

Sisis-cgi

Letzte Aktualisierung Dienstag, 2. Oktober 2007

Im Bibliothekssystem Sisis von OCLC Pica werden Benutzerdaten gespeichert. Diese Benutzerdaten kann man zur Authentifizierung an anderen Dienstleistungen in der Bibliothek nutzen. Dieses Softwareprojekt bietet eine einfache Möglichkeit zur Authentifizierung von Benutzern an einem Sisis-System und zur Recherche von Benutzerdaten für Bibliotheksmitarbeiter.

Worum geht's?

Im Bibliothekssystem Sisis von OCLC Pica gibt es das Protokoll SLNP bzw XSLNP zur Verbindung externer Dienste mit der Datenbank. Primär wahrscheinlich entwickelt, damit die eigenen Clients sich mit der Datenbank verbinden können wurde und wird das Protokoll SLNP an vielen Stellen auch für andere Dinge wie zum Beispiel Authentifizierung von Benutzern eingesetzt. Jedoch enthält zumindest SLNP selbst keine abgestufte Authentifizierung. Einmal angemeldet kann ein Nutzer einer SLNP-Session einiges unternehmen, was nicht erwünscht ist. Dieses Manko wurde nach Aussage des Herstellers bei XSLNP behoben, hier soll man einzelne XSLNP-User für einzelne Dienste freischalten können. Leider ist XSLNP ein lizenzierungspflichtiges Protokoll, also kostenintensiv. Außerdem ist es natürlich proprietär.

Auf der anderen Seite gibt es in einer Bibliothek einige Dienste, die nur Benutzern dieser Bibliothek zur Verfügung stehen sollen. Dies ist zum Beispiel die Nutzung der öffentlichen Internet-PCs außerhalb der von der Bibliothek definierten Positivliste sowie die Anmeldung der Benutzer mit eigenem Notebook im Netzwerk der Bibliothek (Lesesaal). Natürlich soll es dafür nicht eine eigene Accountverwaltung geben sondern die Benutzer sollen sich sinnvollerweise mit ihren bekannten Bibliotheksausweisdaten an solchen Systemen anmelden können.

Die Lösung

Die hier beschriebene Lösung besteht aus zwei Teilen. Der erste Teil ist ein Perlscript, welches auf dem Sisisserver läuft und über eine definierte Schnittstelle passwortgeschützt Daten aus dem Sisisystem zur Verfügung stellt. Der Zugriff dieses Scriptes auf Sisis geschieht direkt per SQL, um hier weitere proprietäre und ggf kostenpflichtige Protokolle zu vermeiden.

Die so zur Verfügung gestellte Schnittstelle kann von beliebigen Diensten genutzt werden. Im Beispiel zeige ich hier die Integration in einen Proxyserver auf der Basis von squid, der für die öffentlichen PCs den Zugang zum Internet regelt. In Squid kann man ein beliebiges externes Programm zur Authentifizierung verwenden. Die Anforderungen von squid an ein solches Programm sind:

- Das Programm darf sich nicht selbst beenden, muss also in einer Endlosschleife laufen
- Das Programm muss zu authentifizierende Paare Benutzer-id Passwort durch ein Leerzeichen getrennt und mit abschließendem CR aus STDIN empfangen.
- Die Rückgabe einer solchen Eingabe muss mit OK (alles klar) oder ERR (Fehler) beginnen.

Die über das Perlprogramm auf dem Sissserver zur Verfügung gestellte Schnittstelle kann aber auch einfach Benutzerrecherchen bieten. Dazu gibt es ein Webformular, welches auf einem beliebigen Server abgelegt werden kann und solche Recherchen erlaubt. Klar ist, dass dieses Formular passwortgeschützt werden muss.

Weitere Funktionen, die über das Script auf dem Sissserver zur Verfügung gestellt werden können, sind denkbar, das Script ist erweiterbar. Auch schreibende Funktionen (bisher nur lesend) sind kein Problem, wobei man sich dabei im Klaren sein muss, dass unter Umständen die Integrität der Datenbank leicht gefährdet sein kann. Ein logischer Schritt wäre übrigens auch die Umsetzung einer ldap-Schnittstelle, die dann standardisiert Daten zur Verfügung stellt und entgegen nimmt.

Bei uns in Dortmund wird das Script zur Zeit für drei Anwendungsgebiete eingesetzt: Die erwähnte Authentifizierung der Internet-PCs über einen Proxy, die Authentifizierung der Benutzernotebooks im Lesesaal bei Netzwerkbenutzung (hier nicht weiter erläutert) sowie die Recherche nach Benutzerdaten durch UB-Mitarbeiter.

Die einmal geholten Benutzernummer-Passwort-Kombinationen werden übrigens im Script nicht gecached, da bereits squid mit der unten angegebenen Konfiguration ein Caching vornimmt. Des weiteren hält weder das Perlscript, welches von squid gestartet wird, noch das Perlscript auf dem Sissserver eine Verbindung länger als notwendig offen. Der Ressourcenverbrauch für eine offen gehaltene Verbindung schien mir schwerer wiegend als der Zeitverbrauch durch das neue Öffnen einer Verbindung.

Installation

- Kopieren des sisiscgi (s.u.) nach /usr/local/sisis-pap/wwwdir/cgi-bin auf dem Sissserver (oder wo auch immer das cgi-bin-Verzeichnis des Sisservers sich befindet).

- Anpassen des Scriptes. Das \$accesspassword muss eingetragen werden. Ich empfehle, die md5sum über eine beliebige Datei als Passwort zu verwenden, das ist hinreichend komplex. Beispiel: md5sum /var/log/messages

Ausserdem muss unten im Script pro Server, auf dem das Script läuft, das Sisis-Sybase-Passwort (Variable \$dbpassword) und der passende Servername (if (\$servername eq "...") angegeben werden. So ist es möglich, das Script ohne weitere Änderungen auf dem Testserver ebenfalls einzusetzen.

- Da wahrscheinlich in anderen Bibliotheken Minderjährigen in anderen Benutzergruppen als 41 eingeordnet werden und die Einschränkung, dass sich Mitglieder von Gruppen > 79 nicht anmelden dürfen, nicht existiert, ist der select-Befehl im Script anzupassen. Im einfachsten Fall (keine weitere Einschränkung) wird aus

```
select d02bnr,d02opacpin,d02sp1 from d02ben where d02bnr=? and d02bg<80 and d02bg !=41
```

dann

```
select d02bnr,d02opacpin,d02sp1 from d02ben where d02bnr=?
```

- Aufbau einer Webseite, die den Code recherche.html-torso.txt (s.u.) enthält, auf einem beliebigen Webserver. Innerhalb dieses Torsos ist anstelle von meinaccesspassword besagtes Passwort einzutragen (kommt mehrfach vor!) sowie die URL des Sisservers bei form action="http://meinsissserver.ub.un....

- Test des Scriptes anhand der verschiedenen Eingabemöglichkeiten im Formular. Wenn alles geht, kann squid umgestellt werden:

- Das Script proxysisauth.pl wird auf den Server kopiert, auf dem der Proxy läuft. Bei uns liegt es in /etc/squid

- Das Script wird angepasst. Mindestens die Variablen \$server und \$accesspassword im Kopf des Scriptes müssen entsprechend verändert werden.

- Das Script wird getestet: Aufruf an der Kommandozeile (es gibt keine erkennbare Reaktion, das Shellprompt erscheint

nicht mehr), Eingabe einer Benutzernummer mit zugehörigem Passwort in einer Zeile durch Leerzeichen getrennt, anschließend Enter, muss die Ausgabe von OK in einer eigenen Zeile ergeben. Das Programm muss anschließend mit STRG-C beendet werden.

- Ist alles korrekt, wird in der squid.conf eingetragen, dass squid dieses Programm zur Authentifizierung verwenden soll:

```
auth_param basic program /etc/squid/proxycisauth.pl
```

```
auth_param basic children 5
```

```
auth_param basic realm external URL/externe URL
```

```
auth_param basic credentialsttl 2 hours
```

Dabei ist wichtig, dass es in der squid.conf nur einmal den Parameter auth_param basic program geben darf.

Es werden so nach einem Neustart von squid fünf Instanzen des Perlscriptes gestartet. Die Ergebnisse der Abfragen werden von squid zwei Stunden gespeichert. Dies ist wichtig zu wissen, wenn Benutzer mit soeben geänderten Passwörtern reklamieren, daß sie nicht "ins Internet kommen". Eine zu kurze Zeit für die Pufferung belastet allerdings den Sissserver wieder relativ stark. Hier ist ein persönlicher Mittelweg gefragt.

Bekannte Probleme/Fehler

- Die Übertragung der Daten zwischen den Servern (dem Proxyserver und dem Sissserver) geschieht in der derzeitigen Fassung unverschlüsselt. Das sollte nichts ausmachen, sofern die beiden Server in einem eigenen Netzwerk stehen oder zumindest an einem eigenen Switch hängen. Besser wäre es, die Kommunikation demnächst mal auf https umzustellen. Übrigens: Auch SLNP und XSLNP arbeiten unverschlüsselt oder mit einer proprietären Verschlüsselung nach dem sehr fragwürdigen Motto Security by obscurity . Aber das muss man ja nicht als schlechtes Beispiel für alle Tage annehmen.

Historie

- 9/07: Erste Version veröffentlicht

Download

Diese Programme sind kostenfrei im nichtkommerziellen Umfeld verwendbar. Das Programm darf ohne meine Einwilligung nicht kommerziell verwendet werden. Dies gilt insbesondere auch für Dienstleistungen, die die Installation oder Anpassung dieses Programmes beinhalten. Die Stellen, an denen mein Name steht, dürfen nicht verändert werden. Selbstverständlich übernehme ich keine Haftung jeglicher Art für irgendwelche Schäden, die direkt oder indirekt durch die Nutzung dieses Programmes entstehen könnten. Fehlerberichte und Anregungen zur Weiterentwicklung sind jederzeit willkommen. Ich freue mich auch über eine Benachrichtigungsmail, wenn das Programm irgendwo eingesetzt wird.

- siscgi.pl

- Dasselbe, aber im Browser lesbar

- proxycisauth.pl

- Dasselbe, aber im Browser lesbar

- recherche.html-torso.txt