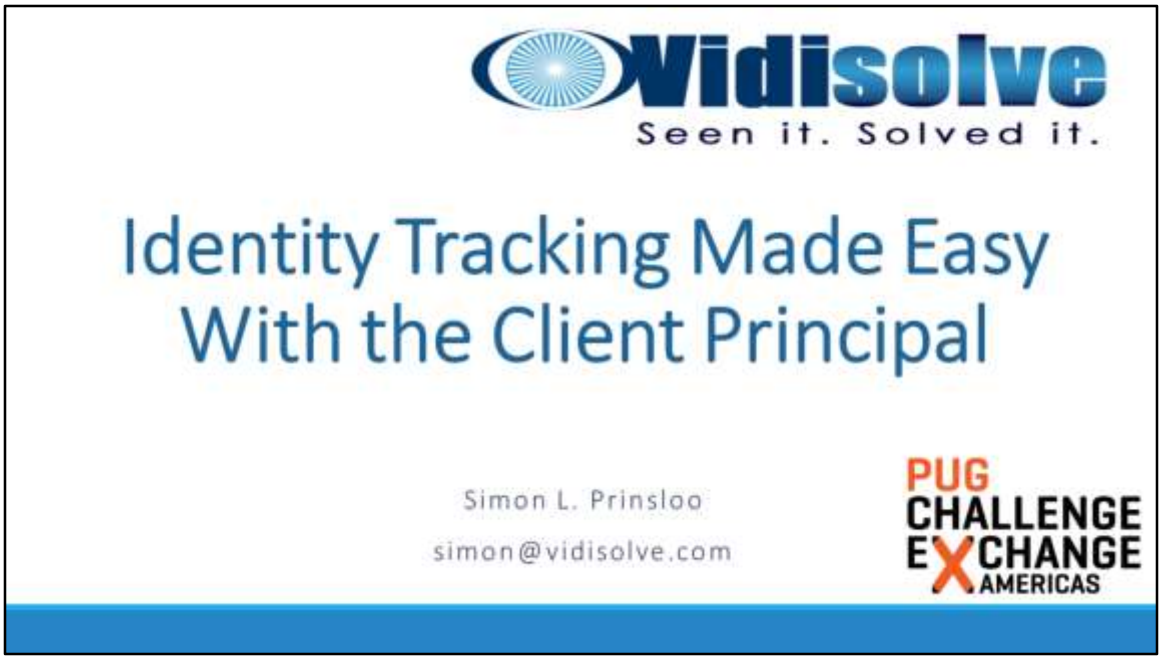


Seen it. Solved it.

Identity Tracking Made Easy With the Client Principal

Simon L. Prinsloo
simon@vidisolve.com

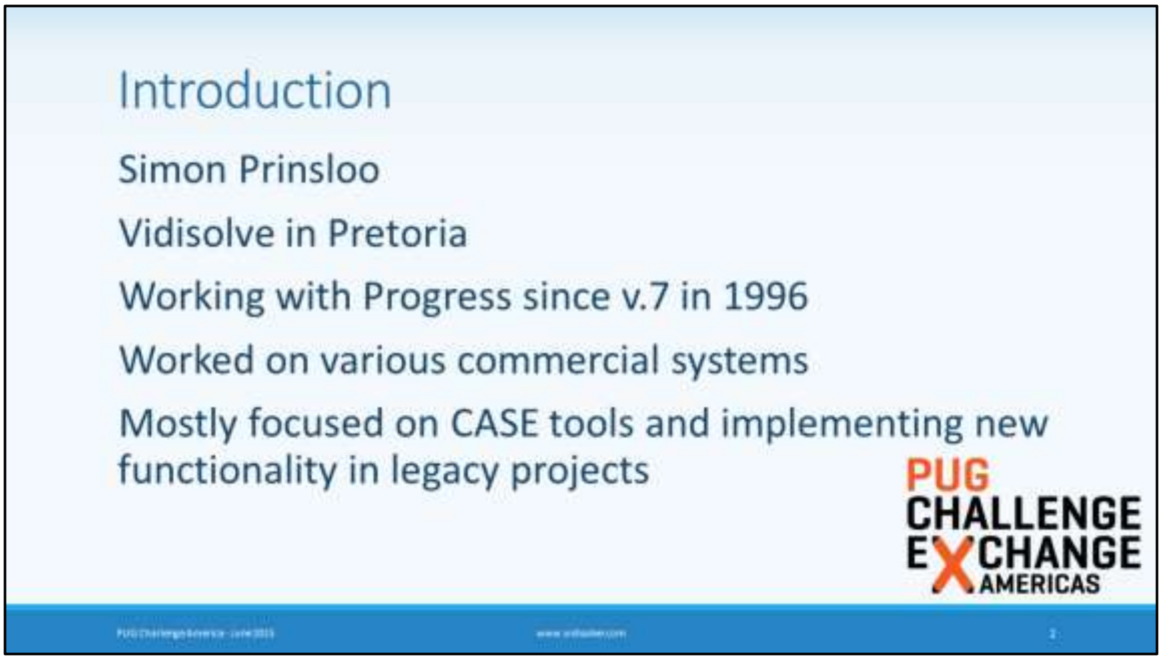
PUG CHALLENGE EXCHANGE AMERICAS

Simon has been working with the Progress 4GL and the database for the last 18 years. Keeping up with the developments in the 4GL and the database has always been his passion.

Simon has been involved in CASE tools since his first introduction to the 4GL and worked on various other systems, including the Namibian Tax System, large ERP systems in Southern Africa and a number of smaller systems.

Progress Software South Africa used Simon at various times to present Progress training courses for version 9 and early version 10. He also presented sessions at three regional Progress conferences in South Africa.

He created Vidisolve in 2011 and started to focus much more actively on the modernisation efforts of his clients. He is currently also busy with his Masters degree in Information Technology at the University of Pretoria.



Introduction

Simon Prinsloo

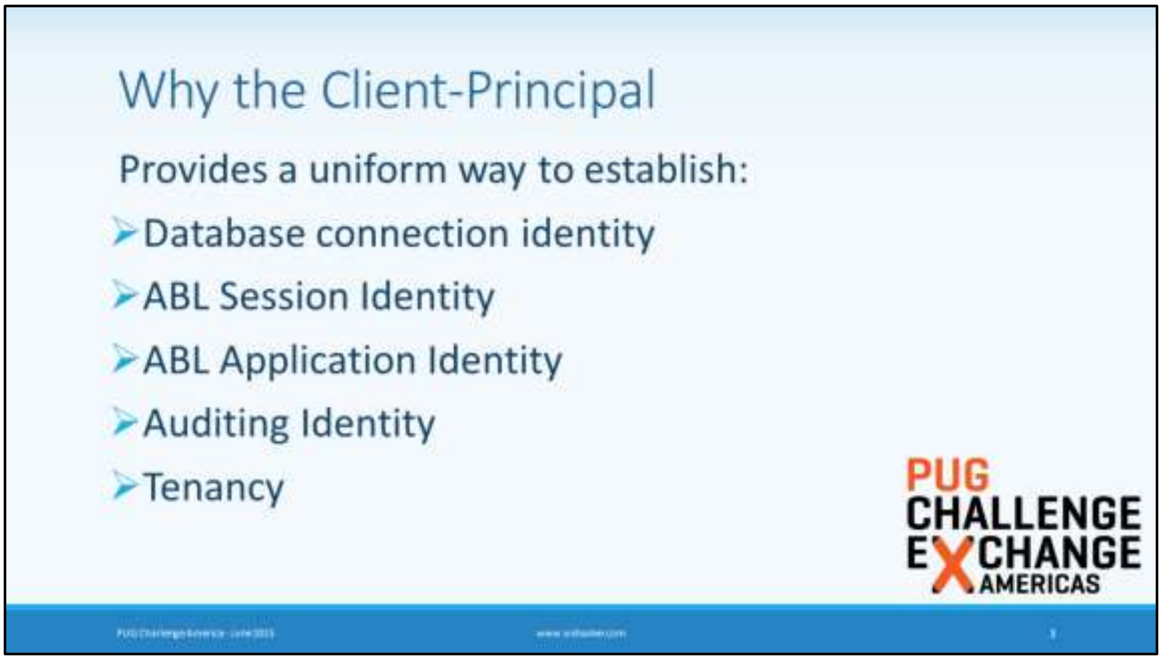
Vidisolve in Pretoria

Working with Progress since v.7 in 1996

Worked on various commercial systems

Mostly focused on CASE tools and implementing new functionality in legacy projects

PUG CHALLENGE EXCHANGE AMERICAS

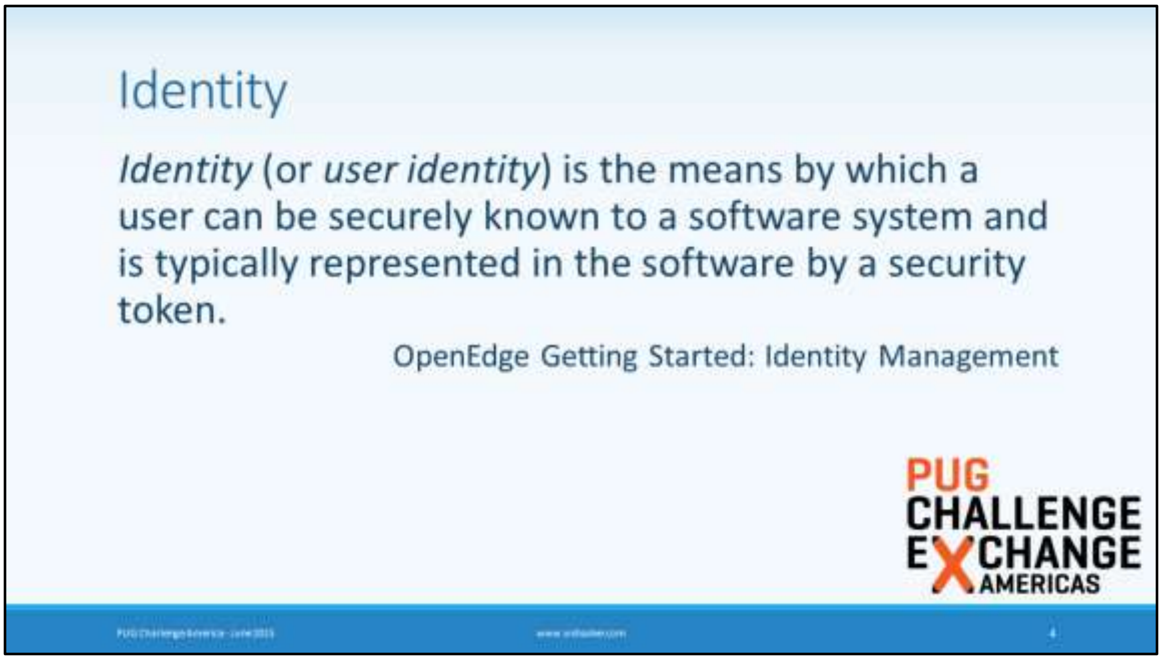


Why the Client-Principal

Provides a uniform way to establish:

- Database connection identity
- ABL Session Identity
- ABL Application Identity
- Auditing Identity
- Tenancy

PUG CHALLENGE EXCHANGE AMERICAS



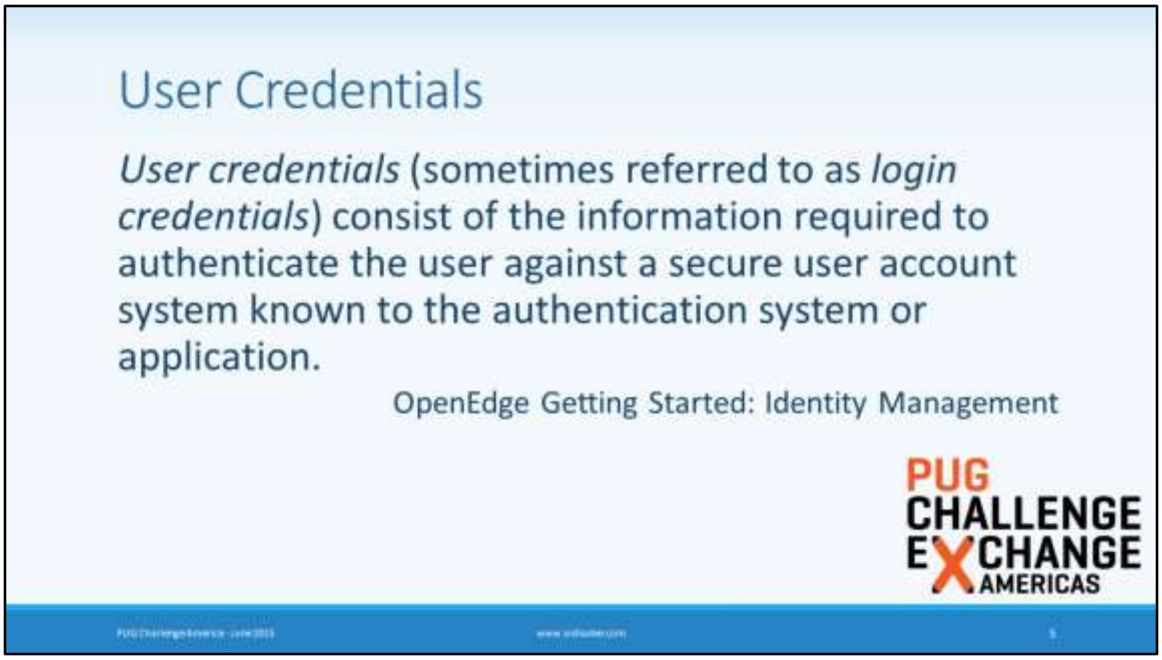
Identity

Identity (or *user identity*) is the means by which a user can be securely known to a software system and is typically represented in the software by a security token.

OpenEdge Getting Started: Identity Management

PUG CHALLENGE EXCHANGE AMERICAS

The security token supplied by the OpenEdge ABL is the CLIENT-PRINCIPAL, a handle based, serializable object.

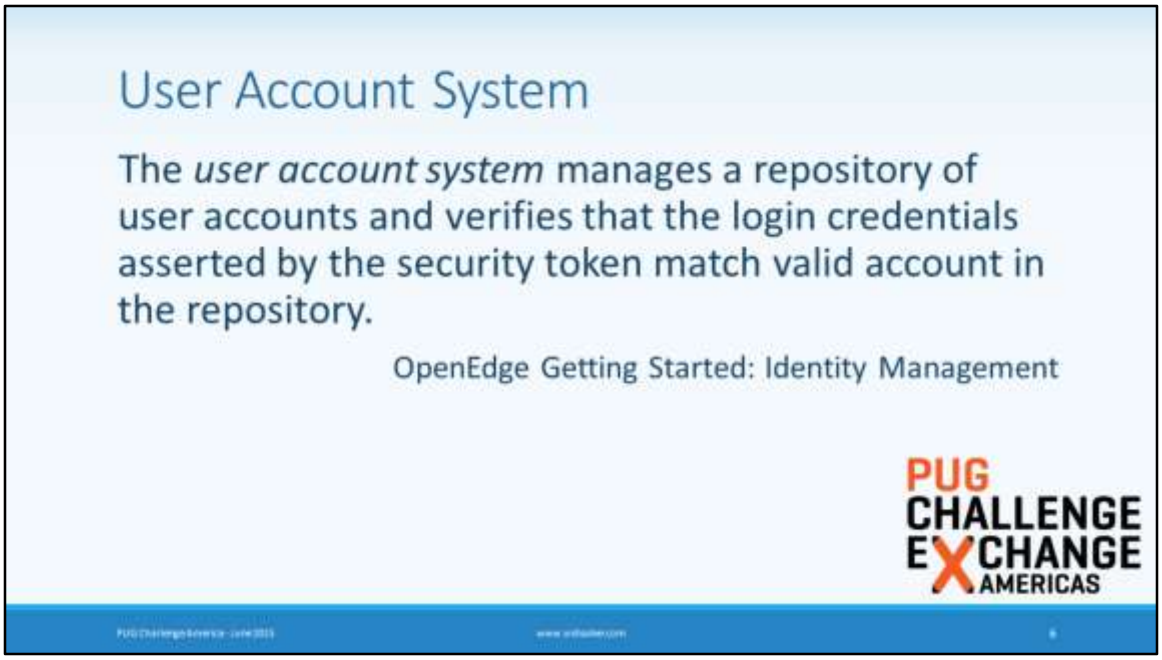


User Credentials

User credentials (sometimes referred to as *login credentials*) consist of the information required to authenticate the user against a secure user account system known to the authentication system or application.

OpenEdge Getting Started: Identity Management

PUG CHALLENGE EXCHANGE AMERICAS

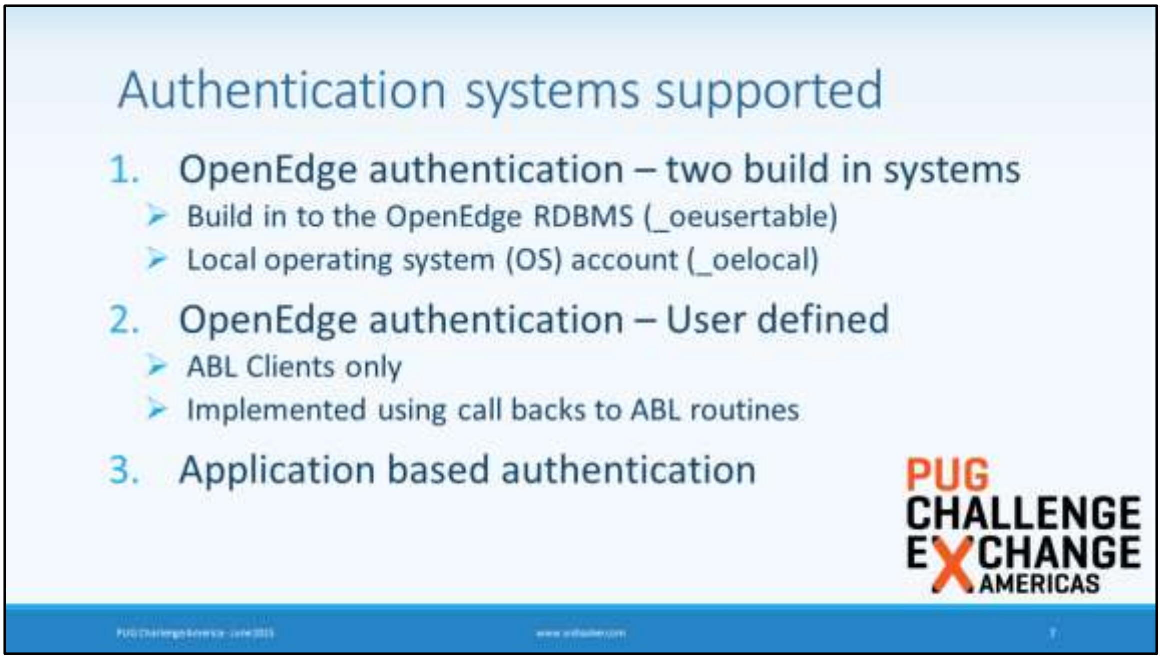


User Account System

The *user account system* manages a repository of user accounts and verifies that the login credentials asserted by the security token match valid account in the repository.

OpenEdge Getting Started: Identity Management

PUG CHALLENGE EXCHANGE AMERICAS

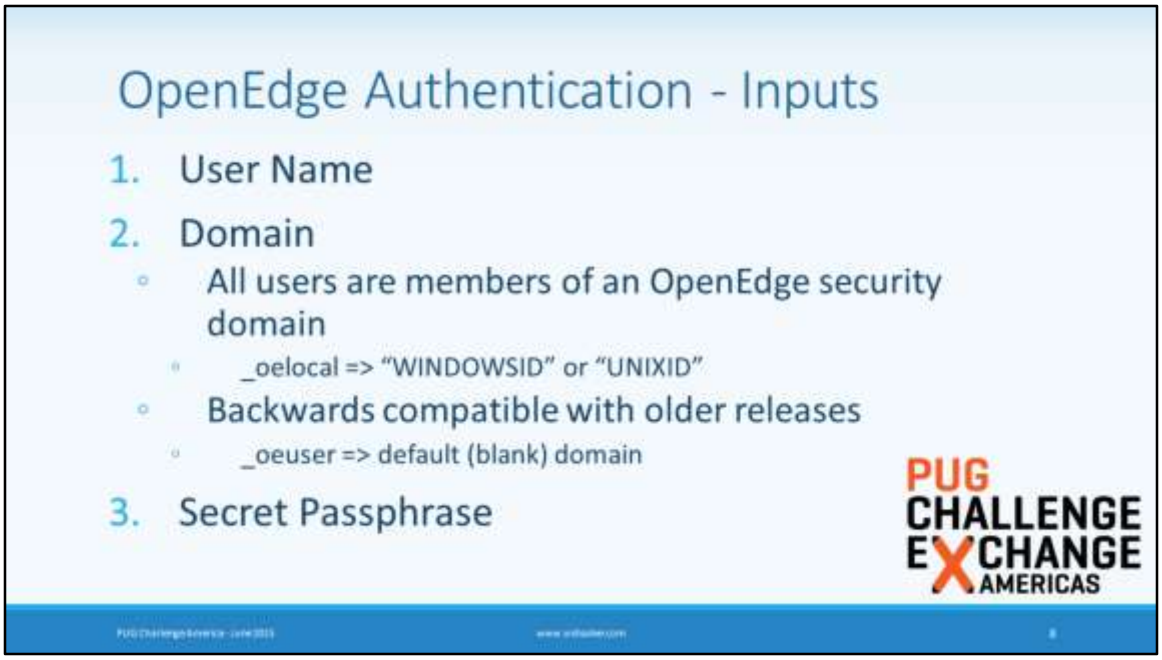


Authentication systems supported

1. OpenEdge authentication – two build in systems
   - Build in to the OpenEdge RDBMS (_oeusertable)
   - Local operating system (OS) account (_oelocal)
2. OpenEdge authentication – User defined
   - ABL Clients only
   - Implemented using call backs to ABL routines
3. Application based authentication

PUG CHALLENGE EXCHANGE AMERICAS

1. Build in domains uses _User or the underlying Unix, Windows or Active Directory.
2. User Defined authentication systems extends the build it authentication system by providing for the configuration of standard ABL-based APIs that will be invoked to establish identity.
3. Application based authentication by-pass the OpenEdge Authentication system completely. This is totally self implemented, but the programmer can still create the CP and use it in the same way as if it was created by one of the previous processes. This is the simplest way of moving a legacy authentication system into the new space.
   1. Note that it is normally extremely easy to create a User Defined implementation of OE Authentication that can replace the legacy login procedure.

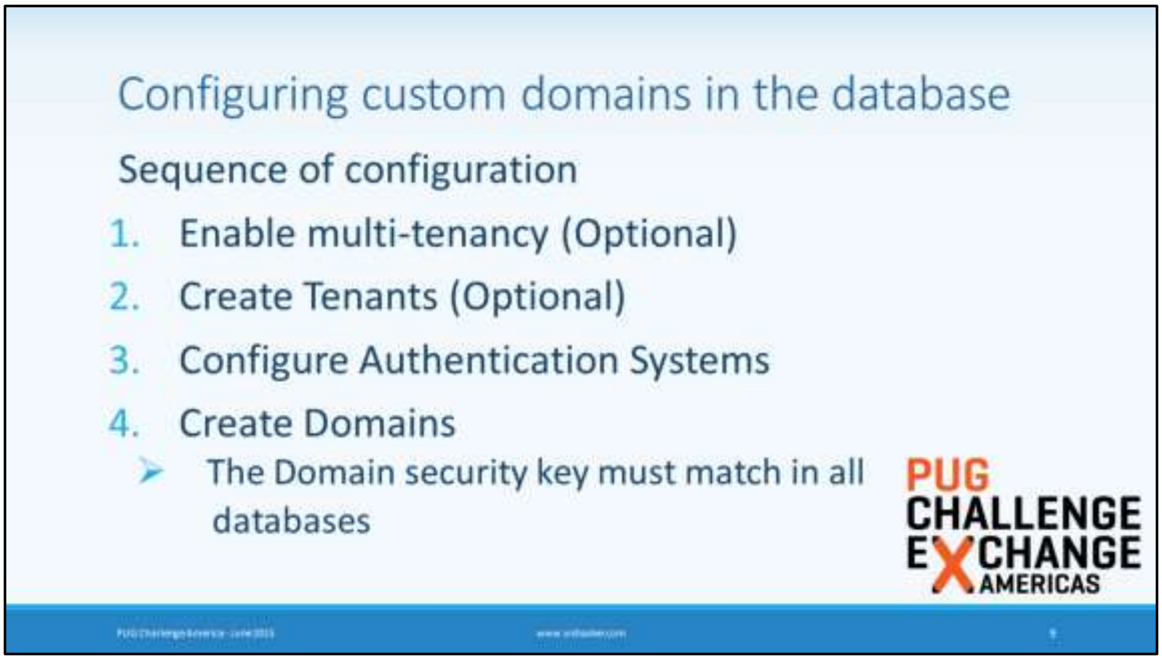


OpenEdge Authentication - Inputs

1. User Name
2. Domain
   - All users are members of an OpenEdge security domain
     - _oelocal => "WINDOWSID" or "UNIXID"
   - Backwards compatible with older releases
     - _oeuser => default (blank) domain
3. Secret Passphrase

PUG
CHALLENGE
EXCHANGE
AMERICAS

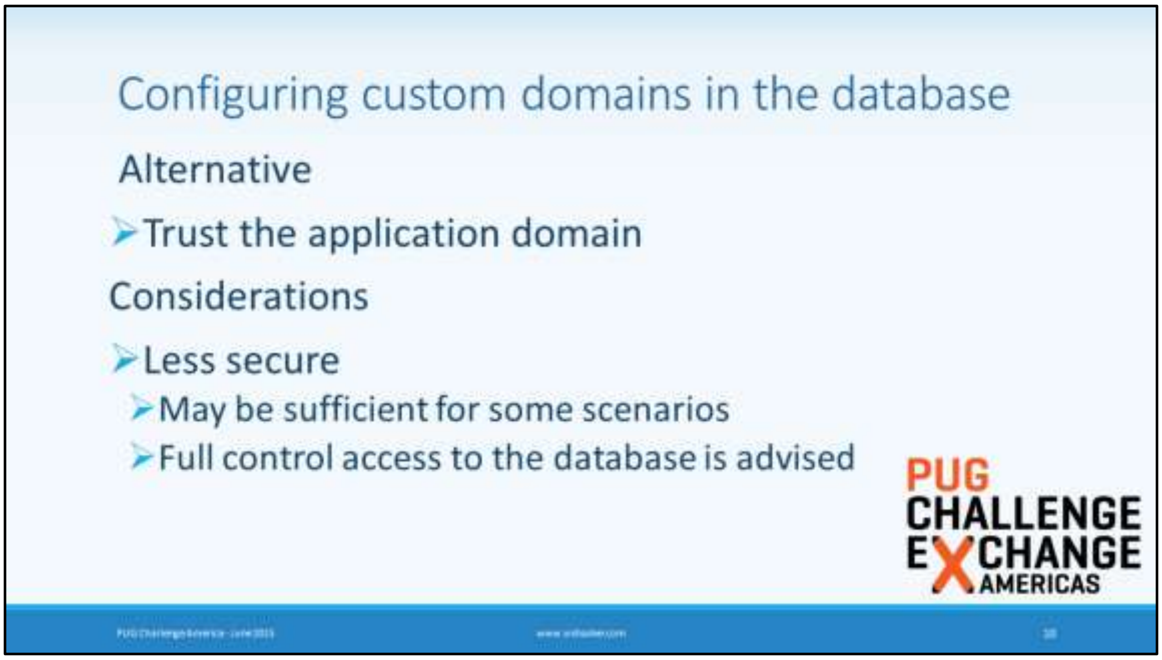
1. Nothing new here.
   1. Well, almost. User name may no longer contain the "@" symbol.
   2. Can (still) be blank.
2. Enables use of multiple different authentication systems.
   1. Similar to a email domain – appears after "@" in the user name string.
   2. Defined in the database.
   3. Same system is used to authenticate all users in the domain.
   4. Has a secret access code used to validate integrity of security tokens.
   5. Can be disabled to prevent authentication of all users in the domain.
3. The good, old, traditional password. By the way – Google "frequent password changes decrease security" and also look for the articles
   1. Adams, Anne and Sasse, Martina Angela. "Users are not the enemy." Communications of the ACM 42.12 (1999): 40-46.
   2. Gehringer, Edward F. "Choosing passwords: security and human factors." Technology and Society, 2002.(ISTAS'02). 2002 International Symposium on. IEEE, 2002.



Configuring custom domains in the database

Sequence of configuration

1. Enable multi-tenancy (Optional)
2. Create Tenants (Optional)
3. Configure Authentication Systems
4. Create Domains
   - The Domain security key must match in all databases



Configuring custom domains in the database

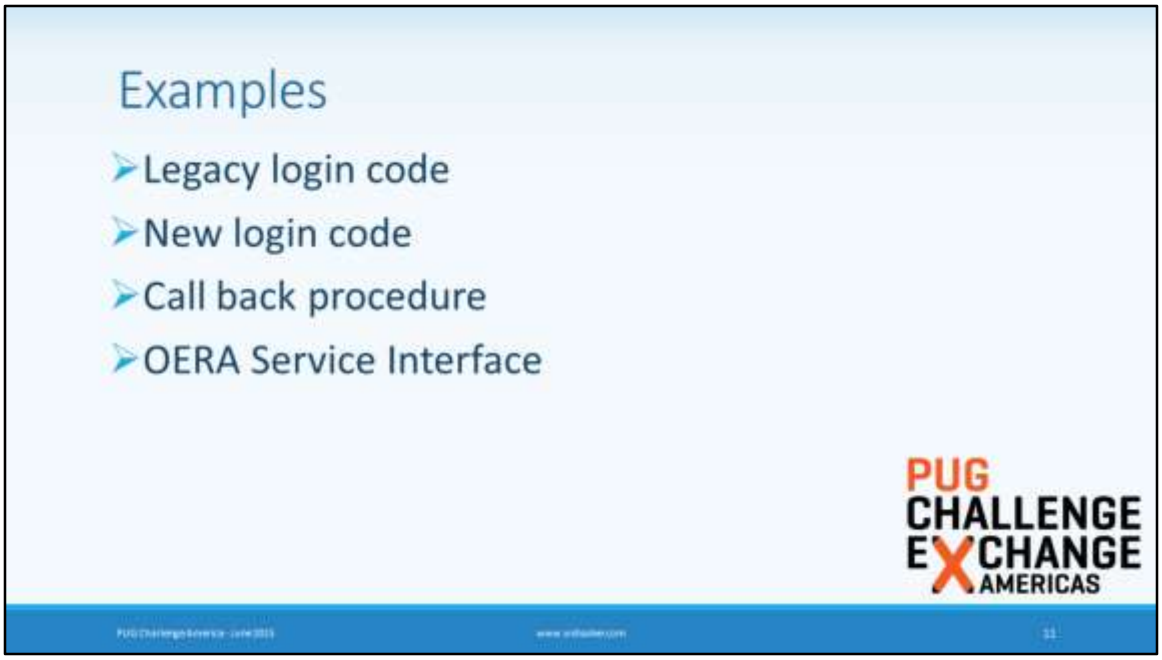
Alternative

- Trust the application domain

Considerations

- Less secure
  - May be sufficient for some scenarios
  - Full control access to the database is advised

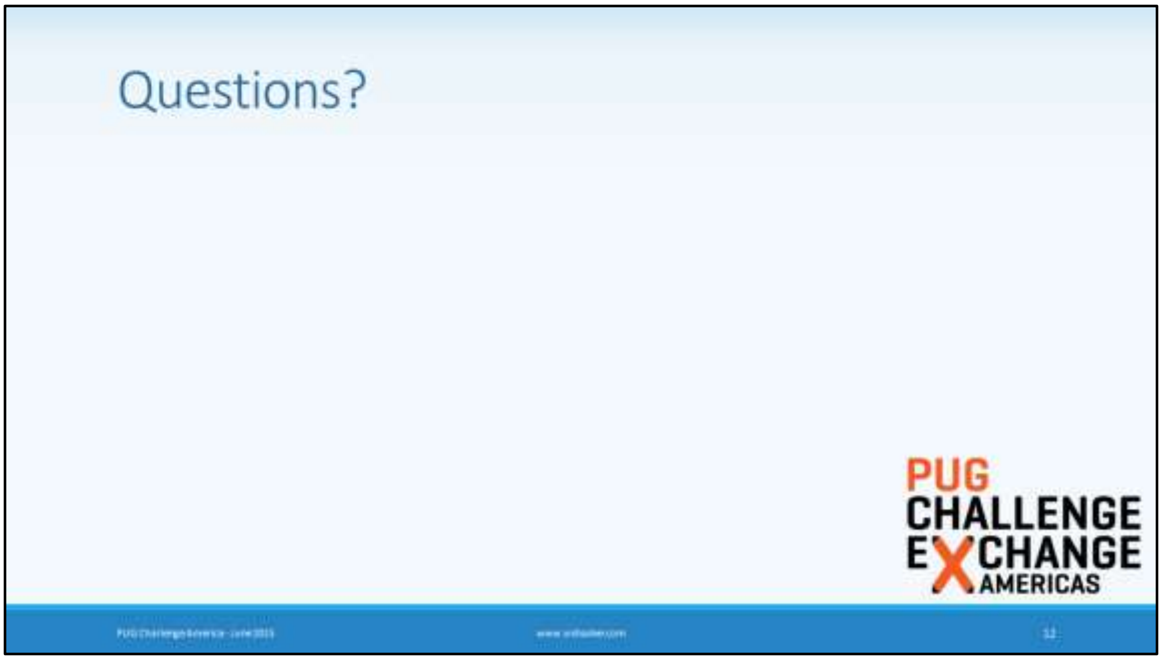
PUG CHALLENGE EXCHANGE AMERICAS



Examples

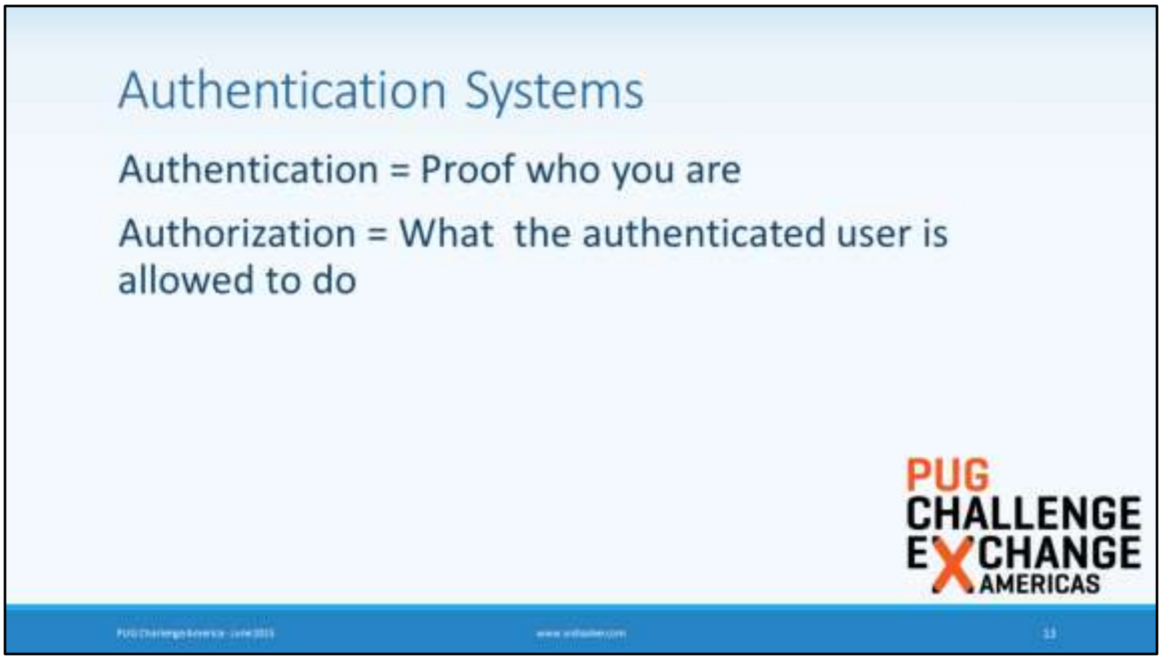
- Legacy login code
- New login code
- Call back procedure
- OERA Service Interface

PUG
CHALLENGE
EXCHANGE
AMERICAS



Questions?

PUG CHALLENGE EXCHANGE AMERICAS



Authentication Systems

Authentication = Proof who you are

Authorization = What the authenticated user is allowed to do