Authentication in Apache HTTP Server 2.1

Justin R. Erenkrantz
University of California, Irvine
justin@erenkrantz.com
Slides: http://www.erenkrantz.com/oscon/

What is 2.1?

- *Current recommended release is 2.0.47*
- *No releases have been done with 2.1 yet*
- *New versioning scheme in force for httpd*
- *Follows Linux versioning scheme*
  - *2.0, 2.2, 2.4 is ‘stable’*
  - *2.1, 2.3, 2.5 is ‘unstable’*

2.1 Policies

- *Commit then review policy on 2.1*
- *Review then commit policy on 2.0*
- *Changes usually first go into 2.1*
- *After review, may be backported to 2.0*
- *Goal is 2.0 series is binary-compatible to 2.0.42 and beyond*

Getting 2.1 or 2.0

```
% CVSROOT=:pserver:anoncvs@cvs.apache.org:/home/cvspublic
% export CVSROOT
% cvs login
<password is anoncvs>
% cvs co -d httpd-2.1 httpd-2.0
OR
% cvs co -r APACHE_2_0_BRANCH httpd-2.0
% cd httpd-2.0/srclib
% cvs co apr apr-util
```

- *2.1 is HEAD in httpd-2.0 CVS*
- *2.0 is APACHE_2_0_BRANCH*

Authentication

- *The three A’s*
- *Authentication, Authorization, Accounting*
- *Authentication - I am this person*
- *Authorization - This person can do this*
- *Accounting - Logging*

Directive Overview

- *AuthName*
  - *Name seen when client prompted*
- *AuthUserFile / AuthDigestFile*
  - *Contains ID and passwords*
- *AuthType*
  - *Type of authentication*

Types of HTTP Authentication

- *Basic*
  - *Base-64 encoded password*
  - *Sent essentially in clear*
- *Digest*
  - *RFC 2069*
  - *Realm, username and password hashed*

Problems Addressed

- *Module names inconsistent*
  - *Hard to maintain and understand*
  - *No separation between authn and authz*
- *mod_auth_dbm had AuthUserFile too*
- *mod_access did authorization*
- *Portion of aaa code was in the core*

Problems Addressed

- *Hard to reuse modules*
- *Massive replication of code*
    - *Cut-and-paste was the recommendation*
- *Lack of digest authentication modules*
    - *Most third-party mod_auth_* do not implement digest!*

Solutions Introduced

- *Modules renamed*
  - *mod_auth_* - Protocol handlers*
    - *mod_auth_basic, mod_auth_digest*
  - *mod_authn_* - Authentication backend*
    - *mod_authn_dbm, mod_authn_file*
  - *mod_authz_* - Authorization*
    - *mod_authz_dbm, mod_authz_host*

How backends separated

- *Provider system introduced*
  - *Originally introduced in mod_dav*
  - *Extension of hooks with versioning*
- *Backend module register methods*
- *Frontend module lookups providers*
  - *May call multiple backends*

1.3/2.0 Configuration

```
<Location /protected>
    AuthName “Server Users”
    AuthType basic
    AuthUserFile conf/users
    require valid-user
</Location>
```

2.1 Configuration

```
<Location /protected>
    AuthName "Server Users"
    AuthType basic
    AuthUserFile conf/users
    require valid-user
</Location>
```

2.1 Extended Configuration

```
<Location /protected>
    AuthName “Server Users”
    AuthBasicProvider file dbm
    AuthUserFile conf/users
    AuthDBMUserFile conf/users-dbm
    require valid-user
</Location>
```

2.1 Digest Configuration

```
<Location /protected>
    AuthName “Server Users”
    AuthType digest
    AuthDigestProvider dbm
    AuthDBMUserFile conf/digest-dbm
    require valid-user
</Location>
```

Writing aaa modules for 2.1

- *Two ways to add modules*
- *New client authentication scheme*
    - *Use the providers*
- *New backend storage*
    - *Register as a provider*

Provider Ordering

- *Auth*Provider directives takes strings*
- *Look up provider at configure-time*
  - *Provider should already be registered*
    - *Done at register_hooks hook*
- *Left-to-right ordering*
  - *First non-DECLINED is returned*

Provider Interface

- *include/ap_provider.h*
- *Group, name, version are the hash keys*

```
apr_status_t ap_register_provider(
                                  apr_pool_t *pool,
                                  const char *provider_group,
                                  const char *provider_name,
                                  const char *provider_version,
                                  const void *provider);
void * ap_lookup_provider(const char *provider_group,
                          const char *provider_name,
                          const char *provider_version);
```

Authentication status codes

- *httpd-2.1/modules/aaa/mod_auth.h*

```
#define AUTHN_PROVIDER_GROUP "authn"
typedef enum {
    AUTH_DENIED,
    AUTH_GRANTED,
    AUTH_USER_FOUND,
    AUTH_USER_NOT_FOUND,
    AUTH_GENERAL_ERROR
} authn_status;

struct authn_provider_list {
    const char *provider_name;
    const authn_provider *provider;
    authn_provider_list *next;
};
```

Authentication Provider

- *modules/aaa/mod_auth.h*

```
typedef struct {
    /* Return AUTH_GRANTED if success */
    authn_status (*check_password)(request_rec *r,
                                   const char *user,
                                   const char *password);

    /* Return AUTH_USER_FOUND if 'user:realm:password’ */
    authn_status (*get_realm_hash)(request_rec *r,
                                   const char *user,
                                   const char *realm,
                                   char **rethash);
} authn_provider;
```

Registering Providers

- *modules/aaa/mod_authn_file.c*

```
static authn_status check_password(...)
static authn_status get_realm_hash(...)
...

static const authn_provider authn_file_provider =
{
    &check_password,
    &get_realm_hash,
};

static void register_hooks(apr_pool_t *p)
{
  ap_register_provider(p, AUTHN_PROVIDER_GROUP,
                       "file", "0", &authn_file_provider);
}
```

Building the Provider List

- *modules/aaa/mod_auth_basic.c*

```
/* Should be in an AP_INIT_ITERATE directive */
...
newp = apr_pcalloc(cmd->pool, sizeof(authn_provider_list));
newp->provider_name = provider_name;

newp->provider = ap_lookup_provider(AUTHN_PROVIDER_GROUP,
                                    newp->provider_name,
                                    "0");

if (newp->provider == NULL ||
    !newp->provider->check_password) {
  return "Unknown or Invalid Authn Provider";
}

/* Add it to the end of the current linked list. */
```

Using the providers

- *modules/aaa/mod_auth_basic.c*

```
authn_status auth_result;
authn_provider_list *current_provider = conf->providers;
do {
 auth_result = current_provider->check_password(r,
                                              sent_user,
                                              sent_pw);
 if (auth_result == AUTH_GRANTED) {
   break;
 }
 current_provider = current_provider->next;
} while (current_provider);
if (auth_result != AUTH_GRANTED) {
  /* Something bad happened */
  return HTTP_UNAUTHORIZED;
}
return OK;
```

Writing a Provider

```
static authn_status check_password(request_rec *r,
                                   const char *user,
                                   const char *password) {
  status = ap_pcfg_openfile(&f, r->pool, conf->pwfile);
  while (!(ap_cfg_getline(l, MAX_STRING_LEN, f))) {
    w = ap_getword(r->pool, &rpw, ':');
    if (!strcmp(user, w)) {
      file_password = ap_getword(r->pool, &rpw, ':');
      break;
    }
  }
  ap_cfg_closefile(f);
  if (!file_password) {
    return AUTH_USER_NOT_FOUND;
  }
  status = apr_password_validate(password, file_password);
  if (status != APR_SUCCESS) {
    return AUTH_DENIED;
  }
  return AUTH_GRANTED;
}
```

Conclusions

- *No major changes in hooks*
  - *Core aaa modules rewritten*
- *Not forced to rewrite aaa modules for 2.1*
  - *Allow reduction of code*
  - *Increase of administrator flexibility*
- *Authorization may see similar changes*