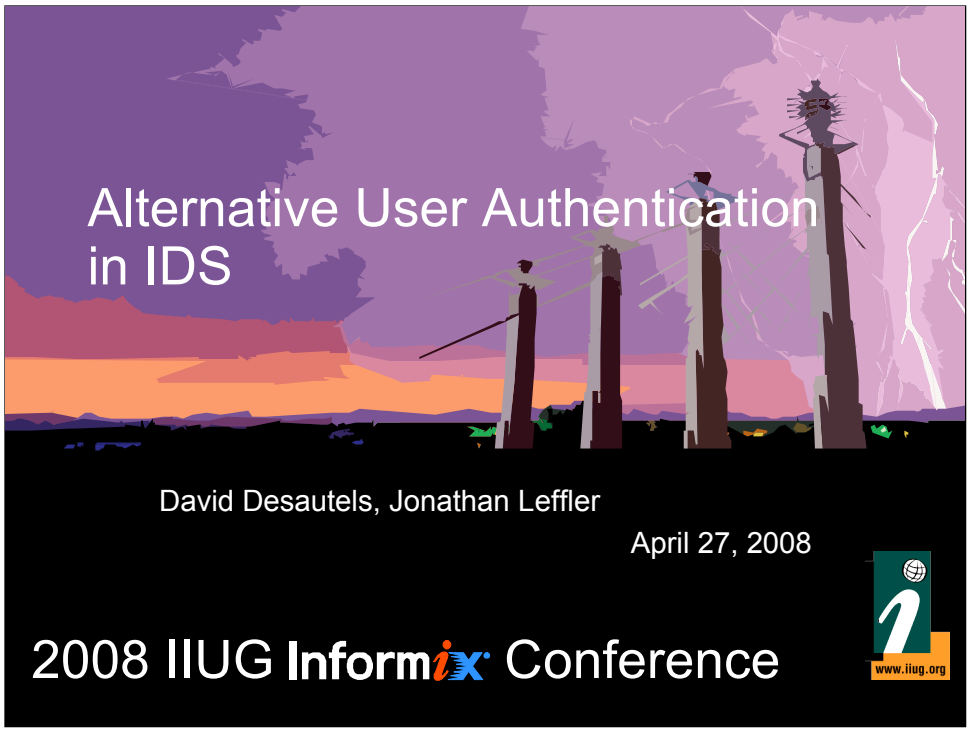


Alternative User Authentication in IDS

David Desautels, Jonathan Leffler

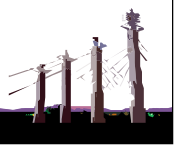
April 27, 2008

2008 IIUG Informix Conference

www.iiug.org

Agenda

- How does IDS do authentication?
- Setup without PAM
- Setup with PAM
- Debugging IDS and PAM configuration problems
- Single Sign-On - SSO



How does IDS do authentication?

- Identification
  - Who are you?
  - getpwnam() – get user info
  - UID, GIDs, home directory, etc.
- Authentication
  - Verification of who you say you are
  - Logon Credential – password
  - getspnam() – get user password
  - Pluggable Authentication Modules - PAM
- IDS only has two choices
  - UNIX password or PAM

Setup without PAM

- **ONCONFIG file has DBSERVERNAME, DBSERVERALIASES**
- **SQLHOSTS file has entry for each name.**
- **Option Settings – 5th column.**

s=0 Disables both hosts.equiv and rhosts lookup from the database server side (only incoming connections with passwords are accepted).

s=1 Enables only the **hosts.equiv** lookup from the database server side.

s=2 Enables only the **rhosts** lookup from the database server side.

s=3 Enables both **hosts.equiv** and **rhosts** lookup on the database server side (server default setting).

r=0 Disables netrc lookup from the client side (no password can be supplied).

r=1 Enables netrc lookup from the client side (default setting for the client side).

r= client side.

Examples

- onconfig

```
SERVERNUM       41          # Instance id
DBSERVERNAME    daveds      # Name of database
  server
DBSERVERALIASES davedsshm,davedspam,davedspamp   #
  List of alternate names
```

- sqlhosts

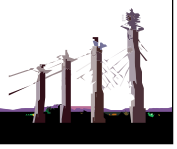
```
daveds      ontlitcp    lx-rama 8741 s=0
davedsshm   onipcshm    xxxxxxx xxxx
davedspamp  ontlitcp    lx-rama 8742
  s=4,pam_serv=login,pamauth=password,csm=spwdcsm
davedspam   ontlitcp    lx-rama 8743
  s=4,pam_serv=ifx_pam,pamauth=challenge
```

UNIX passwd

- IDS uses the UNIX standard getpwnam() and getspnam()
- This does NOT mean that only the local /etc/passwd is used.
- UNIX uses /etc/nsswitch.conf file to control these routines
  - Local passwd/shadow file
  - NIS (and it's variants, NIS+)
  - .rhosts and hosts.equiv
  - LDAP
- What is a "Local User"?
  - getpwnam() does not return NULL

How LDAP Works

- Lightweight Directory Access Protocol
- Protocol – just defines communication
- What is at the other end?
  - Anything that speaks LDAP
- You configure getpwnam(), getspnam() to use LDAP
  - vs. files, NIS, NIS+, etc.
  - /etc/nsswitch.conf file



ldapclient

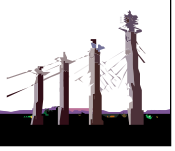
Centrify

- Centrify uses LDAP to communicate with MSAD
- Requires setup of /etc/nsswitch.conf
- IDS requires that getpwnam() NOT return NULL
- Very useful for large organizations
  - Centralizes Authentication Information
  - 1,000's of machines, 100,000's users
  - All controlled from one location

- www.centrify.com

Setup with PAM – Pluggable Authentication Modules

- PAM can handle anything
- Options column s=4
- pam_serv=pam_service_name
  - Linux - /etc/pam.d/pam_service_name
  - UNIX - /etc/pam.conf
    - Entries contain service name
- IDS uses auth and acct_mgmt entries
- 32/64 bit - Be careful if you don't use full path name
  - Linux - /lib/security - /lib64/security
  - Solaris - /lib/security/sparcv9
  - AIX - /lib/security/64
  - HP - /lib/secuity/pa20_64



OS provided PAM modules

- Modules are provided for UNIX passwd authentication
- Kerberos – userid/password
- pam_deny.so
  - Configured for 'other' service
- Many others

http://www.kernel.org/pub/linux/libs/pam/Linux-PAM-html/sag-modulereference.html

pam_debug.so
pam_stack.so

Sample PAM configuration – Solaris pam.conf

```
ifx_pam auth requisite      pam_authtok_get.so.1
ifx_pam auth required       pam_dhkeys.so.1
ifx_pam auth required       pam_unix_auth.so.1
ifx_pam auth required
   /usr/informix/sqldist/lib/pam/pam_daveds.so

ifx_pam account optional
   /usr/informix/sqldist/lib/pam/pam_daveds.so
```

- IDS uses auth and account service
- If no configuration is specified for a given service, PAM uses 'other' service
- If no pathname is specified, lib is located in /lib/security
- Be careful to differentiate for 32 vs 64 bits.
  - `auth  required  /lib/security/$ISA/pam_deny.so`
- Your lib must have suitable permissions for root execution.

```
-rwxr-xr-x   1 root     bin         4256 Mar 30 10:58
   pam_daveds.so
```


You can specify full path but be careful 32/64

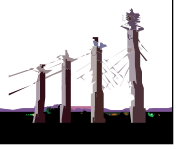
Linux uses $ISA (Instruction Set Architecture)

Solaris looks in sparcv9

HPUX looks in pa20_64

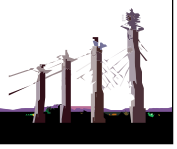
Centrify with PAM

- Centrify also provides PAM module
  - MSAD
  - Kerberos



Sample PAM Configuration - sqlhosts

- `davedspam  ontlitcp lx-rama 8743`
  `s=4,pam_serv=ifx_pam,pamauth=challenge`
- `davedspamp ontlitcp lx-rama 8744`
  `s=4,pam_serv=login,pamauth=password`
- **S=4** indicates PAM
- pam_serv is service name from pam.conf (pam.d)
- pamauth is password or challenge
  - Challenge promises that the pam module can ask for more
  - Password will **never** go back to the user for more
- Be careful of “other” service

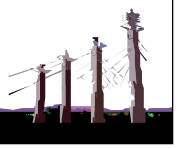


Csm can be added.

How to write your own PAM module

- Shared lib with relocatable code
- Correct permissions/ownership - root 755
- pam_authenticate (auth) will call pam_sm_authenticate
- pam_acct_mgmt (account) will call pam_sm_acct_mgmt
  - _sm_ = service module

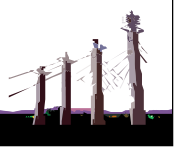
- http://www.kernel.org/pub/linux/libs/pam/Linux-PAM-html/
  - The System Administrators' Guide
  - The Module Writers' Guide
  - The Application Developers' Guide



Processes running as root will not load some libs.

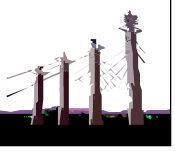
Test programs

- Check UNIX password configuration
- Check PAM configuration



Check UNIX password Configuration

```
/*************************************************************
 * Program to simulate IDS UNIX authentication.
 * Dave Desautels, IBM, 2007
 *
 * Usage: getsec <username>
 *
 *************************************************************/

#include <pwd.h>
#include <stdio.h>
#include <errno.h>
#include <shadow.h>

int
main(int argc, char *argv[])
{
  int retval = 0;
  struct passwd *pw;
  struct spwd *spw;

  pw = getpwnam(argv[1]);
  printf("User: %s  Password: %s\n", pw->pw_name, pw->pw_passwd);

  spw = getspnam(pw->pw_name);
  if (spw)
  {
    printf("Shadow pw: %s\n", spw->sp_pwdp);
  }
  else
  {
    printf("getspnam failed, errno: %d\n", errno);
    retval = 1;
  }

  return retval;
}
```



Check PAM configuration

```
/*****************************************************************
 * Program to simulate IDS PAM Usage.
 * Dave Desautels, IBM, 2007
 *
 * Usage: pam_test <username> <PAMservice>
 *
 *****************************************************************/

#include <stdio.h>
#include <stdlib.h>
#include <strings.h>
#include <pwd.h>
#include <security/pam_appl.h>

#define USER    (argv[1])
#define SERVICE (argv[2])

int conv_func(int num_msg, struct pam_message *msg[],
              struct pam_response **resp, void *appdata_ptr)
{
   int i;
   struct pam_message *msgs = *msg;
   struct pam_response *aresp = calloc(num_msg, sizeof(*aresp));
   char resp_buf[PAM_MAX_RESP_SIZE];

   for (i=0; i<num_msg; i++)
   {
      aresp[i].resp_retcode = 0;
      fputs(msgs[i].msg, stderr);
      fgets(resp_buf, sizeof(resp_buf), stdin);
      resp_buf[strlen(resp_buf)-1] = '\0';
      aresp[i].resp = strdup(resp_buf);
   }
   *resp = aresp;
   return PAM_SUCCESS;
```

Cont.

```
int main(int argc, char *argv[])
{
   int rc = 0, retval = 0;
   struct passwd *pw;
   struct pam_conv conv = {conv_func, NULL};
   pam_handle_t *pamh = NULL;

   pw = getpwnam(USER);
   if (pw)
      printf("Name: %s\nUID: %d\nGID: %d\nHome: %s\n",
        pw->pw_name, pw->pw_uid, pw->pw_gid, pw->pw_dir);
   else
   {
      printf("User %s does not exist.\n", USER);
      rc = 1;
      goto outta_here;
   }

   retval = pam_start(SERVICE, USER, &conv, &pamh);
   if (retval != PAM_SUCCESS)
   {
      printf("Error from pam_start\n%s\n", pam_strerror(pamh, retval));
      rc = retval;
      goto pam_cleanup;
   }

   printf("Pam Handle: 0x%x\n", pamh);

   retval = pam_authenticate(pamh, 0);
   if (retval != PAM_SUCCESS)
   {
      printf("Error from pam_authenticate: %d\n%s\n", retval, pam_strerror(pamh, retval));
      rc = retval;
      goto pam_cleanup;
   }
```

Cont.

```
   retval = pam_acct_mgmt(pamh, 0);
   if (retval != PAM_SUCCESS)
   {
      printf("Error from pam_acct_mgmt: %d\n%s\n", retval, pam_strerror(pamh, retval));
      rc = retval;
      goto pam_cleanup;
   }

if (!rc)
   printf ("User %s is authorized for service %s!\n", USER, SERVICE);

pam_cleanup:
   retval = pam_end(pamh, 0);
   if (retval != PAM_SUCCESS)
   {
      printf("Error from pam_end: %d\n%s\n", retval, pam_strerror(pamh, retval));
      rc = retval;
   }

outta_here:
   return rc;
}
```

Sample PAM Module

```
#include <security/pam_appl.h>
#include <security/pam_modules.h>
#include <stdlib.h>
#include <strings.h>

#define NUM_MSGS 2
struct pam_message msgs[NUM_MSGS] = {
 {PAM_PROMPT_ECHO_OFF, "What is your favorite color? "},
 {PAM_PROMPT_ECHO_OFF, "What is your pet's name? "} };

int
pam_sm_authenticate(pam_handle_t *pamh, int flags, int argc, const char **argv)
{
   int retval;
   struct pam_conv *conv;
   struct pam_message *pam_msg = &msgs[0];
   struct pam_response *pam_resp;
   int i;

   retval = pam_get_item(pamh, PAM_CONV, (void*)&conv);
   if (conv)
   {
      retval = conv->conv(NUM_MSGS, &pam_msg, &pam_resp, conv->appdata_ptr);
   }

   if (retval == PAM_SUCCESS)
   {
      for (i=0; i<NUM_MSGS; i++)
      {
         if(strcmp(pam_resp[i].resp, "reject") ==0)
            retval = PAM_PERM_DENIED;
         free(pam_resp[i].resp);
      }
      free(pam_resp);
   }

   return retval;
}
```

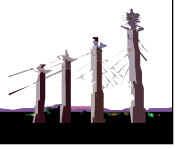
http://www.ibm.com/software/data/informix

Single Sign On In IDS

- SSO is an authentication mechanism which allows users to enter the password once to gain access (authentication) to other resources (computers & software systems)

SSO in IDS

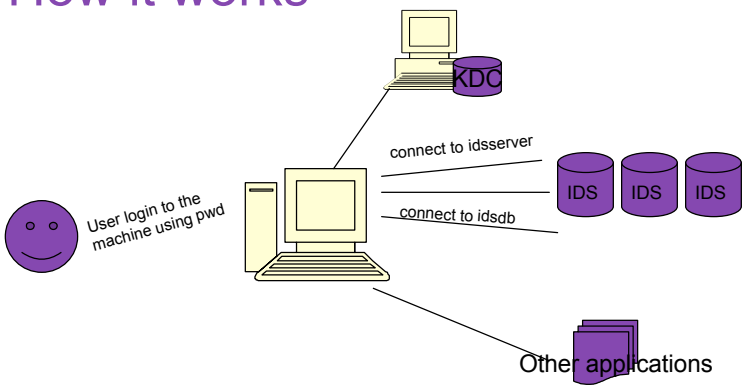
- User enters password during login
- User need not enter password again to authenticate to IDS
- Authentication is invisible to the user
  - New csm module
  - Gets IDS ticket and passes it to server
- IDS uses Kerberos for SSO support



Benefits of SSO

- Easy administration
  - User does not have to reenter password
  - Password management becomes centralized and easier
- SSO is becoming ubiquitous in particular with more web apps. With IDS SSO support IDS easily integrates with existing single sign on infrastructure.

How it works



With SSO the user does not have to enter the password again to authenticate with IDS or any other SSO enabled applications

http://www.ibm.com/software/data/informix

Alternative User Authentication in IDS

David Desautels
daveds@us.ibm.com
Jonathan Leffler
jleffler@us.ibm.com
IBM Information Management