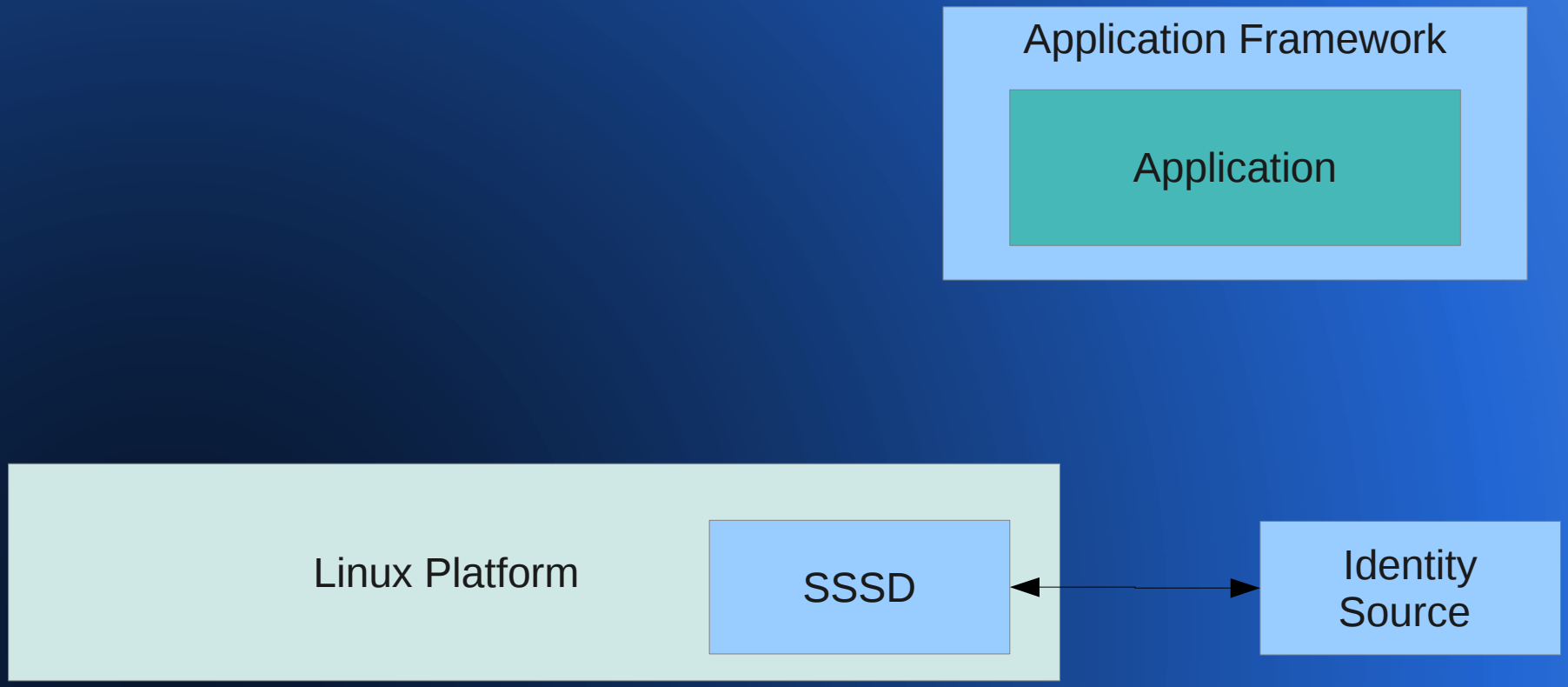
# Architecture



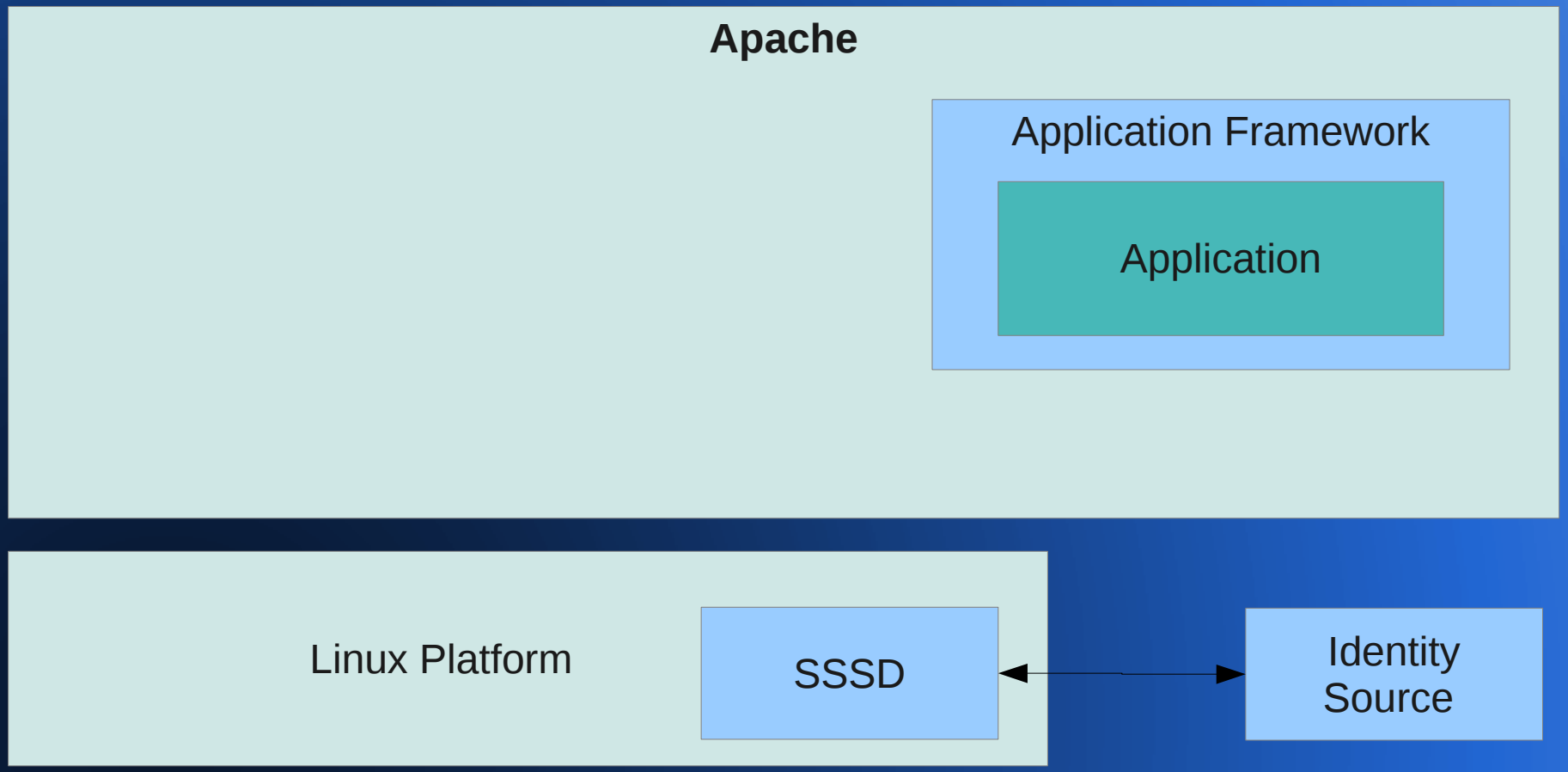
# Architecture



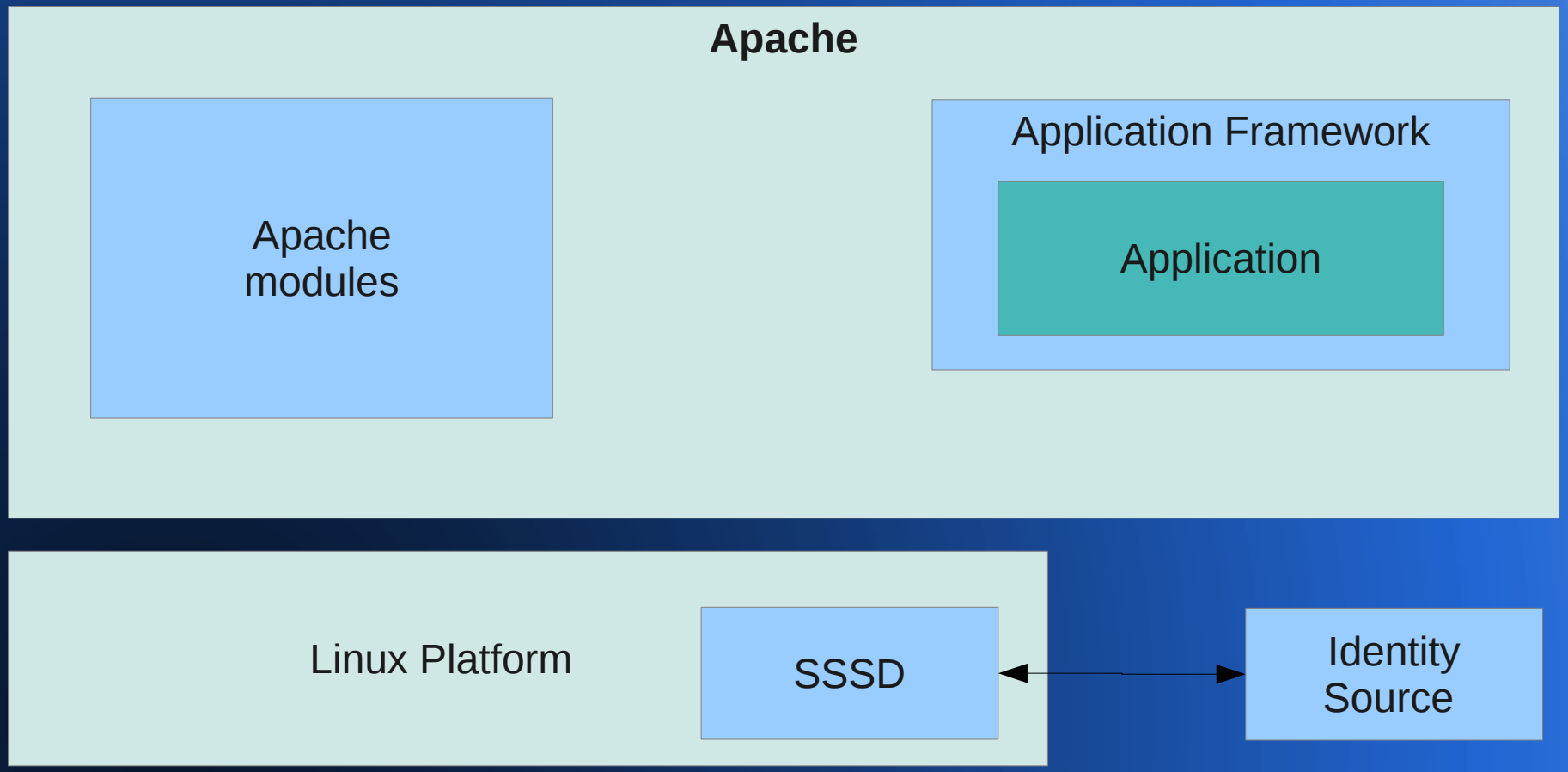
# Architecture



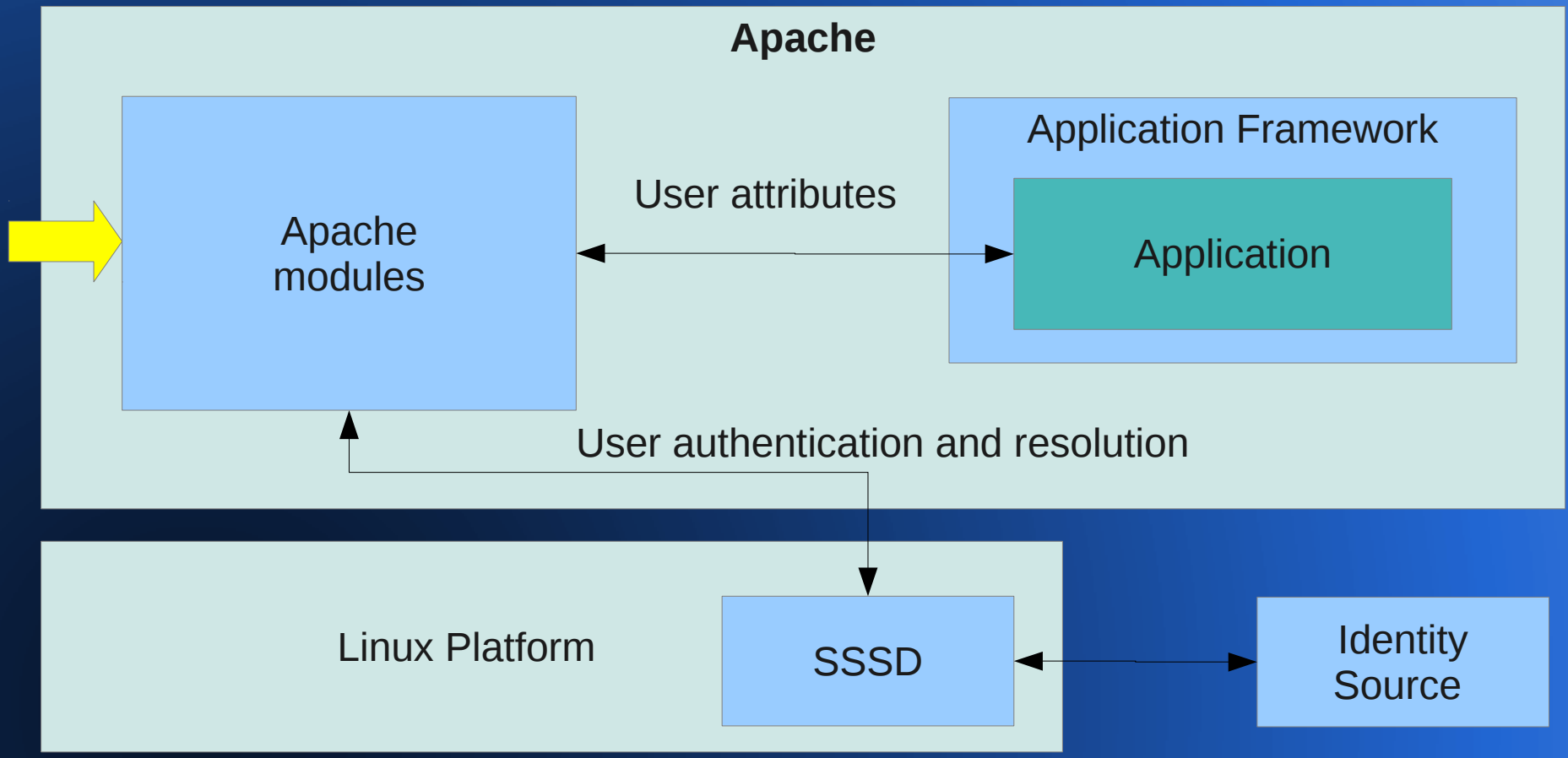
# Architecture



# Architecture



# Architecture



# Information Flow

- Request hits a URL
- Modules intercept the request
- Authenticate
- Fetch related information
- Pass it to the application via environment variables
- Application reads the variables and uses them

# Variables and Data

- REMOTE_USER – login of the authenticated user
- Extended attributes
- Group membership
- Other:
  - Domain
  - Subject, issuer (for certs)

www.freeipa.org/page/Environment_Variables

# Apache Modules

- Authentication
  - mod_auth_krb – kerberos SSO
  - mod_intercept_form_submit
    - Intercepts an application provided form
    - Preserves look and feel
  - mod_nss/mod_ssl
    - Certificate based authentication
  - mod_auth_mellon
    - SAML based federation

# Authorization/Access Control

- Access control might be checked outside of the application too
- mod_authnz_pam
  - Use PAM stack to perform authorization check
  - Can leverage centralized access control capabilities like FreeIPA host-based-access-control
  - Application can be given a name and this name can be factored into the access control rules

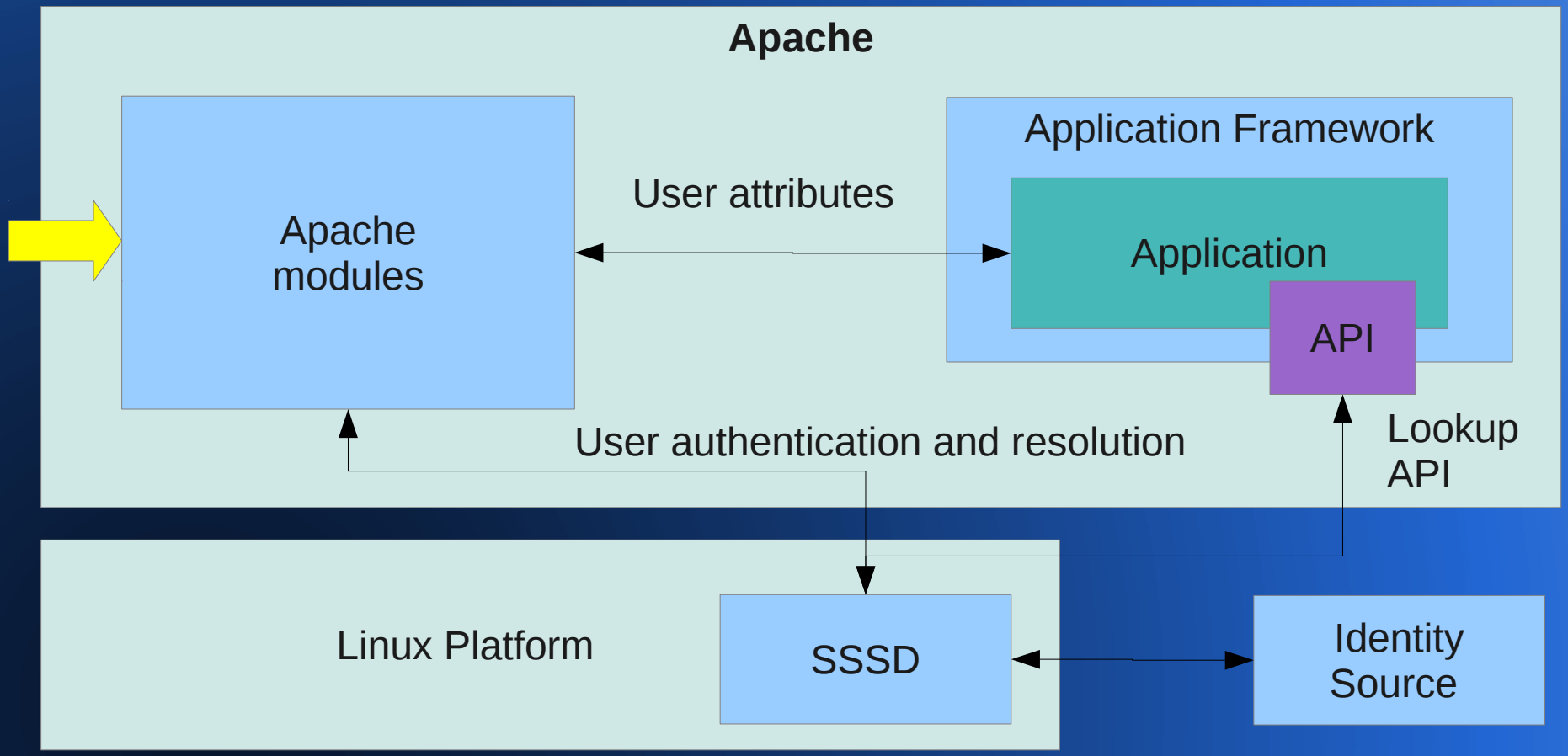
# Identity Data

- Sometimes comes from authentication module
  - User Kerberos principal
  - Data from a certificate
  - Data from a SAML assertion
- mod_lookup_identity
  - fetch additional attributes from SSSD based on the REMOTE_USER attribute

# Use Cases

- Authentication and user info to determine what user is entitled to do
- Administrative workflows require more lookups
  - Mapping central group to application roles
  - Mapping users to local groups
  - Mapping users to roles

# Architecture



# Benefits

- All complexity is removed from the application
- Easy to use application in all required modes
- Enables use of the application in multiple deployment scenarios without modification
- Faster development and delivery
- Optional and flexible
- **NO MORE DIRECT LDAP CONNECTION**

# Next Steps

- Read the integration guide
  - **http://www.freeipa.org/page/Web_App_Authentication**
- Update application to use attributes and lookup API
- Setup your application in different modes
  - Development, demo, POC, production
- Provide feedback
- Enjoy variety of deployments

# Resources

- FreeIPA
  - Project wiki: www.freeipa.org
  - Project trac: https://fedorahosted.org/freeipa/
  - Code: http://git.fedorahosted.org/git/?p=freeipa.git
  - Mailing lists:
    - freeipa-users@redhat.com
    - freeipa-devel@redhat.com
    - freeipa-interest@redhat.com
- SSSD: https://fedorahosted.org/sssd/
  - Mailing lists:
    - sssd-devel@lists.fedorahosted.org
    - sssd-users@lists.fedorahosted.org

# Questions?