Inkl. CD!

Ausgabe **2.09** Februar | März www.**phpmagazin**.de S&S

PHPmagazin

Deutschland 9,80 € Österreich 10,80 € | Schweiz 19,20 SFr
Niederlande 11,25 € | Luxemburg 11,25 €

PHPmagazin

Twitter-API
Einsatz und Verwendung in PHP-Anwendungen

Solar Framework
Konkurrenz zu Symfony, CakePHP & Co.?

Skripting mit Dojo
Einführung in das Framework und der Einsatz in Zend

WordPress 2.7
Persönlicher, schicker und komfortabler bloggen

IPC-KEYNOTE „TO DRIZZLE MYSQL“
Bonus Video
von Brian Aker als Video in voller Länge (45 Minuten)

NetBeans 6.5
Das aktuelle Release der IDE glänzt mit Unterstützung für PHP

MAGENTO 1.1.8
Das Open-Source-Shopsystem, mit iPhone-Support und umfangreichen Management-Features

JavaScript
Mit jQuery das Skripting vereinfachen

WebDAV
Unabhängig vom Server: WebDAV für eigene Anwendungen

Message Queues
Für freie Systeme und Tasks: Aufgaben per Message Queue verteilen

Bonus Video
international PHP2008 conference

Datenträger enthält Info- und Lehrprogramme gemäß §14 JuSchG

Außerdem mit auf der CD
Die wichtigsten PHP-Frameworks von A bis Z, WordPress 2.7, TYPO3, Joomla!, Drupal und vieles mehr!

4 195649 609809 02

Mit PHP gegen ein LDAP-Verzeichnis authentifizieren

Zugriffsmonopol

Wenn jede Software im Unternehmen seine eigene Authentifizierung implementiert, werden die Sysadmins bald Ihren Job kündigen. Deshalb ist es eine gute Idee, unserer PHP-Applikation beizubringen, mit LDAP gegen eine zentrale Benutzerdatenbank zu authentifizieren.

▶ von Thomas Pfeiffer und Andreas Wenk

Oftmals kommt es ja anders als man denkt. Da hat man nun seine Applikation entwickelt und mit einem eigenen Authentifizierungssystem versehen. Jetzt kommt der erste potenzielle Kunde um die Ecke und möchte Ihre Applikation in seine bestehende Verzeichnisstruktur integrieren. Verständlich ist das schon, denn gerade in großen Unternehmen werden Benutzer in der Regel zentral verwaltet. Von einem Unternehmen mit vielleicht mehreren hundert Mitarbeitern wird man kaum erwarten können, die Zugangsdaten der Mitarbeiter in den jeweiligen Applikationen zu pflegen. Typischerweise wird man hier z. B. auf Produkte wie den Active-Directory-(ADS-) oder Lotus-Notes-Server (NDS) oder aber auch einen OpenLDAP-Server treffen. Netterweise kann uns am Ende jedoch völlig egal sein, welches Produkt eingesetzt wird, denn gemeinsam sprechen sie alle LDAP.

Installation von php5-ldap

Dieses so genannte *Lightweight Directory Access Protocol* (LDAP) ist im Prinzip nichts weiter als eine Möglichkeit, einen Datenspeicher für hierarchisch geordnete Informationen abzufragen. Oft wird hier auch der Begriff „Verzeichnisdienst" verwendet. Wir wollen uns aber hier jetzt nicht mit LDAP als solchem auseinander setzen, sondern betrachten einen entsprechenden Server einmal als gegebene Herausforderung, der wir uns jetzt stellen wollen. Als Erstes müssen wir unser PHP also um die Fähigkeit erweitern, mit LDAP-Servern zu sprechen. Besitzer einer Debian-basierten Distribution sind jetzt fein raus. Diese tippen einfach

```
apt-get install php5-ldap
```

starten den Webserver neu und haben damit das Thema der Installation vom Tisch. Ist alles glatt gegangen, wird dies durch den entsprechenden LDAP-Abschnitt in der Ausgabe von *phpinfo()* quittiert (Abb. 1).

Windows-User, die z. B. XAMPP einsetzen, genügt in der Regel das Entfernen des Kommentarzeichens in der Datei *php.ini* (z. B. in *c:\xampp\apache\bin*), um die entsprechende Erweiterung zu installieren:

```
extension=php_ldap.dll
```

Auch hier muss anschließend der Webserver neu gestartet werden. Benutzer, die andere Linux-Betriebssysteme wie Suse oder RedHat Ihr Eigen nennen, installieren die Extension analog mit dem jeweils integrierten Paketmanager. Und zu guter Letzt kann die Extension natürlich bei der Kompilierung von PHP auch mit

```
./configure --with-ldap
```

integriert werden.

Mit dem LDAP-Verzeichnis verbinden

Wie bereits erwähnt, werden Informationen hierarchisch in einem LDAP-Server organisiert. Abbildung 2 zeigt eine

ldap

LDAP Support	enabled
RCS Version	$Id: ldap.c,v 1.161.2.3.2.11 2007/07/17 09:09:42 jani Exp $
Total Links	0/unlimited
API Version	3001
Vendor Name	OpenLDAP
Vendor Version	20407
SASL Support	Enabled

Abb. 1: phpinfo() belegt es: Die LDAP-Unterstützung steht bereit

Beispielstruktur eines Active-Directory-Servers. Hier wird die Domäne *eunique.de* verwaltet.

Eine Organisationseinheit (OU = Organizational Unit) innerhalb dieser Domäne ist die Firma NMMN. Innerhalb dieser Organisationseinheit gibt es dann Abteilungen (wiederum Organisationseinheiten), denen die einzelnen Mitarbeiter zugeordnet sind. Gehen wir nun davon aus, dass der LDAP-Server neelix heißt und auf dem Port 389 lauscht. Sie können dann das Codeschnipsel aus Listing 1 verwenden, um sich mit dem Server als User *tp* zu verbinden.

Die Funktion *ldap_connect* baut hierbei noch nicht die Verbindung auf, wie man vermuten könnte [1]. Dies geschieht erst durch die Funktion *ldap_bind*, der wir die Kennung der Verbindung sowie einen Usernamen und ein Passwort übergeben, um uns gegenüber dem LDAP-Server auszuweisen. In der Regel muss hier der Username (*tp*) um die Domaine (*eunique.de*) ergänzt werden. Klappt *ldap_bind*, haben Sie bereits erfolgreich eine Authentifizierung gegen einen LDAP-Server durchgeführt. Es versteht sich von selbst, dass eine bestehende Verbindung zum LDAP-Server mit *ldap_unbind* wieder getrennt wird.

Einzelne Einträge aus dem LDAP-Verzeichnis holen

Jetzt wollen wir allerdings noch ein paar weitere Informationen aus dem Verzeichnis erhalten. Dazu verwenden wir den Inhalt aus Listing 2.

In *$attributes* legen wir alle Felder ab, die wir als Ergebnis erhalten wollen. Möchten Sie erst einmal alles sehen was an Informationen im Verzeichnis verfügbar ist, lassen Sie das Array *$attributes* als dritten Parameter beim Aufruf von *ldap_search* einfach weg. Als Nächstes legen wir unsere Suchbedingung fest. Hier fragen wir nach dem (eindeutigen) Feld *samaccountname*, in dem der Username (diesmal ohne Domäne) abgelegt sein soll-

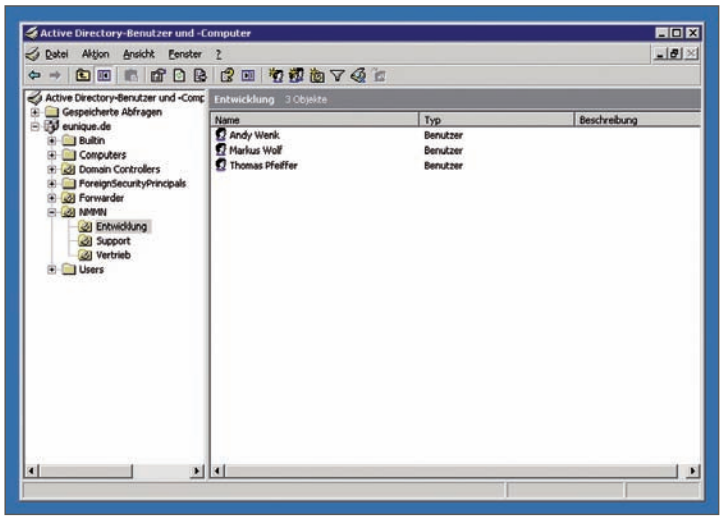

Abb. 2: Beispielstruktur eines Active-Directory-Servers

te. Denkbar wäre natürlich auch, hier nach anderen Feldern zu fragen, z. B. nach dem Nachnamen. Da würde der Suchstring folgendermaßen lauten:

```
$search = 'sn=' . $user;
```

wobei *sn* dem Feldnamen für den Nachnamen (*surname*) entspricht. Ist alles glatt gegangen, liefert uns die Funktion *ldap_search* eine Ergebniskennung und *FALSE* im Fehlerfall.

Im LDAP-Verzeichnis suchen

Um dann die eigentliche Suche durchzuführen, verwenden wir die Funktion *ldap_search*. Dieser müssen wir neben der Verbindung auch noch die Basis für das Verzeichnis mitgeben. In unserem Fall ist das der String *'OU=NMMN,DC=eunique,DC=de'*, um alles unterhalb der Organisationseinheit NMMN abzusuchen. Die Angabe DC steht dabei für die jeweiligen Domain Components, in diesem Fall also für *eunique.de*. Das Ergebnis der Suche können wir dann mit der Funktion *ldap_get_entries* abrufen. Wenn auch hier alles glatt gegangen ist, liefert uns die Funktion als Ergebnis ein Array zurück. Dieses dürfte dann in etwa folgendermaßen aussehen:

```
Array
(
  [count] => 1
  [0] => Array
    (
      [sn] => Array
        (
          [count] => 1
          [0] => Pfeiffer
        )

      [0] => sn
      [givenname] => Array
        (
          [count] => 1
          [0] => Thomas
        )
    ...

)
```

Es sollte kein Problem sein, über das Ergebnis-Array zu iterieren und die Daten darzustellen.

LDAP in PHP5

LDAP wird in PHP bei Weitem nicht stiefmütterlich behandelt. Unter [2] finden Sie eine Auflistung aller vorhandenen LDAP-Funktionen, die in PHP integriert sind. Wir haben in diesem Artikel die Funktionen *ldap_connect* und *ldap_bind* genutzt, um eine Verbindung zum LDAP-Server herzustellen. Für die Suche und Auswahl von Daten haben wir *ldap_search* und *ldap_get_entries* eingesetzt. Das sind die rudimentären Funktionen, um überhaupt mit dem Server zu sprechen. Sie können das LDAP-Verzeichnis natürlich auch verändern, indem Sie mit *ldap_add* Daten hinzufügen oder mit *ldap_delete* Daten löschen. Natürlich unter der Voraussetzung, dass Sie dazu berechtigt sind.

Fazit

Wir haben Ihnen gezeigt, wie Sie mit ein paar wenigen Zeilen Code eine Authentifizierung gegen ein LDAP-Verzeichnis durchführen können. Unter Umständen und je nach Art der Applikation können Sie dadurch sogar auf eine eigene Userverwaltung verzichten. Es ist allerdings auch denkbar, verschiedene Authentifizierungsmechanismen parallel einzusetzen. Damit erreichen Sie eine große Flexibilität und können die Authentifizierung je nach Anforderung implementieren. Den deutschen Wiki-Eintrag zu LDAP und weiterführende Links erreichen Sie unter [3]. Den freien LDAP-Server OpenLDAP finden Sie unter [4].

LISTING 1

Der User „tp" verbindet sich mit dem neelix-Server

```
$cfg = array( 'ldap_host' => 'neelix',
   'ldap_port' => '389',
'ldap_base_dn' => 'OU=NMMN,DC=eunique,DC=de',
'ldap_domain' => 'eunique.de',
   );

 $user = 'tp';
 $pass = 'geheim';
$user_str = $user . '@' . $cfg['ldap_domain'];

$con = ldap_connect($cfg['ldap_host'], $cfg['
                                        ldap_port'])
   or die ("ldap_connect fehlgeschlagen!");

 $ldapbind = ldap_bind($con, $user_str, $pass)
      or die ("ldap_connect fehlgeschlagen!");


if ($ldapbind) {
 print "Verbindung zum LDAP Server {$cfg['ldap_host'
                                    ]} erfolgreich\n";
} else {
 print "Verbindung mit {$cfg['ldap_host']}
                                fehlgeschlagen\n";
}
```

LISTING 2

Zugriff auf weitere Verzeichnisinformationen

```
$attributes = array('sn', 'givenname', 'mail',
                                'telephonenumber');
$search = 'samaccountname=' . $user;
$result = ldap_search($con, $cfg['ldap_base_dn'],
$search, $attributes);
if ($result) {
$info = ldap_get_entries($con, $result);
}
```

Links & Literatur

[1] http://bugs.php.net/bug.php?id=15637

[2] http://de3.php.net/ldap

[3] http://de.wikipedia.org/wiki/Ldap

[4] http://www.openldap.org/

[5] Active Directory: http://de.wikipedia.org/wiki/Active_Directory

[6] Novell NDS: http://de.wikipedia.org/wiki/Novell_Directory_Services

Thomas Pfeiffer & Andreas Wenk

Thomas Pfeiffer und **Andreas Wenk** sind Applikationsentwickler bei der NMMN – New Media Markets & Networks GmbH in Hamburg. Dort entwickeln sie das Ressourcen-Management-System eUNIQUE unter heftigem Einsatz von PHP und der PostgreSQL. Sie erreichen die beiden unter tp@nmmn.com und aw@nmmn.com oder über die Webpräsenz http://www.e-unique.com.