Subversion SSS

Çeviren:
Volkan "knt" YAZICI
<v0lkany (at) yahoo.com>

25 Şubat 2005

Özet

Bu belge Subversion paketinin 1.0.9 sürümünde bulunan `www/project_faq.html` dosyasının çevirisidir.

Konu Başlıkları

Geçmiş

1.0.9	25 Şubat 2005	Özgün belge: CollabNet, Çeviri: knt

Çevirinin özgün sürümünü http://www.students.itu.edu.tr/~Eyazicivo/doc/subversion-«sss.html adresinde bulabilirsiniz.

Özgün belge için yasal uyarı

```
 * This software consists of voluntary contributions made by many
 * individuals on behalf of CollabNet.
 */
```

Bu çeviri için yasal uyarı

1. Genel Sorular

1.1. Neden böyle bir proje mevcut?

CVS kullanıcısını ele geçirmek için. Yapmaya çalıştığımız daha çok CVS'in eksik birçok yanını tamamlayarak yeni bir sürüm denetim sistemi oluşturmak. Daha fazla bilgi için anasayfamıza[B2] bakabilirsiniz.

1.2. Subversion mülkiyeti özel bir yazılım mı? Subversion'ın CollabNet'e ait olduğunu duydum.

Hayır, Subversion açık kaynak kodlu, özgür bir yazılım. CollabNet sadece tüm zamanını yazılım geliştirmeye harcayan geliştiricilerin ücretlerini ödeyerek, yazılım kodunun telif haklarını, Debian Özgür Yazılım Kılavuzu[B3] ile uyumlu Apache/BSD–tarzı bir lisans ile, elinde tutuyor. Diğer bir ifadeyle, Subversion'ı CollabNet ya da başka birinden hiçbir şekilde izin almaya gerek kalmadan, istediğiniz şekilde bilgisayarınıza yükleyebilir, Subversion'ın üzerinde değişiklikler yapabilir ve onu dağıtabilirsiniz.

1.3. Kendi projelerimde kullanmak için Subversion yeterli kararlılığa sahip mi?

Evet, kesinlikle. Başlıca öneme sahip olan ('prime–time') üretimler için kullanıma hazırdır.

Subversion 2000 yılından beri geliştirilmektedir ve gelişiminden 1 yıl sonra kendi kendini sunabilecek hale gelmiştir. Bir yıl sonra "alpha" sürümü duyurulduğu zaman ise zaten düzinelerce geliştirici ve şirket tarafından kullanılmaktaydı. Alfa sürümünün duyurulmasından sonraki iki sene boyunca, yazılım 1.0 sürümüne ulaşana kadar, hataları düzeltilip daha yazılımın kararlı bir hale gelmesi sağlandı. Yazılımı 1.0 ile çağırmamayı biz ne kadar uzun tutmaya çalışsak da, bir çok insan 1.0 sürümüne çıkmadan önce bile, Subversion'ı 1.0 sürüm numarası ile çağırıyordu. Şunun da farkındaydık ki, bir çok insan Subversion'ı kullanmak için 1.0 sürümünü bekliyordu ve yazılım hakkında bu sürüm etiketine bağladıkları çok farklı beklentileri vardı. Ve biz de bu standardı bozmadık.

1.4. Subversion'ın istemci/sunucu uyumluluğunda izlediği yöntem nedir?

İstemci ve sunucu tarafı sürüm numaralarının ilk kısımları aynı olduğunda uyumlu çalışacağı düşünülüp, bu şekilde tasarlanmıştır. Şöyle ki: `1.X` sürümüne sahip bir istemci, `1.Y` sürüm numaralı bir sunucu üzerinde çalışabilecektir. Fakat şu da unutulmamalıdır ki, sürüm numaraları farklı istemci ve sunucular arasındaki iletişimde bazı işlemlerin kullanılamayacak olması olasılığı vardır.

Sunucu ve istemcinin birlikte işlerlik politikası HACKING[B4] dosyasının "Compatibility" (Uyumluluk) kısmında belgelenmiştir.

1.5. Subversion hangi işletim sistemleri üzerinde çalışabilir?

Günümüzdeki tüm Unix, Win32, BeOS, OS/2, MacOS X türevlerinde çalışabilir.

Subversion ANSI C ile yazılmış olup, APR ()Apache Portable Runtime[B5]) kütüphanesini taşınabilirlik katmanı olarak kullanmaktadır. Bu nedenle Subversion, APR'nin çalıştığı her platformda (ki bu neredeyse tüm platformları kapsıyor) rahatlıkla çalışabilir. Subversion'ın sunucu (arşiv) tarafı ise dosya sistemi BDB olmadığı sürece Win9x platformlarda da çalışmaktadır. (Çünkü Berkeley DB[B6]'nin Win9x sistemlerdeki paylaşımlı bellek kısımlarında bazı sorunları var.) 1.1 sürümünde kullanılmaya başlanan FSFS arşivleri bu kısıtlamaya sahip değildir. Fakat Win9x sistemlerdeki dosya kilitleme mekazimasındaki bir kısıtlamadan dolayı FSFS de Win9x sistemlerde çalışmamakta. Ama FSFS için bu kısıtlamanın da 1.1.2 sürümünde ortadan kalkması için çalışılıyor.

Toplamak gerekirse; Subversion istemcisi APR'nin çalıştığı her platformda; Subversion sunucusu ise yine, arşivi Win95/Win98/WinMe sistemlerde tutamasa bile, APR'nin çalıştığı her platformda çalışabilir.

1.6. Bu yeni dosya sistemi nasıl bir şey? Ext2 benzeri bir şey mi yoksa?

Hayır. "Subversion dosya sistemi" çekirdek seviyesinde, işletim sistemlerine kurulabilen bir dosya sistemi değildir. Bunun yerine, Subversion'ın arşiv tasarımı için kullanılan bir terimdir. Arşiv, veritabanı üstüne kurulmuş olup, sürüm numaralarına sahip bir dosya sistemini taklit eden bir C API'si sunuyor. Bu yüzden arşive erişen bir yazılım için yazmak, diğer dosya sistemleri için yapılanlara benzer. Normal bir dosya sistemi ile Subversion dosya sistemi arasındaki ana fark ise, Subversion dosya sisteminde, herhangi bir dosya/dizin yenisi ile değiştirilse ya da silinse dahi, arşivde kendine ait bir sürüm numarası ile varlığını korur.

1.7. Subversion'ın bir Apache eklentisi olduğunu duydum. Subversion sunucular için ne kullanıyor?

Hayır. Subversion bir çeşit kütüphaneler topluluğudur. Yanında bu kütüphaneleri kullanan bir komut satırı istemcisi ile gelir. İki çeşit Subversion sunucusu vardır: **`svnserve`**, cvs pserver benzeri, başlı başına tek bir sunucudur ya da Apache **`httpd-2.0`** ile **`mod_dav_svn`** modülü. **`svnserve`** özel bir protokol kullanırken, **`mod_dav_svn`** ağ protokolü olarak WebDAV kullanır. Daha fazla bilgi için Subversion kitabındaki 6. bölüme[B7] bakınız.

1.8. Yani bu Subversion kullanmak için Apache yüklemek zorunda olduğum anlamına mı geliyor?

Kısaca cevaplamak gerekirse: Hayır.

Eğer sadece bir Subversion arşivine erişmek istiyorsanız, istemciyi kurmanız yeterli. Eğer ağdaki bir Subversion arşivine ev sahipliği yapmak istiyorsanız Apache2'yi ya da **`svnserve`**'ü kurmalısınız.

Ağ üzerinden ulaşılabilir bir Subversion sunucusu kurmak için daha fazla bilgiye Subversion kitabındaki 6. bölümden[B8] ulaşabilirsiniz.

1.9. Şu an Apache 1.x kullanıyorum ve Subversion arşivlerini sunmak için Apache 2.0 kuramam. Bu bir Subversion sunucusu çalıştıramayacağım anlamına mı geliyor?

Hayır, Subversion sunucusu olarak **`svnserve`** kullanabilirsiniz. **`svnserve`**'de çok iyi bir şekilde çalışıyor.

Eğer WebDAV ve onun ile gelen diğer bir çok eklentiyi istiyorsanız, evet, o zaman Apache 2.0 kurmalısınız. Apache 2.0'ı başka bir port numarası üzerinde çalıştırırken, Apache 1.x'i 80. port altında çalıştırmak daima bir seçenek olmuştur. Apache'nin farklı sürümleri aynı makine üzerinde sorunsuzca çalışabilmektedir. Yapmanız gereken tek şey `httpd.conf` dosyasındaki `Listen` ibaresinin karşısındaki 80 değerini 8080 gibi istediğiniz bir değer ile değiştirmek ve arşivinizin URL'sini doğru port değeri ile vermek. (Örnek: `http://svn.mydomain.com:8080/repos/blah/trunk/`)

1.10. Neden SCM'nin Y sistemi gibi bir X sistemi kullanmıyorsunuz?

SCM (Kaynak Denetim Yönetimi [Source Control Management]) sistemlerinde yeni bir çığır açma ya da piyasadaki tüm SCM'lerin en iyi özelliklerini toplama gibi bir niyetimiz yok. Yaptığımız tek şey CVS'in yerine bir şeyler oluşturmak. Lütfen ilk soruya bakınız.

1.11. Neden tüm arşivim aynı revizyon numarasını paylaşıyor? Tüm projelerimin kendinlerine ait revizyon numaralarının olmasını istiyorum.

Arşivdeki genel bir revizyon numarasının kullanıcının bakış açısından hiçbir anlamı yoktur. Bu şema tasarımının asıl hedefine ulaşmasını sağlayan bir iç mekanizmadır. Bu sayede kullanıcı herzaman saçma sapan uzun tarih/zaman katarları yazmaktan kurtarılmış olur.

Revizyon numaraları sadece arşiv ve kullanıcı açısından kullanım kolaylığı ile ilgilidir. Bunun arşivde ne sakladığınız ile ilgili bir nedeni yoktur. Hatta arşivdeki revizyon numaraları projenin gerçek gelişimi hakkında isabetli bir tahmin yapmak için dahi yeterli değildir. Projenin gerçek gelişimi hakkında fikir sahibi olmanın çok daha karmaşık yolları vardır.

1.12. Subversion 'changeset' özelliğine sahip mi?

Bu biraz yüklü bir soru. Çünkü *changeset* denildiği zaman neredeyse herkes ya farklı bir tanım ile yaklaşıyor ya da bir sürüm denetim sistemi *changeset* özelliği sunduğu zaman herkesin beklentisi farklı oluyor.

Bu ufak tartışmanın amacı doğrultusunda *changeset*'in ufak bir tanımını şu şekilde verebiliriz: Tek bir etiket altındaki değişikliklerin bir toplamı. Değişiklikler düz bir metin dosyası üzerinde olabileceği gibi, dizin yapısındaki düzenlemeleri ya da metadata ayarlamalarını da içerebilir. Daha genel anlamda, *changeset* sadece etikete sahip bir yamadır.

Subversion ilk sırada sürüm numaralarına sahip arşiv ağaçlarını yönetir (arşiv bu ağaçların toplamından oluşur) ve *changeset*'ler de bu arada türetilir. Arch ya da BitKeeper gibi sistemlerde öncelik *changset*'lerde olup (yani tüm arşiv bir yama deposudur) ağaç yapıları bu yamalar ile birlikte oluşturulur.

Kesin olarak ifade edilmeye çalışıldığında iki felsefenin de eksileri ve artıları var: hesap 30 yıl öncesine kadar gidiyor. Her ikisinin de geliştirilecek olan yazılımın türüne bağlı olaraktan iyi ve kötü yanları var. Bunu burada tartışmayacağız. Onun yerine, Subversion ile neler yapabileceğiniz hakkında bir açıklama bulacaksınız burada.

Subversion'da, global bir revizyon numarası olan 'N' arşivde bir ağacı etiketler ve bu sayede arşiv N. onaylamadan sonraki durumu görebilir. Bu ayrıca açıkca belirtilmiş bir *changeset*'in de etiketidir: Eğer N. ağaç ile N–1. ağacı karşılaştırırsanız, bu iki ağaç arasındaki onaylanmış kesin farkların bir yamasını elde edersiniz.

Bu sebepten dolayı, "N. revizyon"un sadece bir ağaç olmadığı kolayca anlaşılabilir olup, bunun ayrıca bir *changeset* olduğu da açıkça görülebilir. Eğer projenizdeki hataları herhangi bir durum takip sistemi kullanarak tespit ediyorsanız, revizyon numaralarını size ilgili hataların yamalarını vermeleri için kullanabilirsiniz. Örnek verecek olursak: "Bu durum 9238. revizyonda düzeltildi." şeklinde açıklamaya sahip bir arşiv kaydının düzeltilmesi için ilgili *changeset*'i `svn log -r9238` çıktısı ile gördükten sonra, yamayı `svn diff -r9237:9238` ile oluşturabilirsiniz. Buna ek olarak **svn**'nin `merge` komutu da revizyon numaralarını kullanır. Başka bir daldaki (branch) değişiklikleri kendi dalınıza aktarmak içinse, diğer dalın URL'sini parametreler arasına eklemeniz yeterli: `svn merge -r9237:9238`. Artık #9238 *changeset*'i de kendi çalışır kopyanıza eklenmiş oldu.

Düşünüdüğünde bu *changeset*'ler çevresine kurulmuş bir sistem için hiç de karmaşık değil, fakat hala CVS'e karşı uçsuz bucaksız bir kolaylık.

1.13. Subversion'ın yeni sürümü ne zaman çıkacak?

Proje durum sayfamıza bakabilirsiniz: http://subversion.tigris.org/project_status.html

1.14. Subversion sembolik bağları destekliyor mu?
Subversion 1.1 ve üstü sürümlerde normal **svn add** komutu ile sembolik bağ ekleyebilirsiniz.

Ayrıntıda ise, Subversion'ın kendi tuttuğu arşiv için sembolik bağ kavramı yoktur. Onun yerine sürüm numarasına sahip sembolik bağları svn:special özelliği ekleyerek normal bir dosyaymışcasına saklar. Unix istemcileri bu özelliği fark edip, çalışma kopyasında dosyayı bir sembolik bağmış gibi gösterir. Win32 istemcilerinin sembolik bağ özelliği olmadığından dolayı dosyanın bir sembolik bağ olduğunu algılayamaz ve normal bir dosyaymış gibi gösterir.

1.15. Subversion'ın yüksek çözünürlükte bir logosuna ihtiyacım var. Bunu nereden bulabilirim?
Subversion'ın logosunun vektörel biçimleri de mevcut olmak üzere, Subversion arşivine[B10] ve logoların bulunduğu dizine[B11] bakabilirsiniz.

Özel olarak logoların EPS[B12] ve Adobe Illustrator[B13] biçimleri de mevcut.

1.16. Sormak istediğim başka sorularım var. Daha fazla bilgiye nereden ulaşabilirim?
Lütfen merak ettiğiniz sorularınızı Subversion Kullanıcılar Grubunun e–posta listesine <users (at) subversion.tigris.org> gönderiniz. Diğer bir seçenek olarak ise bir çok Subversion kullancısını etkin olarak bulabileceğiniz irc.freenode.net[B15] IRC sunucusundaki #svn kanalına bakabilirsiniz.

2. Neyi Nasıl Yapabilirim?

2.1. Subversion'ın kaynak kodunu nasıl kendi çalışma dizinime çekebilirim (Nasıl 'checkout' yapabilirim)?

Subversion istemcisini kullanarak aşağıdaki komutu vermeniz yeterli:

```
$ svn co http://svn.collab.net/repos/svn/trunk subversion
```

Bu sayede Subversion'ın kaynak kodununun bir kopyasını kendi makinenizdeki `subversion` dizini altına indirmiş olacaksınız.

2.2. Nasıl bir arşiv oluşturup içine veri atabilirim?

`http://svn.collab.net/repos/svn/trunk/README` adresine ya da özel olarak *Quickstart Guide*'ın IV. bölümüne bakabilirsiniz.

Hatta daha ayrıntılı bilgi için Subversion Kitabı[B17] nın 5. bölümüne göz atınız.

2.3. Önceki bir CVS arşivimi, nasıl bir Subversion arşivine çevirebilirim?

Bu iş için Subversion geliştirme ekibi üyeleri **`cvs2svn`** adından bir araç geliştirmiş ve halen de devamlılığını sağlaması için bakımını sağlamaktadır. **`cvs2svn`**'i `http://cvs2svn.tigris.org/` adresinde bulabilirsiniz. Ama kullanmadan önce README[B19] dosyasını okumalısınız.

Eğer **`cvs2svn.py`** işinizi görmezse (örneğin arşiviniz üzerinde çevirme işlemi yaparken hata verip çıkıyorsa ya da dal ve etiketlerle sizin istediğiniz gibi işlem yapmıyorsa) kullanabileceğiniz başka 2 araç daha var. Tüm bu araçlar değişik özelliklere sahip (ve tabii ki beraberinde içerdikleri değişik hatalar ile):

Bunlardan birisi Chia–liang Kao tarafından yazılan VCP[B20]'ye dayanıyor. Buna CPAN[B21]'den ulaşabilirsiniz. Lev Serebryakov tarafından yazılmış **`refinecvs`**'e `http://lev.serebryakov.spb.ru/refinecvs/` adresinden ulaşabilirsiniz.

Bunun dışında Subversion bağlantılarına[B23] da göz atınız.

2.4. Ya eğer bir vekil sunucu arkasındaysam?

Eğer Subversion istemcisinde doğru ayarları yapacak olursak, vekil sunucu arkasındayken de rahatlıkla çalışabilirsiniz. İlk önce `servers` ayar dosyanız üzerinde gerekli değişiklikleri kullanacağınız vekil sunucuya göre yapın. Ayar dosyanın bulunduğu dizin kullandığınız işletim sistemine göre değişir. Linux ya da Unix sistemlerde `~/.subversion` dizini altında bulabilirsiniz ayar dosyasını. Windows sistemlerde ise `%APPDATA%\Subversion` dizini altında. (**`echo %APPDATA%"`** komutunu deneyin, fakat unutmayın ki bu gizlenmiş bir dizindir.)

Dosyanın içinde açıklayıcı yorum satırları ne yapmanız gerektiğini zaten yazar. Eğer bu dosyanız yoksa, en son Subversion istemcilerinden birini indirip herhangi bir Subversion komutu çalıştırın; bu sayede ayar dizinleri ve şablon dosyaları otomatik olarak oluşturulmuş olacaktır.

Eski Subversion sürümleri, 0.14.3 bootstrap paketi de dahil olmak üzere, bu dosyanın yerine `~/.subversion/proxies` kullanırlar. Bu eski dosya şu anki Subversion tarafından gözardı edilmektedir.

Bir sonraki adımda, vekil sunucunuzun Subversion tarafından kullanılan tüm HTTP yöntemlerini desteklediğine emin olun. Bazı vekil sunucular şu yöntemleri öntanımlı olarak desteklememektedir: `PROPFIND, REPORT, MERGE, MKACTIVITY, CHECKOUT`. Genel olarak, böyle bir sorunu çözmek ise kullandığınız vekil sunucusu yazılımı ile ilgilidir. Squid için ayar dosyası şu şekilde olacak:

```
#  TAG: extension_methods
#       Squid only knows about standardized HTTP request methods.
#       You can add up to 20 additional "extension" methods here.
#
#Default:
# none
extension_methods REPORT MERGE MKACTIVITY CHECKOUT
```

(Squid'in 2.4 ve üstü sürümleri zaten `PROPFIND`'ı desteklemektedir.)

Ek olarak, vekil sunucunuzdan kullanabileceğiniz HTTP yöntemleri ile ilgili daha fazla ayrıntı için *Subversion kullandığı tüm HTTP yöntemleri neler?* (sayfa: 32) başlıklı soruya bakabilirsiniz.

Eğer Subversion trafiğinizi vekil sunucudan geçirmek çok zor ya da imkansızsa ve hala Subversion kodlarını `checkout` etmek istiyorsanızvekil sunucunun etrafından dolanabilirsiniz. 80. portu filtreleyen bazı vekil sunucular yine de 81. porta hiçbir kısıtlama getirmezler. Bu nedenden dolayı, `svn.collab.net` arşiv sunucusu hem 80. portu hem de 81. portu dinlemektedir. Yani farklı olarak şunu da deneyebilirsiniz:

```
$ svn checkout http://svn.collab.net:81/repos/svn/trunk subversion
```

ve belki bu sayede vekil sunucuyu atlatmayı başarırsınız. Bir diğer strateji olarak `checkout` işlemini SSL üzerinden gerçekleştirmek olabilir (ki çoğu vekil sunucu yazılım buna izin vermektedir):

```
$ svn checkout https://svn.collab.net/repos/svn/trunk subversion
```

Elbette bunu yapmak için Subversion istemcinizin SSL desteği ile kurulmuş olması lazım. Bunu yapmak için **`./configure`** esnasından `--with-ssl` desteğini vermeniz yeterli. Kurulu Subversion istemcinizin SSL destekleyip desteklemediğini öğrenmek için ise **`svn --version`** komutunu deneyebilirsiniz.

2.5. Sistem yöneticilerim Subversion için bir HTTP sunucu kurmama izin vermiyor? Uzaktan erişim istiyorsam başka bir yolum var mı?

Bir seçenek olarak Apache yerine **`svnserver`** kullanmak olabilir. Ayrıntılı bilgi için Subversion kitabının 6. bölümüne[B25] göz gezdirebilirsiniz.

Fakat şu da var ki, eğer sistem yöneticileriniz sizin Apache çalıştırmanızı izin vermiyorsa, 3690. porttan başka bir sunucu çalıştırmanıza da büyük bir ihtimalle izin vermeyeceklerdir. Bundan sonra cevabın kalan bölümü sistem yöneticilerinizin var olan bir SSH altyapısını kullanmanıza izin veriyor olup olmamasından ibaret.

Eğer daha önceden CVS kullandıysanız, CVS sunucusuna bağlanmak için SSH kullanmış olmanız lazım. `ra_svn` Subversion bağlantıyöntemini kullanmak da bununla birebir. Sadece `svn+ssh` ön ekini arşiv URL'nize eklemeniz yeterli:

```
$ svn checkout svn+ssh://your.domain.com/full/path/to/repository
```

Bu sayede SSH ile karşı tarafta, sizin kullanıcı kimliğiniz ile giriş yapılan, özel bir **svnserve** işlemi başlatmış olup, tüm veri aktarımını şifrelenmiş bir kanal üzerinden gerçekleştirmiş olursunuz.

Bir diğer çözüm yolu olarak SSH port yönlendirmesini arkamıza alarak korunan sunucuya ra_dav ile bağlanmak. Güvenlik duvarınızın arkasında sizin arşivinize bağlanabilen bir makineye SSH ile bağlanmalısınız. Dikkat ederseniz bağlandığınız SSH sunucusu sizin arşivinizin bulunduğu makinede değil. Orada da olabilir, ama olmak zorunda da değil.

Daha sonra bulunduğunuz makineden sizin arşivinizi sunan HTTP sunucusuna bir port yönlendirmesi tanımlamanız gerekli. Daha sonra Subversion arşivinize bu yerel porttan rahatlıkla bağlanabilirsiniz. Ardından, istekleriniz Subversion arşivinize bu SSH sunucusu sayesinde kanal aracılığıyla iletilecektir.

Örnek olarak: Bir Subversion ra_dav kurulumu şirket güvenlik duvarınızın arkasındaki 10.1.1.50 makinesinde olsun (buna svn-server.example.com diyelim). Şirketiniz dış dünya tarafından kullanıma açık ssh-server.example.com'a SSH üzerinden bağlantıya izin veriyor olsun. Artık firewall arkasındaki arşivinize http://svn-server.example.com/repos/ours'den erişebilirsiniz.

Örnek: İstemci ssh sunucusuna port yönlendirmesi ile bağlanalım ve port yönlendirmesini kullanarak arşivi checkout edelim:

```
$ ssh -L 8888:svn-server.example.com:80 me@ssh-server.example.com
$ svn checkout http://localhost:8888/repos/ours
```

Şu da var ki svn-server.example.com httpd işlemini hakları kısıtlanmış bir kullanıcı ile çalıştırıyor olabilir. Bu durumda Subversion sunucunuz root bağlantısına ihtiyaç duymamış olacak.

Joe Orton şunu ekler:

> Subversion sunucusu MOVE ve COPY isteklerindeki "Destination" başlığında yer alan sunucu adına karşı hassastır. Bu yüzden bu noktada biraz dikkatli olmanız gerekmekte – bir "ServerAlias localhost" ibaresi her şeyin yolunda çalışması için gerekli.

SSH port yönlendirme ile ilgili bazı bağlantılar:

- http://www.onlamp.com/pub/a/onlamp/excerpt/ssh_11/index3.html
- http://csociety.ecn.purdue.edu/%7Esigos/projects/ssh/forwarding/
- TTSSH: Port yönlendirmesi yapabilen bir Win32 SSH istemci[B28]

2.6. Subversion altında birden fazla projeyi nasıl yönetebilirim?

Bu ilgilendiğiniz proje bağlı. Eğer projeler birbirleri ile ilgiliyse ve kendi aralarında veri paylaşıyorlarsa, en iyisi ikisini tutan bir çok alt dizinden oluşmuş tek bir arşiv oluşturmak olur. Şunun gibi:

```
$ svnadmin create /repo/svn
$ svn mkdir file:///repo/svn/projA
$ svn mkdir file:///repo/svn/projB
$ svn mkdir file:///repo/svn/projC
```

Eğer projeler birbiri ile tamamen ilişkisizse ve aralarında bir veri paylaşımı olmayacak gibi görünüyorsa, bu durumda her bir proje için kendine ait ayrı bir arşiv oluşturmak en iyisi olacaktır:

```
$ mkdir /repo/svn
$ svnadmin create /repo/svn/projA
$ svnadmin create /repo/svn/projB
$ svnadmin create /repo/svn/projC
```

(Ben Collins–Sussman'in <sussman (at) collab.net> açıklandığı gibi) Bu iki yaklaşım arasındaki fark ise şu şekilde:

> İlk durumda projeler arasında kolayca kopyalama ya da yer değiştirme işlemi gerçekleştirirken bunların hepsinin kaydı da saklanmış olur. (`svn cp/mv` aslında sadece tek bir arşivde çalışıyor.) Revizyon numaraları arşiv genelinde olduğundan, yapılan bir onaylamanın revizyon numarası diğer tüm projeler için de geçerli olacak. Ve bu şöyle bir tuhaflığa yol açacak: Geliştiricilerden birisi `projB`de bir değişiklik onaylatacak ve bir de bakacak ki bundan önce 10 tane revizyon olmuş, fakat bunları hiçbiri `projB` üzerinde olmamış. Aslında tam bir anlaşma sayılmaz. Sadece biraz tuhaf geliyor ilk bakışta. Bu daha çok **svn**'in başına, geliştiriciler sıklıkla **rapidsvn** üzerinde değişiklik onaylatıp, **rapidsvn** ile **svn** aynı arşivde olduğunda geliyor. :–)İkinci seçenek güvenlik açısından daha kolay olabilir: Apache'nin erişim denetimi mekanizmasını kullanarak projeleri birbirleri arasında izole etmek (kullanıcı hak ve dosya izinleri açısından) daha kolay olur. İlk durumda ise projeleri birbirlerinden ayırmak için bayağı becerikli askı (hook) betikleri yazmak zorundayız ("Acaba bu kullanıcı şu dizine yazma hakkına sahip mi?"). Elbette kullanıma hazır böyle bir betiğimiz mevcut.

2.7. Birbirinden tamamen farklı iki arşivi nasıl birleştirebilirim?

Eğer arşivlerden birinin geçmişi önemli değilse, diğer projenin altında yeni bir dizin oluşturup, bunu oraya atabilirsiniz.

Eğer iki projenin de geçmişi sizin için önemliyse, birini **svnadmin dump** ile yedekledikten sonra, bu yedeği **svnadmin load** ile diğerinin içine atabilirsiniz. Revizyon numaraları kaybolacak, fakat projelerin geçmişi korunmuş olacak.

Peter Davis <peter (at) pdavis.cx> bu konu ile ilgili **svn**'in CVS modüllerindeki kullanımına benzer bir yöntem anlatıyor:

> Eğer söz konusu olan ayrı ayrı iki farklı dizin ağaçlarının aynı arşivde birleştirilmesi ise, **svn**'in CVS modül versiyonlarını kullanabilirsiniz.
>
> Bir dizin üzerindeki `svn:externals` değişkenini orjinal arşiv `checkout` edilmeye çalışıldığında diğerlerinin de kendiliğinden `checkout` edilebileceği şekilde ayarlayınız. Diğer arşiv hala ayrı gözükürken, elinizdeki çalışma kopyasında ikisi de `merge` edilmiş gözükecektir. Eğer `checkout` ettiğiniz arşivde bir değişiklik onaylatmak isterseniz de bu diğer bağladığımız arşivi de etkileyecektir.
>
> `merge` işlemi tam olarak o kadar da net değildir: Çekim işlemi sadece çalışan örnekleri etkileyecektir, bu yüzden ikinci arşivden çekilmiş olan modüllere ulaşmak istediğinizde birinci adresin URL'sini kullanamayacaksınız. İkisi de ayrı URL'de kalmışlardır.

2.8. Arşivimi ya da elimdeki çalışma kopyamın yedeğini bir NFS sunucu üstünde tutmalı mıyım?

Eğer arşivinizi Berkeley DB üstünde tutuyorsanız (ki öntanımlı olarak Berkeley DB kurulur; dosya sistemi olarak) arşivinize NFS üzerinde erişmeyiniz. BDB veritabanının uzaktaki bir dosya sisteminde tutulmasına desteklememektedir[B29]. Bazı NFS sunucuları NFS ile bağlanmış disk bölümlerinde BDB desteklediklerini iddia etseler de, bu konuda gördüğümüz tek şey NFS ya da SMB ağı ile bağlı BDB veritabanlarında veri bozukluğu.

Eğer arşiviniz bir FSFS[B30] dosya sistemi kullanıyorsa, bunu (kitleme (locking) özelliği destekleyen) modern bir NFS sunucu üzerinde tutarsanız bir sorun olmaz.

Çalışan örnekler de NFS üzerinde tutulabilir (çok rastlanan bir senaryo olan ev dizininiz bir NFS sunucuda tutuluyorsa gibi). Linux NFS sunucularında, Subversion içinde yeniden adlandırma işlemleri çok yüklü

olduğu için bir arşivi `checkout` ederken bazı kullanıcıların dediğine göre alt dizinlerin alımı (subtree checking) özelliği kapalı olmalıymış (ki öntanımlı olarak açık gelir). NFS sunucularında alt ağaç alımının kapatılmasının nasıl olacağı hakkında ayrıntılı bilgi için NFS Sunucu Kılavuzu[B31] ve **exports**(5)'a göz atabilirsiniz.

SMB üzerinde arşiv çekimi yapılmaya çalışıldığında verinin bozulduğu şeklinde en az bir tane hata raporu aldık. Sorudaki sunucu Samba'nın eski bir sürümüydü (2.2.7a). Samba'nın yeni sürümü (3.0.6) ile bu sorun tekrarlanmadı.

2.9. Neden arşivim çok yer kaplıyor?

Arşiv tüm verinizi bir Berkeley DB ortamı altında `repos/db/` dizininde saklar. Burada bir çok tablo ve günlük dosyası (`log.*`) topluluğu bulunmaktadır. Berkeley DB bütün bu tablolarda yapılan değişikliklerin kaydını tutar ve bu sayede herhangi bir kesinti esnasında veri kurtarılabilir. (*Daha fazla bilgi* (sayfa: 33).)

Eğer günlük dosyaları hakkında bir şey yapmazsanız, sürekli değişikliklerin kaydını tutarak, çok fazla disk alanı kaplayacak şekilde büyüyecektir. İstenilen herhangi bir anda Berkeley DB aslında sadece bir kaç günlük dosyasına ihtiyaç duyar (Şu mesaja[B33] ve devamına bakınız); geri kalan rahatlıkla silinebilir. Eğer tüm günlük dosyalarını sonsuza kadar koruyacaksanız, Berkeley DB teorik olarak oluşumunun ilk anından sonsuza kadar tüm değişiklikleri kaydedecektir. Fakat pratikte, eğer yedek alıyorsanız, bu harcadığı disk alanına değmeyecektir.

svnadmin'i kullanarak hangi günlük dosyalarının silinebileceğini görebilirsiniz. Bunu yapacak bir cron görevi dahi atayabilirsiniz.

```
$ svnadmin list-unused-dblogs /repos
/repos/db/log.000003
/repos/db/log.000004
[...]
$ svnadmin list-unused-dblogs /repos | xargs rm
# disk space reclaimed!
```

Ayrıca Berkeley DB'nin `db_archive` komutunu da kullanabilirsiniz:

```
$ db_archive -a -h /repos/db | xargs rm
# disk space reclaimed!
```

Ayrıca **svnadmin hotcopy** ya da **hotbackup.py**'ye de göz atınız.

Bilgi

Berkeley DB 4.2, Subversion 0.35 ve üzerinde yeni oluşturulan arşivlerde otomatik günlük dosyası silme işlemi etkin haldedir. Bunu kullanmak için **svnadmin create** komutuna `--bdb-log-keep` seçeneğini eklemeniz yeterli. Berkeley DB kılavuzundaki DB_LOG_AUTOREMOVE[B34] kısmına göz atabilirsiniz.

2.10. Arşiv izinlerini nasıl doğru bir şekilde ayarlayabilirim?

Mümkün olduğu kadar az kullanıcının arşiv üzerinde yetkisi olmasını sağlayın. Örnek olarak, apache ya da **svnserve -d** komutunu özel bir kullanıcı ile başlatın ve tüm arşiv yetkisini bu kullanıcıya verin. Bu kullanıcı dışında başka hiçbir kullanıcının arşive `file:///` şeklinde bir URL kullanarak erişimini kısıtlayın ve **svnlook**, **svnadmin** ile arşiv üzerinde işlem yapmaya çalışırken, tahsis ettiğimiz o özel kullanıcının hakları ile bu komutları çalıştırdığınızdan emin olun.

Eğer istemciniz arşive `file:///` veya `svn+ssh://` URL'leri ile ulaşıyorsa, o zaman birden fazla kullanıcının arşive erişiminden kurtulamayız. Bu durumda, Subversion kitabının 6. kısmının son bölümüne[B35]

bakın ve en altta yer alan `checklist` kenar çubuğuna dikkat edin. Bu senaryoyu daha güvenli bir hale getirmek için atmanız gereken bir kaç adımın altını çiziyor.

SELinux / Fedora Core 3+ / RedHat Enterprise kullanıcıları için bilgi:

Normal Unix izinlerine ek olarak SELinux altında her dosya, dizin, işlem için bir 'güvenlik bağlamı' (security context) vardır. Bir işlem herhangi bir dosyaya erişmeye çalıştığında, sistem normal dosya izinlerinin ötesinde dosyanın ve işlemin güvenlik bağlamlarının uyumlu olup olmadığı yönünde kontrol yapar.

Fedora Core 3'de diğer sistemlerin aksine SELinux yüklü ve ayarlı olarak gelir, ki bu nedenle Apache çok kısıtlı haklar altında çalışır. Subversion'ı Apache altında çalıştırabilmek için arşivin güvenlik bağlamlarını Apache'nin erişebileceği şekilde ayarlamanız gerekmekte. **`chcon`** komutunu güvenlik bağlamlarını ayarlamakta kullanabilirsiniz (**`chmod`** komutunda hakları ayarlamaya benzer şekilde). Örnek vermek gerekirse: Kullanıcılardan biri şu komutu çalıştırabilir:

```
$ chcon -R -h -t httpd_sys_content_t PATH_TO_REPOSITORY
```

ki, güvenlik bağlamı arşive ulaşabilecek şekilde ayarlansın.

2.11. Neden salt–okunur işlemler için de arşive yazma izni gerekiyor?

Çoğu istemci işlemi sadece okumadan ibarettir; `checkout` ya da `update` gibi. Bir erişim denetimi açısından, apache de hepsini bu şekilde görür. Ama `libsvn_fs` (arşiv dosya sistem API'si) hala herhangi bir veri üretmek için arşive geçici yazma işlemi yapmak zorundadır. Bu nedenle arşive erişen tüm işlemler, Berkeley DB dosyalarına erişip onları kullanabilmek için, hem yazma hem de okuma işlemi yapmak zorundadır.

Özel olarak, arşiv çok fazla "salt–okunur" işleme iki ağaç yapısını karşılaştırarak cevap verir. Ağaçlardan biri genellikle HEAD revizyonudur ve diğeri de sıklıkla geçici bir hareket ağacıdır, ki bu nedenle bir yazma işlemine ihtiyaç duyulur.

Bu kısıtlama sadece BDB altyapısına aittir, FSFS altyapısında[B36] böyle bir kısıtlama yoktur.

2.12. Bir dosyayı arşivden nasıl tamamen silebilirim?

Bazı özel durumlar vardır, bir dosya ya da onaylamanın tüm kayıtlarını ortadan kaldırmak isterseniz. (Belki birisi yanlışlıkla kişisel bir dosya onaylatmıştır.) Bu o kadar kolay değildir, çünkü Subversion kasıtlı bir şekilde bir verinin asla kaybolmaması için tasarlanmıştır. Revizyonlar, biri diğerinin üzerine eklenen değişmez ağaç yapılarıdır. Herhangi bir revizyonu geçmişten silmek, domino taşı etkisi yaratacaktır; tüm onun üstüne yığılmış diğer revizyonlarda bir karmaşaya sebep olup büyük olasılıkla `checkout` edilmiş arşiv örneğinizi de geçersiz hale sokacaktır.

Fakat Subversion'ın ilerisi için **`svnadmin obliterate`** komutu altında böyle bir planı var. (Bkz. Hata Raporu 516[B37].)

Şu an için ise tek bir çözüm yolunuz var: Arşivinizin **`svnadmin dump`** ile bir yedeğini aldıktan sonra buna **`svndumpfilter`** uygulayarak istenmeyen dizini dışladıktan sonra **`svnadmin loadile`** ile yeni bir arşivin içine atmak. Daha ayrıntılı bilgi için Subversion kitabının 5. bölümüne[B38] bakabilirsiniz.

2.13. Onaylanan bir değişikliğin günlük kaydını daha sonradan nasıl değiştirebilirim?

Günlük kayıtları arşivde her bir revizyona iliştirilmiş özellikler olarak saklanırlar. Öntanımlı olarak, günlük kayıt özelliği (`svn:log`) bir kere onaylandıktan sonra değiştirilemez. Bunun nedeni (`svn:log` gibi) revizyon özelliklerinde[B39] yapılan değişiklikler, bu özelliğin önceki değerinin tümüyle yok sayılmasına yol açmasıdır ve Subversion bunu yanlışlıkla yapmanızı engellemek için etkisiz kılmıştır. Bunun yanında, revizyon özelliklerini değiştirmek için Subversion'da kullanabileceğiniz bir kaç yöntem mevcut.

İlk yöntem olarak arşiv yöneticisi revizyon özelliklerinin değiştirilebilir olmasını sağlayabilir. Bunu `pre-revprop-change` adında bir askı ile yapabilir. (Bkz. Subversion Kitabı, 5. Bölümün ilgili kısmı[B40].) `pre-revprop-change` askısı revizyon özelliğinin değişmemiş haline ulaşıp onu bir şekilde koruyabilir. (Örneğin, eposta atarak.) Bir kere revizyon özelliklerinin değiştirilmesi etkinleştirildiğinde, bunu `--revprop` seçeneğini **`svn propedit`** ya da **`svn propset`** komutuna ekleyerek aşağıdaki yöntemlerden biri ile yapabilirsiniz:

```
$ svn propedit -r N --revprop svn:log URL
$ svn propset -r N --revprop svn:log "yeni gunluk kaydi" URL
```

Burada *N* revizyon numarasını, *URL* ise arşiv adresini belirtiyor. Eğer bu komutu çalıştığı kopyanın içinde giriyorsanız *URL* kısmını yazmanıza gerek yok.

Bir günlük kaydını değiştirmenin ikinci bir yolu ise **`svnadmin setlog`** komutunu kullanmaktır. Bunu arşivin dosya sistemindeki yerini belirterek yapabilirsiniz. Uzaktaki bir arşivi bu komut ile değiştiremezsiniz.

```
$ svnadmin setlog REPOS_PATH -r N FILE
```

Burada *REPOS_PATH* arşivin yeri, *N* revizyon numarası ve *FILE* da yeni günlük kaydının bulunduğu dosya. Eğer `pre-revprop-change` askısı yoksa (ya da bu askıyı bir sebepten dolayı geçmek istiyorsanız), `--bypass-hooks` seçeneğini kullanabilirsiniz. Eğer bu seçeneği kullanacaksanız çok dikkatli olun. Eposta ile haber verme ya da revizyon numaralarının kaydını tutan yedek alma askılarını da atlıyor olabilirsiniz.

2.14. Subversion için yazdığım bir yamayı nasıl onaylatabilirim?

`İlk olarak`, HACKING[B41] dosyasını okuyunuz.

Bunu tam olarak anladıktan sonra, `dev` listesine başlığı `[PATCH]` kelimesi ile başlayıp tek satırlık bir açıklama ve iletinin içinde ilgili yamanızı içeren bir eposta atınız (Eğer MUA'nız bunu 'spam' olarak görmezse). Sonra geliştiricilerden biri yamanızı alıp (üstünde gerekli değişiklikleri yaptıktan sonra) uygulayıp, sonucu kontrol edecektir.

Basit olarak izleyeceğiniz adımlar şu şekilde olacak:

```
$ svn co http://svn.collab.net/repos/svn/trunk subversion
$ cd subversion/www

    [ faq.html'de değişklikleri yap ]

$ svn diff faq.html > /tmp/foo
$ Mail -s "[PATCH] FAQ updates" < /tmp/foo
```

Elbette gönderdiğiniz eposta yamanızın ne yaptığını hakkında uzun ve tatmin edici bir açıklama içerecek, HACKING[B42] belgesinde da yazdığı gibi, ki siz bunu zaten okudunuz, değil mi?

2.15. Arşivde belirli bir yere nasıl aktarma ('import') işlemi gerçekleştirebilirim? (Orjinal arşiv kopyasında herhangi bir yer değiştirme ya da silme işlemi yapmadan, sıfırdan bir arşiv kolu oluşturmak gibi.)

Örnek vermek gerekirse, `/etc` dizininin bir kısmını arşivinizin altında versiyon kontrolü için saklayacağımızı varsayalım:

```
shell# svn mkdir file:///root/svn-repository/etc \
 > -m "Arşivde /etc'ye karşılık gelecek bir dizin oluştur."
shell# cd /etc
shell# svn checkout file:///root/svn-repository/etc .
shell# svn add apache samba alsa X11
shell# svn commit -m "Ayar dosyalarımın ilk sürümü."
```

Bu şekilde **svn checkout** komutunun şık bir özelliğinden yararlanmış oluyoruz: Arşivden belirli bir dizini, /etc altında zaten varolan bir dizine açıyoruz. İlk önce arşivimizde etc adında bir dizin oluşturduk ve /etc sanki bir normal bir çalışan kopyaymış gibi, arşivimizin etc dizinini /etc'ye checkout ettik. Bunu bir kere yaptıktan sonra, **svn add** komutunu kullanarak arşive istediğiniz dosya ya da dizini ekleyebilirsiniz.

1328 nolu hata raporunda[B43] import edilen bir dizinin doğrudan, çalışan bir arşiv kopyasına otomatik olarak çevrilebilmesi şeklinde de bir istek var.

2.16. Bazılarının Subversion sunucularını yükseltirken bahsettikleri "dök/yükle ('dump/load') döngüsü" tam olarak ne anlama geliyor?

Subversion'ın arşiv veritabanı şeması gelişim sürecinde sık sık değişti. Subversion'ın pre–1.0 sürümü ile oluşturulmuş eski arşivler bir üst sürüme güncellenmek istendiğinde aşağıdaki işlemlere ihtiyaç duyabilirler. Eğer Subversion'ın X ve Y sürümleri arasında bir veritabanı şeması değişikliği olmuşsa, arşiv yöneticileri Y sürümüne güncelleme işlemini şu şekilde yapabilirler:

1. **svnserve**, Apache ve arşive erişen diğer ne varsa kapatın.
2. **svnadmin**'in X sürümünü kullanarak aşağıdaki komutu çalıştırın:

```
shell# svnadmin dump /path/to/repository > dumpfile.txt
```

3. Arşivi başka bir yere taşıyın:

```
shell# mv /path/to/repository /path/to/saved-old-repository
```

4. Şimdi kurulu Subversion'ı X sürümünden, Y sürümüne (derleyerek ya da kurarak) yükseltin.
5. Subversion'ın yeni kurulu Y sürümü ile aşağıdaki komutu çalıştırın:

```
shell# svnadmin create /path/to/repository
```

6. Yine Y sürümü ile:

```
shell# svnadmin load /path/to/repository < dumpfile.txt
```

7. Tüm askı betiklerini ve diğer şeyleri eski arşivden yenisine kopyalayın.
8. **svnserve**, Apache ve diğerlerini baştan başlatın.

dump ve load işlemleri hakkında daha ayrıntılı bilgi için http://svnbook.red-«bean.com/html-«chunk/ch05s03.html#svn-«ch-«5-«sect-«3.4 adresine bakınız.

Bilgi

Çoğu yeni Subversion sürümü bir dump ve load işlemini aslında gerektirmemekte. Eğer böyle bir gereksinim doğarsa, bunu duyurularda ve CHANGES dosyasında göze çarpan bir şekilde belirtiriz. Eğer böyle bir uyarı ibaresi ile karşılaşmazsanız, herhangi bir dump veya load işlemine ihtiyacınız yok demektir.

2.17. İstemcilerin bir SSPI kimlik denetimi kullanarak Windows 'Domain Controller'dan sisteme giriş yapmalarını nasıl sağlayabilirim?

TortoiseSVN[B45] Windows sistemlerde Subversion kurulumu ile ilgili çok güzel bir belgeye sahip. http://tortoisesvn.tigris.org/docs/TortoiseSVN_en/ch03.html#tsvn-«serversetup-«apache-«5 adresindeki SSPI ile kimlik dentimi kısmına bakınız.

Bu belgenin eski bir sürümünde şöyle bir eksik satır var:

```
SSPIOfferBasic On
```

Bu satır olmadan, bir tarayıcı kullanıcıdan kimlik bilgilerini isteyecekken, bir Subversion istemcisi bunu yapmayacaktır. (Bir tarayıcı SSPI kimlik denetiminden anlıyor, fakat NEON – Subversion'ın HTTP kütüphanesi – sadece basit kimlik denetimini anlayabiliyor.) İstemci hiçbir zaman kimlik denetimi için sormayacağından, kimlik denetimi isteyen herhangi bir işlem geçersiz olacaktır. Bu satır sayesinde `mod_auth_sspi` istemci ile basit kimlik denetimi kullanacak, fakat kimlik doğrulaması için Windows 'Domain Controller'dan faydalanacak.

2.18. `.svn` dizin isimleri hoşuma gitmedi; `SVN` tarzı bir şeyi daha çok tercih ederdim. Bunu nasıl değiştirebilirim?

Önerimiz, eğer yapabiliyorsanız `.svn` kullanmanız yönünde olacak. Eğer değiştirecek olursanız, sizin kullandığınız Subversion istemcisinden başkasını kullananlar için sorun çıkabilir. Eğer yapmak zorunda iseniz, `subversion/include/svn_wc.h` dosyasındaki

```
#define SVN_WC_ADM_DIR_NAME ".svn"
```

satırını

```
#define SVN_WC_ADM_DIR_NAME "SVN"
```

satırı ile değiştirip, Subversion'ı yeniden derlemeniz yeterli.

2.19. Bir dosyanın büyük–küçük harfini nasıl değiştirebilirim?

Bu sorun iki durumda çıkıyor. Eğer dosyalarınızı dosya sistemi büyük–küçük harf duyarsız bir işletim sisteminde, Windows gibi, ekliyorsanız kendinizi herhangi bir dosyayı yanlış harf kullanarak yazmış bulabilirsiniz. Alternatif olarak, bir dosya isminin harflerinin büyük–küçüklüğünü değiştirmek istemiş olabilirsiniz.

Eğer büyük–küçük harf duyarlı bir dosya sisteminde çalışıyorsanız, bu zaten bir sorun oluşturmayacaktır. Dosyayı yeni ismine taşımanız yeterli:

```
svn mv file.java File.java
```

Fakat bu büyük–küçük harf duyarsız bir işletim sisteminde çalışmayacaktır. Windows'da bunu başarmak için, dosyayı geçici bir yere kopyalarsanız, ardından arşivdekini sildikten sonra bunu tekrar yerine koyarsınız. Ya da daha iyi bir yöntem olarak Subversion URL'lerini kullanarak dosyanın yerini değiştirmek suretiyle yeniden adlandırabilirsiniz. Bu sayede boş yere geçmiş kayıtlarında yer harcamamış olup anında istediğiniz hale gelecektir.

İki yöntem de Windows'daki çalışan kopyalarınızı sorunlu bırakacaktır; çünkü, Windows dosyaları güncellemeye çalıştığında dosya isimlerini birbirine karıştırabilir. (Şöyle bir mesaj alacaksınız: *svn: Failed to add file 'File.java': object of the same name already exists.*) Eğer bunu yapmak istemiyorsanız, iki adımdan oluşan bir işlem gerçekleştirmelisiniz.

Harfleri yanlış yazılmış her dosya için, şu komut harfleri değiştirecektir:

```
$ svn mv svn://svnserver/path/to/file.java ¬
        svn://svnserver/path/to/File.java
```

Çalışan örneği güncellemek için, ilgili dizine geçin ve şunu yapın:

```
$ svn update file.java
$ svn update
```

İlk güncelleme `file.java` dosyasını kopyanızdan silecek, ikincisi ise `File.java`'yı ekledikten sonra sizi elinizde doğru çalışır bir kopya ile bırakacak. Eğer çok fazla problem yaratan dosyanız varsa, şunu da deneyebilirsiniz:

```
$ svn update *
$ svn update
```

Gördüğünüz üzere, yanlış harf ile girilen bir dosyayı, büyük–küçük harf duyarsız bir dosya sisteminde düzeltmek oldukça meşakkatli bir iş. Elinizden geldiğince dosyayı ilk oluşturduğunuzda doğru karakterler ile oluşturmaya çalışın.

2.20. Katıştırma ('merge') işlemi için CVS'de kullandığım 'tag'ları kullanabilir miyim?

Aşağıda da gösterildiği gibi, birinin revizyon numarasını hatırlamanıza gerek kalmadan da bir dal, arşiv yığınına katılabilir. Ya da tersi (bu, aşağıdaki örnekte gösterilmemiştir).

Aşağıdaki örnekte içinde `foo` adlı değiştirmek istediğimiz dosyanın bulunduğu `bar` adında bir dalın, var olan arşiv depomuz olan `/home/repos`'da nasıl oluşturulacağı anlatılıyor.

Dal katıştırmalarını izlemek için, arşivde `tags/branch_traces/` etiketleri saklamak için kuruldu.

```
# dalları ve etiketleri ayarlayalım
$ svn copy file:///home/repos/trunk \
>          file:///home/repos/branches/bar_branch \
>          -m "start of bar branch"
$ svn copy file:///home/repos/branches/bar_branch \
>          file:///home/repos/tags/branch_traces/bar_last_merge \
>          -m "start"

# Dalın çalışan kopyasını checkout edelim
$ svn checkout file:///home/repos/branches/bar_branch wc
$ cd wc

# foo.txt dosyasını düzenleyip arşive teslim edelim
$ echo "some text" >> foo.txt
$ svn commit -m "edited foo"

# trunk'ı değiştirip daldaki değişikleri katıştıralım
$ svn switch file:///home/repos/trunk
$ svn merge file:///home/repos/tags/branch_traces/bar_last_merge \
>           file:///home/repos/branches/bar_branch

# Şimdi 'foo.txt' içeriğini kontrol edelim,
# değişiklikleri içeriyor olmalı.

# katıştırdığımız kısmı teslim edelim
$ svn commit -m "Merge change X from bar_branch."

# Son olarak, son durumu yansıtması için izlemek
# için kullandığımız 'trace' dalını güncelliyoruz.
$ svn delete -m "Remove old trace branch in preparation for refresh." \
>            file:///home/repos/tags/branch_traces/bar_last_merge
$ svn copy file:///home/repos/branches/bar_branch \
>          file:///home/repos/tags/branch_traces/bar_last_merge \
>          -m "Reflect merge of change X."
```

2.21. Neden `$Revision$` anahtar değişkeni değer olarak dosyanın değiştirilmiş son revizyon numarasını alıyor; halbuki ben o anki geçerli sürüm numarasını almasını istiyorum.

Subversion arşivin revizyon numarasını genelin tümü itibariyle arttırdığı için, herhangi bir anahtar kelimenin bu numara yerine geçmesi mümkün değildir. Aksi halde her güncelleme ya da onaylamadan sonra elinizdeki kopyada yer alan tüm örnekler teker teker taranıp, muhtemelen de değiştirmeli.

İstediğiniz bilgiye (elinizdeki kopyanın revizyon numarasına) **svnversion** komutu ile ulaşabilirsiniz. Bunun ile belirttiğiniz kopyanın revizyon numarası hakkında bilgi sahibi olabilirsiniz. (Bkz. **svnversion --help**)

Windows kullanıcıları TortoiseSVN yükleme sayfasından[B47] SubWCRev.exe uygulamasını indirebilirler. Bunu kullanarak belirtilen bir dosyadaki tüm `$WCREV$` etiketlerini o anki kopyanızın revizyon numarası ile değiştirebilirsiniz.

2.22. Subversion da CVS'deki `$Log$` gibi bir anahtar değişkene sahip mi?

Hayır. CVS'deki `$Log$` anahtar kelimesi için bir karşılık yok. Eğer belirli bir dosyanın günlüğüne ulaşmak istiyorsanız, **svn log dosyam** ya da **svn log dosyamın_yolu** komutlarını kullanabilirsiniz. Posta listesinden, `$Log$`'un neden kötü olduğuna dair bir kaç açıklama:

> Dallar arasındaki değişikleri birbirine katıştırmaya başladığınız andan itibaren, `$Log$` tam bir felaket haline geliyor. Bu anahtar kelimenin doğası gereği otomatik olarak kolayca çözülemeyen uyuşmazlıklar pratik olarak garantilenmiştir.

Ve:

> Subversion günlük kayıtları, `svn:log` revizyon özelliği sayesinde, değiştirilebilir değerlerdir. Bu nedenle herhangi bir dosyada `$Log:$` yerine yazılmış olan değerinin tarihi geçmiş olabilir. İlgili dosya değişmemiş olsa bile `$Log:$`'un geçtiği her yerde girilmiş olan değer ileride değiştirilmek zorunda kalınabilir.

Umrumda bile değil. Her ne olursa olsun ben kullanmak istiyorum. Bir sonraki sürüme böyle bir özellik ekleyecek misiniz?

Hayır. Bu sürümü eklemek ya da eklenmesi için gerekli yama gönderilse bile, bu yamayı onaylamak gibi bir planımız yok henüz. Eğer dosyalarınızı günlük kayıtları ile dağıtmak istiyorsanız, kurulu sisteminizin imkanları doğrultusunda bununla uğraşabilirsiniz.

2.23. Projemdeki bir dosyanın yerel kullanıcılar tarafından paylaşılabilmesini, ama yapılan değişikliklerin arşive asla geçirilememesini istiyorum. Bu dosya için `svn commit`'i nasıl etkisiz hale getirebilirim?

Dosyayı sürüm denetimi altına koymayın. Bunun yerine sürüm denetimine `file.tmpl` gibi bir şablon koyun.

Ardından, başlangıç **svn checkout**'un ardından, kullanıcılarınızın (ya da kurulu sisteminizin) normal bir OS kopyalama işlemi ile bu dosyayı çekip, dosyanın üzerinde çalışmalarına izin verin. Dosya sürümlenmiş olduğundan, asla `commit` edilemeyecektir. Ve eğer isterseniz, dosyayı bulunduğu dizinin `svn:ignore` özelliğine ekleyerek, **svn status** çıktılarında bu dosyanın başında bir **?** işareti görmekten de kurtulmuş olursunuz.

2.24. Herhangi bir arşive `svn+ssh` ile giriş yaptığımda parolam `~/.subversion/auth/` içinde saklanmıyor. Her seferinde parolayı baştan yazmaktan başka bir yol var mı?

ssh'ın kendine ait parola ve kimlik denetleme önbellek şeması mevcut. Kimlik denetimi için kullandığı önbellekte saklama işlemi Subversion'ın dışında bir şey olduğu için, Subversion'dan ayrı olarak çözülmelidir.

OpenSSH ile beraber, anahtar oluşturmak için **ssh-keygen** ve parolaları istemcinin önbelleğine eklemek için **ssh-add** komutları gelmektedir. **keychain**, **ssh-agent**'ın kullanımını kolaylaştıran bir betik. Windows'da ise, **PuTTY** popüler bir ssh istemcisi; OpenSSH anahtarlarını almak için **PuTTYgen**'e, parolaları önbelleğe atmak için ise **pageant**'a bakınız.

ssh-agent'ın ayarlanması ise bu belgenin amaçları dışında, ama **ssh-agent**'ın Google araması[B48] sizi çabucak ilgili yanıtlara ulaştıracaktır. Eğer sabırsız biriyseniz, şu bağlantıları deneyebilirsiniz:

- http://www.csua.berkeley.edu/ssh-"howto.html
- http://mah.everybody.org/docs/ssh
- http://kimmo.suominen.com/docs/ssh/

2.25. Arşivdeki tüm dosyalar üzerindeki bazı özellikleri nasıl ayarlayabilirim? Ve arşive yeni eklenen tüm dosyaların da bu özelliğe sahip olduğundan nasıl emin olurum?

Subversion öntanımlı olarak kendiliğinden bir dosyanın içeriğini değiştirmeyecektir; bunun olması için dosyanın svn:eol-style ve svn:keywords özelliklerini kendiniz ayarlamalısınız. Bu Subversion'ı, CVS'in öntanımlı bu özelliğine karşılık daha güvenli kılmasının yanında bazı güçlükler de doğurmaktadır.

İlk soruya cevap verecek olursak: Bir arşivdeki tüm dosyalar üzerinde belirli bir özelliği sağlamak için bunu zor yoldan yapmalısınız. Yapabileceğiniz tek şey elinizdeki arşiv kopyasındaki tüm dosyalar üzerinde **svn propset** komutunu çalıştırmak ve ardından **svn commit** ile bu değişiklikleri onaylatmak. Betikler ile bu işi daha kolay bir hale sokabilirsiniz.

Ya ilerideki dosyalar için? Ne yazık ki, onaylanan dosyalarda sunucu tarafında bu değişiklikleri yapabilecek otomatik bir mekanizma yok. Bunun anlamı, kullanıcılarınız arşive **svn add** ile bir dosya eklerken bunun doğru haklarını bilip, ona göre ayarlamaları kendileri yapmalı. Neyse ki, bunu istemci tarafından yapabilecek bir aracımız var. Subversion kitabındaki auto–props[B52] kısmını okuyabilirsiniz. Tüm kullancılarınızın dosyalar için auto-props ayarlarını doğru yaptığından emin olmalısınız.

Sunucuya, onaylanmak istenen dosyaların haklarını kontrol edip ona göre geri çevirilip çevirilmeyeceğine karar veren bir pre-commit askı betiği ekleyebilirsiniz. (Bkz. http://svn.collab.net/repos/svn/trunk/contrib/hook-"scripts/check-"mime-"type.pl) Fakat bu yaklaşımın da bazı yan etkileri var. Birisi svn:eol-style özelliğini ayarlamayı unutulursa, örneğin, birisi o dosyayı başka bir OS'da açtığı an anlaşılacaktır. Bir kere fark edildikten sonra da düzeltilmesi kolaydır: Doğru özellikleri ayarla ve dosyayı arşive onayla.

Bilgi

Bir çok kullanıcının, sunucuların istemcilere auto-props ayarlarını otomatik olarak dağıtması gibi bir isteği oldu. 1974 numaralı özellik isteğinde[B54] bu zaten dile getirilmiş ve geliştiriciler tarafından hala üzerinde tartışılmaktadır; fakat şu ana kadar üzerinde başka bir çalışma olmamıştır.

2.26. Arşivimin hangi Berkeley DB sürümünü kullandığını nasıl anlayabilirim?

Eğer sunucunuz kendi makinenizde çalışıyorsa, bunun cevabı basit: sisteminizde hangi BDB sürümü yüklü ise. Eğer arşiv bir yedekten ya da bilinmeyen bir kaynaktan alınmışsa o zaman şöyle bulabilirsiniz:

En büyük numaralı db/log.* dosyası üstünde 12 ve 16. baytlardaki iki 4 baytlık tam sayıyı gösterecek bir komut çalıştırın. Bunu GNU **od** ile yapmak için **od -j12 -N8 -tx4 log.*<numara>*** ya da Mac OS X **hexdump** ile yapmak için **hexdump -s12 -n8 -x log.*<numara>*** komutlarını kullanabilirsiniz. İlk tam sayının sihirli numarası, dosyayı bir BDB günlük dosyasını tanıtan, 0x00040988 olmalı. İkinci sayı ise günlük biçiminin sürümüdür; aşağıdaki tablodaki BDB sürümlerinin hangi günlük biçimini kullanıldığını bulmak için kullanabilirsiniz.

```
Günlük biçimi sürümü    BDB uygulamasının sürümü
   5 (0x00000005)               4.0
   7 (0x00000007)               4.1
   8 (0x00000008)               4.2
  10 (0x0000000a)               4.3
```

2.27. Arşivimde bir internet sayfasının kayıtlarını tutuyorum. Yaptığım her değişiklik onayından sonra asıl internet sayfasının da aynı anda bu güncellemeleri yapmasını nasıl sağlayabilirim?

Bu sıklıkla karşılaşılan bir sorun olup, çözümü `post-commit` askı betikleri ile gayet kolaydır. Kitabın 5. bölümü[B55] olan askı betiklerine bakabilirsiniz. Yapmanız gereken tek şey çalışan internet sitesini sunucuda bir arşiv kopyası olarak tutmak ve `post-commit` askısına da karşı sunucuda **`svn update`** komutunun çalışmasını sağlayacak bir betik eklemek.

Pratikte, dikkat etmeniz gereken bir kaç nokta var. Onaylama işlemini sağlayan sunucu (**`svnserve`** ya da Apache), `post-commit` askı betiğinin çalıştığı sunucu ile aynı olmalı. Bunun anlamı, uygulamanın arşiv kopyasını düzgün bir şekilde güncelleyebilmek için doğru haklara sahip olması gerektiği. Diğer bir deyişle, arşiv kopyasının sahibi ve **`svnserve`** ya da Apache'yi çalıştıran kullanıcının hakları aynı olmalı; ya da en azından kopya arşiv üzerinde doğru haklar olamalı ki sunucu onu güncelleyebilsin.

Eğer sunucu izninin olmadığı bir arşiv kopyasını (örneğin, `joe` kullanıcısının `~/public_html/` dizini gibi) güncellemeye çalışacaksa, bir yöntem olarak uygulamayı `+s` izni ile çalıştırmanız olabilir, fakat Unix betiklerin `+s` ile çalışmasına izin vermez. Bunun yerine ufak bir C yazılımı kullanabilirsiniz:

```
#include <stdio.h>
#include <stdlib.h>

int main(void)
{
    return \
    ( system("/usr/local/bin/svn update /home/joe/public_html/") != -1 ) \
    ? 0 : 1;
}
```

ve derlenmiş yazılımın üzerinde **`chmod +s`** komutunu çalıştırıp, yazılımın `joe` kullancısınına ait olduğuna emin olduktan sonra, `post-commit` askı dosyasına yazılımı çalıştıracak bir satır eklenebilir.

Ek olarak Apache'nin `.svn/` dizinlerini de sunmasını istemeyiz sanırım. Bunun için şu satırı `httpd.conf` dosyanıza eklemeniz yeterli:

```
# Subversion çalışma kopyasının yönetsel
# dizinlerinde gezinmeyi yasakla.
<DirectoryMatch "^/.*/\.svn/">
    Order deny,allow
    Deny from all
</DirectoryMatch>
```

2.28. Arşivdeki tek bir dosyayı çalışma dizinime nasıl çekebilirim?

Subversion tek bir dosyanın `checkout` edilmesini desteklememektedir, sadece bir dizin yapısını `checkout` edebilirsiniz.

Bunun yerine tek bir dosyayı almak için **`svn cat`** komutunu kullanabilirsiniz. Bu dosyanın içeriğini alacaktır, ancak sürümlenmiş bir arşiv kopyası oluşturmayacaktır.

2.29. Çalışma dizinimdeki bütün ekleme, silme, kopyalama ve isim değiştirme işlemlerinden, tüm bu işlemler olduktan sonra nasıl haberdar olabilirim?

Olamazsınız. Denemesi kötü bir fikir.

Arşiv kopyasının tasarımında basit iki nokta vardır: (1) Dosyaları nasıl isterseniz öyle değiştirin ve (2) ağaç yapısında herhangi bir değişiklik yapmak istediğinizde (ekleme, silme, yer değiştirme, kopyalama) bir Subversion istemcisi kullanın. Eğer bu kurallara uyulursa, istemci arşiv kopyasını rahatlıkla yönetebilir. Eğer yeniden adlandırmalar ya da diğer işlemler Subversion'ın haberi dışında olursa, kullanıcı arayüzü ihlal edilmiş olup, arşiv kopyanız bozulmuş olabilir. İstemci bu durumda ne olduğuna karar veremez.

İnsanlar bazen böyle bir sorunla karşılaşırlar, çünkü arşivlerinin sürüm denetimini saydamlaştırmak isterler. Kullanıcıları kandırarak, onları bir arşiv kopyasına yönlendirirler ve daha sonra bir betik ile bu kopya üzerinde ne değişiklikler yapıldığına bakılıp ona göre ilgili istemci komutu çalıştırırlar. Ne yazık ki, bu yöntem uzun vadede pek işlemez. **`svn status`**, betiğin sonradan kolaylıkla **`svn add`** ya da **`svn rm`** yapabileceği, kayıp ve sürümlenmiş elemanları gösterir. Fakat bir silme ya da yer değiştirme işlemi gerçekleşirse, şanssızsınız demektir. Betiğiniz bunları algılayabilecek tuhaf yöntemler içerse bile, **`svn mv`** ya da **`svn copy`** işlem olduktan sonra çalışmayacaktır.

Özet olarak, arşiv kopyaları Subversion altında bir bütündür ve Subversion saydam olmak için tasarlanmamıştır. Eğer istediğiniz saydamlık ise bir Apache sunucusu kurarak "SVNAutoversioning" özelliğini kullanabilirsiniz. (Bkz. Subversion kitabında Ek Bölüm C[B56].) Bu sayede kullanıcıların arşive bir ağ diskiymiş gibi erişmesini sağlamış olup, yığına yapılan herhangi bir değişikliği otomatik olarak sunucu tarafında onaylatabilirsiniz.

3. Sorun Giderme

3.1. Arşivim sürekli kurtarma işlemine (`DB_RUNRECOVERY`) ihtiyacı olduğu yönünde hatalar verip donuyor. Bunun sebebi ne olabilir?

Arşivinizdeki BerkeleyDB veritabanı kesilmelere karşı hassastır. Eğer veritabanına bağlanmış bir uygulama çıkarken düzgün bir şekilde bağlantısını koparmazsa, veritabanı kararsız bir durumda bırakılır. Bunun en sık rastlanan nedenleri:

- Uygulama bir izin sorunu ile karşılaşıp çıkmıştır.
- Uygulama çökmüş ya da 'Segmentation Fault' vermiş olabilir.
- Uygulama kasıtlı olarak öldürülmiş olabilir.
- Diskte yer kalmamış olabilir.

Bu çeşit durumların çoğunda **`svnadmin recover`** komutunu, arşivi tekrar eski kararlı durumuna getirmek için kullanabilirsiniz. (*Daha fazla bilgi* (sayfa: 33).) Şu da unutulmamalı ki, disk alanı sıkıntısı yaşandığı zaman, sık sık `checkout` ve `update` işlemleri sunan bir arşiv geri dönüşü olmayacak şekilde çökebilir. (Bu yüzden yedek almayı eksik etmeyin.)

'SegFault' hataları, kasıtlı öldürülen uygulamalar ve disk alanında sıkıntı yaşaması gibi sorunlar ile nadiren karşılaşılır. İzin sorunları ise çok daha seyrek gözlenir: Bir uygulama arşive ulaşır ve yanlışlıkla bir yapının izin ya da sahibini değiştirir; ardından başka bir uygulama aynı yapıya erişmeye çalıştığında tıkanır.

Bundan kaçınmanın en iyi yolu, arşiv izin ve sahiplerini doğru bir şekilde ayarlamaktır. Öneriler için *buraya* (sayfa: 12) bakabilirsiniz.

3.2. Ne zaman arşive erişmeye çalışsam işlemim hep donup kalıyor. Arşivim zarar mı görmüş?

Arşiviniz zarar görmüş ya da veriniz kaybolmuş değil. Eğer uygulamanız veritabanına doğrudan (**`mod_dav_svn`**, **`svnlook`**, **`svnadmin`** ya da `file://` şeklinde URL kullanarak) erişiyorsa, verilerinize BerkeleyDB kullanarak ulaşıyordur. Berkeley DB kayıt günlüğü tutan bir sistemdir, ki bu da onun herhangi bir işlemde önce onun kaydının tutulduğu anlamına gelir. Eğer uygulamanız bir nedenden dolayı kesilmişse (Control+C ya da 'SegFault' ile), arkada bir kilit dosyası (lock file) ve işlemin neden bitirilemediğine dair de bir log dosyası bırakılmıştır. Arşivinize ulaşmaya çalışan herhangi bir uygulama, bu kilit dosyasının ortadan kalkmasını bekleyecektir. Arşivi tekrar ayağa kaldırmak için, Berkeley DB ile yaptığı işlemi bitireceğine ya da bitirmeyip eski kararlı haline geri dönüp dönmeyeceği hakkında seçim yapmanız lazım.

Uyarı

Eğer aynı anda arşivi onarmaya ve bir uygulama ile de ona erişmeye çalışırsanız, verinize ciddi şekilde zarar verebilirsiniz.

Bunu yapmadan önce arşive erişen tüm yolları kapattığınıza emin olun. (Örneğin, Apache'yi kapatın, **`svn`** komutunun çalıştırma izinlerini etkisiz kılın.) Bu komutu, root olarak değil de, arşivin sahibi olarak çalıştırmaya dikkat edin; aksi halde arkanızda root kullanıcısına ait dosyalar bırakacaksınız ve bunlar da o arşivi yöneten kullanıcı tarafından erişilemez olacaklar. Ayrıca doğru `umask` değerlerine sahip olduğunuza da dikkat edin; herhangi yanlış bir yerde arşive erişme iznine sahip grup üyeleri dışarda bırakılacaklardır.

En basit hali ile onarma işlemini çalıştırmak için:

```
$ svnadmin recover /path/to/repos
```

Komut bir kez işini bitirdikten sonra `db` dizinindeki hakları gözden geçiriniz.

3.3. Ne zaman bir `svn` komutu çalıştırsam, çalışma dizininin kilitlendiği şeklinde uyarı veriyor. Arşivim zarar görmüş olabilir mi?

Arşiv kopyanız kilitlenmedi ya da herhangi bir veri kaybınız olmadı. Subversion'ın arşiv kopyaları günlük kaydı tutma özelliği sahip olduklarından, yaptığınız her işlem öncesinde ve sonrasında kaydedilir. Eğer **`svn`** istemciniz bir şekilde kesilirse (işlem öldürülür, Control+C kullanılınır ya da işlem 'SegFault' verirse), işlemin neden durduğuna dair bilgilerin kayıtları ile birlikte arkada bir kaç kapatılmamış kilit dosyası bırakılır. (**`svn status`** komutunda başında L olan dosyalar kilitlenmiş anlamındadır.) Arşiv kopyanıza erişmeye çalışacak olan herhangi bir uygulama bu kilit dosyalarını gördüğünde hata verip çıkacaktır. Arşiv kopyanızı tekrar ayağa kaldırmak için **`svn`** istemcinize en son yapmaya çalıştığı işi bitirmesini söylemelisiniz. En basitinden:

```
$ svn cleanup çalışma-kopyası
```

3.4. Yaptığım değişiklikleri onaylatmaya çalıştığımda Subversion kopyamın tarihinin geçmiş olduğu uyarısını veriyor.

Buna şu üç durum neden olabilir:

1. Başarısız bir onaylama işleminin ardında bıraktığı döküntü arşivinizi kirletir.

Sunucuya yeni bir revizyon eklendiği anda sizin istemcinizin `post-commit` işlemlerini (sizin arşiv kopyanızın da güncellenmesi buna dahil olmak üzere) bitiriyor olması, arşiv ile aranızda bir sorun yaratabilir. Buna bir çok neden olabileceği gibi, (nadiren rastlansa da) veritabanı tarafından ya da (ki en sık bununla karşılaşılınır) yanlış zamanlı ağ kesintilerinden kaynaklanabilir.

Eğer böyle bir şey olursa, büyük ihtimalle onaylamak istediğini değişiklikler arşivde çoktan onaylanmıştır. **`svn log -rHEAD`** komutunu başaramadığınızı zannettiğiniz onaylama işleminizin gerçekleşip gerçekleşmediğini öğrenmek için kullanabilirsiniz. Eğer onaylamayı başarmışsanız, **`svn revert`** komutu ile yaptığınız değişiklikleri geri aldıktan sonra **`svn update`** ile kopyanızı yeni revizyona yükseltebilirsiniz. (Şuna da dikkat etmek gerek ki, sadece **`svn update`** sizin değişikliklerinizin de dahil olduğu güncel sürümü alabilir,**`svn revert`** bunu yapmaz.)

2. Karma revizyonlar.

   Subversion herhangi bir değişikliği onaylarken, istemci sadece revizyon numaralarını ilgilendiren düğümleri onaylar, arşiv kopyasındaki tüm düğümleri değil. Bunun anlamı, bir arşiv kopyasında, son olarak ne onayladığınıza bağlı olarak birbirinden farklı revizyonlara sahip dosya ve dizinler olabilir. Belirli işlemlerde (dizin özelliklerinin değiştirilmesi gibi), eğer arşiv dosyanın daha güncel bir revizyonuna sahip ise veri kaybı olmaması için onaylama kabul edilmeyecektir. (Bkz. Karma Revizyon ile Yapabilecekleriniz[B59].)

   `svn update` komutunu çalıştırarak arşiv kopyanızı düzeltebilirsiniz.

3. Gerçekten eski bir revizyonda olabilirsiniz. Şöyle ki, onaylamaya çalıştığınız değişikliğin bulunduğu dosya başka birisi tarafından çoktan yenilenmiştir. Bunun için **`svn update`** komutu ile arşiv kopyanızı güncellemelisiniz.

3.5. Dağıtımım ile gelen derlenmiş Subversion paketlerini kurdum ve Subversion'ı çalışma dizinime çekmeye çalıştığımda "Unrecognized URL scheme." hatası alıyorum. Bu neden kaynaklanabilir?

Subversion arşive ulaşmak için bir eklenti sistemi kullanır. Şu an bu eklentilerden üçü mevcut: `ra_local`, yerel bir arşive; `ra_dav`, WebDAV üzerinden bir arşive ve `ra_sav` de yerel ya da uzaktaki bir **`svnserve`** sunucusuna erişmenizi sağlar. Subversion'da herhangi bir arşive erişmeye çalıştığınızda, uygulama verdiğiniz URL şemasına göre uygun eklentiyi dinamik olarak yükler. `file://URL` şeması `ra_local` eklentisini ve `http://URL şeması ra_dav` eklentisini yüklemeye çalışacaktır.

Gördüğünüz hatanın anlamı, dinamik yükleyicinin (ya da bağdaştırıcının) yüklemek istediği ilgili eklentiyi bulamadığıdır. Bu daha çok Subversion'ı kütüphane paylaşımı (shared library) desteği ile kurduğunuzda karşılaşılır. Bu yüzden bir de **`make install`** yapmadan önce çalıştırmayı deneyin. Diğer olası bir neden ise, **`make install`** dediniz fakat kütüphaneler öyle bir yere yüklendi ki dinamik yükleyici bunları bulamıyor. Linux'da bunu çözmek için paylaşımlı kütüphanelerin bulunduğu dizini `/etc/ld.so.conf` dosyasına ekleyip **`ldconfig`** komutunu vermeniz yeterli. Eğer bunu yapmak istemiyorsanız ya da bunu yapmak için root haklarınız yoksa, aynı dizini `LD_LIBRARY_PATH` değişkenine de ekleyebilirsiniz.

3.6. Arşivimi aramaya ya da açmaya çalıştığımda hata alıyorum, fakat arşivin URL'sini doğru girdiğimden eminim. Nerede yanlış yapıyor olabilirim?

Şu soruya (sayfa: 33) bakınız.

3.7. **`configure`** komutunu çalıştırdığımda `subs-1.sed line 38: Unterminated 's' command.` hatası alıyorum. Sorun neden kaynaklı olabilir?

Büyük ihtimalle `/usr/local/bin/apr-config` ve `/usr/local/bin/apu-config` dosyalarının eski kopyaları sisteminizde mevcuttur. Bunları kaldırın, `apr/` ve `apr-util/` dizinlerinin tamamen güncel olduğundan emin olduktan sonra tekrar deneyiniz.

3.8. Subversion'ı *N*X altında BerkeleyDB 4.2 desteği ile derlemeye çalıştığımda sorun yaşıyorum. Ne yapmalıyım?

Subversion, `apr-util`'e uygun BDB kurulum seçeneklerini sorduktan sonra, BerkeleyDB'yi derler. Bunun anlamı, Subversion ya da Apache ile gelen `apr-util`'inizin BDB'yi doğru bir şekilde algılayabilmeli. Bunu yapmanın bir yolu `apr-util`'in **`./configure`** komutuna `--with-berkeley-db` seçeneğini vermek. (Bu seçeneği Apache ya da Subversion'a bir kez verdiğinizde, aynısı `apr-util` için de geçerli oluyor.)

Sorun şundan kaynaklanıyor: BerkeleyDB 4.2, `apr-util`'in son yayınlanan sürümünden daha yeni; dolayısıyla `apr-util` onu nasıl algılayacağını bilmiyor.

Uzun vadeli çözüm şimdiden hazır: CVS'deki `apr-util` kolaylıkla BDB 4.2'yi algılayabiliyor. `apr-util` ya da Apache httpd farklı sürümlere sahip olsa dahi bu özellik geniş ölçüde çalışacak.

Kısa vadede, elinizdeki sürüm için yapılabilecek en iyi şey `apr-util`'in **`./configure`** betiğine şu yamayı uygulamak[B61].

Eğer ilk önce Apache'yi kuruyorsanız, bu yamayı `httpd-2.0.48` ile gelen `apr-util`'in `configure` betiğine uygulayın ve şu seçenekler ile kurun:

```
$ ./configure \
> --enable-dav \
> --enable-so \
> --with-berkeley-db=/usr/local/BerkeleyDB.4.2 \
> --with-dbm=db42
```

Apache'nin doğru BDB kütüphanesi ile kurulup kurulmadığını şöyle anlayabilirsiniz:

```
$ ldd /usr/local/apache2/bin/httpd | fgrep libdb-4.2.so
```

Ardından Subversion'ı BDB ile daha fazla ilgilenmeye gerek kalmadan kurabilirsiniz. (...fakat Subversion, Apache standart bir yere kurulmamışsa, Apache'nin nereye kurulduğunun ufak bir parametre ile bildirimini isteyebilir.)

Eğer Apache'yi kurmuyorsanız, yamayı Subversion ile gelen `apr-util`'in **`./configure`** betiğinine uyguladıktan sonra benzer derleme seçenekleri girerek devam edin:

```
$ configure \
> --with-berkeley-db=/usr/local/BerkeleyDB.4.2 \
> --with-dbm=db42
```

Subversion'ın doğru BDB kütüphanesine göre kurulup kurulmadığını yine benzer bir şekilde öğrenebilirsiniz:

```
$ ldd /usr/local/bin/svn | fgrep libdb
```

Eğer kütüphanelerinizi öntanımlı olandan farklı yerlere kurduysanız, yukarıda izlediğimiz adımlardaki komutların yollarını ona göre ayarlamanız gerekmekte.

3.9. Subversion'ı Windows altında MSVC++ 6.0 ile derlemeye çalıştığımda sorun yaşıyorum. Ne yapmalıyım?

Büyük olasılıkla sadece en son platform SDK'sını edinmeniz yeterli. VC++ 6.0 ile gelen yeterli derecede güncel değil.

3.10. `file:///` URL'sinde bir Windows sürücüsünün harfini nasıl yazabilirim?

Şu şekilde:

```
$ svn import file:///d:/some/path/to/repos/on/d/drive
```

Bkz. Arşiv URL'leri[B62]

3.11. VS.NET/ASP.NET'in `.svn` dizini ile ufak bir uyuşmazlığı var gibi. Bu durumda ne yapabilirim?

VS.Net, yayınını IIS üzerinde yapmak için WebDAV kullanan, ASP.Net adında bir alt sisteme sahiptir. Bu alt sistem, '.' ile başlayan herhangi bir dosya yolunu reddeder. Bu uzaktan bir Subversion arşivi yayınlamaya çalıştığınızda `.svn` dizininden dolayı sorun oluşturur. Hata mesajı olarak da *unable to read project information* gibisinden bir şeyler alırsınız.

Bunun için bir kaç çözüm yolu mevcut:

- *Burada anlatıldığı gibi* (sayfa: 16), istemcinizi `.svn` yerine başka bir dizin kullanacak şekilden yeniden derleyin. Ya da,
- Eğer TortoiseSVN kullanıyorsanız, bu sorun için düzeltilmiş bir istemci kullanabilirsiniz. (Bkz. `http://tortoisesvn.tigris.org/download.html`. Veya,
- `http://weblogs.asp.net/fbouma/archive/2004/02/28/81479.aspx` adresindeki Jim Bolla'nın tavsiyesini deneyin: *"Eğer yerel ya da uzakta paylaşımdaki bir internet projelerinde çalışıyorsanız, projeyi normal bir sınıf kütüphanesi (regular class library) projesine çevirerek bu sorundan kurtulabilirsiniz. Bunun nasıl yapılacağına dair internetteki kaynaklardan derlediğim bazı belgelerim var."* (Jim Bolla ile bu belgeleri alabilmek için bağlantı kurduk. Eğer alabilirsek onları buraya ekleyeceğiz.) Ya da,
- `http://weblogs.asp.net/fbouma/archive/2004/02/28/81479.aspx` adresinde Steele Price'ın önerisini deneyebilirsiniz: *"Şu adımları izleyerek `.svn` isimli dizin sorununun ardından kolaylıkla gelebilirsiniz:* `http://blog.steeleprice.net/archive/2003/11/09/134.aspx`*"*

3.12. Ağ üzerinden bir Subversion arşivine yaptığım değişiklikleri onaylatmaya çalışırken sorun yaşıyorum.

Diyelim ki, kullanıcılardan biri yerel arşivi `import` ederken sorun yaşamadığını bildirmiş olsun:

```
$ mkdir test
$ touch test/testfile
$ svn import test file:///var/svn/test -m "Initial import"
Adding         test/testfile
Transmitting file data .
Committed revision 1.
```

Fakat aynı şey uzaktaki bir arşiv söz konusu olduğunda sorun yaşadığını da bildirmiş olsun:

```
$ svn import http://svn.sabi.net/test testfile -m "import"
nicholas's password: xxxxxxx svn_error:

#21110 : <Activity not found>

The specified activity does not exist.
```

Görüldüğü üzre, **`httpd`** işlemi `REPOS/dav/` dizini üzerinde yazma haklarına sahip değil. Apache'nin `dav/` (ve tabii ki `db/`) dizinine yazma yetkisi olduğundan emin olun.

3.13. Windows XP üzerinde çalışan Subversion sunucum bazen zarar görmüş veri yolluyor gibi. Böyle bir şey gerçekten olabilir mi?

Windows XP Servis Paketi 1'i yüklemelisiniz. Servis paketleri hakkında tüm bilgilere şu adreste ulaşabilirsiniz: `http://support.microsoft.com/default.aspx?scid=kb;EN-«US;q317949`

3.14. Subversion sunucusu ile istemci arasında geçen haberleşmeyi en iyi şekilde ağdan nasıl dinleyebilirim?

İletişimi Ethereal[B69] programı ile dinleyebilirsiniz:

1. *Capture* menüsünden, *Start*'ı seçiniz.
2. *Filter* kısmına `port 80` yazın ve karma özelliğini ('promiscuous mode'u) kapatın.
3. Subversion istemcinizi çalıştırın.
4. *Stop* tuşuna basın (çoğunlukla küçük bir penceredir). İşte şimdi dinlemiş oldunuz. Bir sürü satırdan oluşuyor olması gerek.
5. Listenin oluştuğu tablonun başındaki *Protocol*'e tıklayarak, paketleri kullandıkları protokole göre dizin.
6. Ardından ilk ilgili TCP satırını seçin.
7. Üzerine sağ tıklayıp, *Follow TCP Stream* seçeneğini seçin. Subversion istemcisinin HTTP işletişiminde kullandığı istek ve cevapları görüyor olmalısınız.

Yukarıdaki talimatlar Ethereal'ın grafik arayüzlü sürümüne göredir ve (`tethereal` ile bilinen) komut satırı sürümünde uygulanmamalıdır.

Bunun yanısıra, eğer güncel (0.16 sürümünden yüksek) istemciler kullanıyorsanız, sunucular-daki `neon-debug-mask` seçeneğini ayar dosyalarında etkin hale getirirseniz, **`svn`** istemcinizi çalıştırdığınızda hata ayıklama iletileri ekrana dökülecektir. `neon-debug-mask` değişkeninin nümerik değerleri olan `NE_DBG_...` tanımlarının değerlerine `ne_utils.h` başlık dosyasından ulaşabilirsiniz. 'neon' 0.23.7 için, `neon-debug-mask`'ı 130 (`NE_DBG_HTTP+NE_DBG_HTTPBODY`) gibi bir değere ayarlamanız, HTTP iletişimde akan verinin ekrana dökülmesini sağlayacaktır.

Ağ üzerinde tarama yaparken, sıkıştırma özelliğini de kapamak isteyebilirsiniz. Bunun için `config` ayar dosyasındaki `compression` parametresine bakabilirsiniz.

3.15. Neden `svn revert` açık bir hedefe ihtiyaç duyuyor? Neden öntanımlı olarak devirli değil? Bu özelliği neredeyse diğer tüm komutlar ile farklılaşıyor.

Kısaca: Bu sizin iyiliğiniz için.

Subversion verinizin korunmasına çok öncelikli bir önem verir ve bunu sadece sürümlenmiş veriniz için de yapmaz. Sürümlenmiş ve sürüm denetim sistemine eklenmek üzere ayarlanmış dosyalar büyük bir dikkatle işlenmelidir.

`svn revert` komutu açık bir hedefe ihtiyaç duyar (istenilen hedef dizin o an bulunduğunuz yer olsa bile) ve bu bunu başarmanın tek yoludur. Bu ihtiyaç (ve eğer istiyorsanız beraberinde `--recursive` (`-R`) seçeneği sizin ne yaptığınızdan emin olmanız için kasıtlı bir şekilde yer almaktadır. Çünkü dosya bir kez geri alındığında (`revert` edildiğinde), yerel değişiklikleriniz tamamiyle ortadan kalkar.

3.16. Apache'yi başlattığımda `mod_dav_svn` db–4.X sürümü yerine db–3.X sürümünü bulduğundan "bad database version" hatası veriyor.

`apr-util` kurulumunuz DB–3, **`svn`** ise DB–4 ile bağlanmış. Ne yazık ki, DB sembolleri farklı değil. `mod_dav_svn` modülü Apache'nin işlem alanına alındığında, `apr-util`'in DB–3 kütüphanesine bağlanmış sembollerine ulaşıyorlar.

Çözüm, `apr-util`'in DB–4 desteği ile kurulduğundan emin olmak. Bunun için Apache ya da `apr-util`'in **`./configure`** betiğine doğru seçenekleri (`--with-dbm=db4 --with-berkeley-db=/the/db/prefix`) vermeniz yeterli.

3.17. RedHat 9'da "Function not implemented" hataları aldığımdan Subversion hiçbir şekilde çalışmıyor. Bu sorunu nasıl giderebilirim?

Bu aslında bir Subversion sorunu olmamasına karşın, kullanıcıları sık sık etkiliyor.

RedHat 9 ve Fedora ile gelen Berkeley DB kütüphaneleri çekirdekteki NPTL (Native Posix Threads Library) desteğini kullanırlar.

RedHat tarafından dağıtılan çekirdeklerde bu özellik öntanımlı olarak eklenmiş olarak gelir zaten, fakat eğer kendi çekirdeğinizi derlediyseniz bu özelliği etkin kılmamış olabilirsiniz. Böyle bir durumda şuna benzer hata mesajları alabilirsiniz.

```
svn: Berkeley DB error
svn: Berkeley DB error while creating environment for filesystem tester/db:
Function not implemented
```

Bu aşağıdaki bir kaç yöntemden biri ile giderilebilir:

- db4'ü kendi kullandığınız çekirdeğe göre baştan derleyebilirsiniz.
- Bir RedHat 9 çekirdeği kullanabilirsiniz.
- Kullandığınız çekirdeğe NPTL yamaları uygulayabilirsiniz.
- NPTL desteğine sahip güncel bir çekirdek (`>= 2.5.x`) kullanabilirsiniz.
- Subversion'ı (Apache'yi) başlatmadan önce, `LD_ASSUME_KERNEL` değişkeninin `2.2.5` değerine ayarlı olup olmadığını kontrol edip, eğer öyleyse değişkeni kaldırın (**`unset LD_ASSUME_KERNEL`**). (Bu değişken genellikle RedHat 9 altından Wine ve WineX kullanmak için ayarlanır.)

Ayrıca NPTL desteğine sahip bir Berkeley DB kullanmak için de, NPTL desteğine sahip bir glibc kütüphanesi kullanmanız gerek ki, bu da glibc'nin i686 versiyonu anlamına gelir. (Bkz. `http://svn.haxx.se/users/archive-«2004-«03/0488.shtml`.)

3.18. Neden SVN günlük dosyasına Apache ile onaylatılan değişikliklerde, değişikliği yapan kullanıcı olarak "(no author)" ibaresi düşülüyor.

Apache ile 'anonymous' (kimliksiz) kullanıcıların arşiv üzerinde yazma haklarına sahip olmasına izin verirseniz, Apache sunucusu, **`svn`** istemcisinden kullanıcı adı istemeyecektir ve hatta, herhangi bir kimlik sorgulaması yapmadan kullanıcıya işlemini yapma izni verecektir. Bu nedenle, Subversion, onaylamanın kim tarafından yapıldığından haberdar olamayacağı için, günlük dosyalarında şöyle bir ibare yer alacaktır:

```
$ svn log
------------------------------------------------------------------------
rev 24:  (no author) | 2003-07-29 19:28:35 +0200 (Tue, 29 Jul 2003)
```

Apache'de erişim kısıtlamarının ayarlanması ile ilgili olarak Subversion kitabındaki Bir Arşivin Ağa Bağlanması[B71] bölümüne göz atınız.

3.19. Arada sırada "Access Denied" hataları alıyorum Windows'ta. Genelde rastgele zaman aralıklarında tekrarlıyor bu hatayı. Neden kaynaklı olabilir?

Bu daha çok dosya sistemi değişikliklerini denetleyen Windows servislerinden (anti–virüs yazılımları, indeksleme servisleri, COM+ Durum Bildiri Servisi) kaynaklı bir sorun. Sorun, Subversion tarafındaki bir hatadan kaynaklı olmadığından, bu da sorunun çözülmesini bizim için güç kılıyor. Konu hakkında yapılan araştırmanın bir son durum özetine buradan[B72] ulaşabilirsiniz. 7598. revizyonda bu rastlantının karşıya çıkma sıklığını azaltacak yeni bir yöntem geliştirildiğinden, eğer 7598'den daha düşük bir revizyona sahipseniz, bunu son sürüme yükseltmeniz tavsiye olunur.

3.20. FreeBSD üzerindeyken bazı komutlar (özellikle `svnadmin create`) zaman zaman donuyor. Neden kaynaklı olabilir?

Bu genelde sistemde var olan entropi eksikliğinden olur. Büyük olasılıkla sisteminizi sabit disk ve ağ kesmeleri gibi kaynaklarından entropi kazanacak şekilde ayarlamanız gerekmekte. Başta **`random`**(4) ve **`rndcontrol`**(8) olmak üzere, bu değişikliği etkileyecek komutların kılavuz (man) sayfalarına bakınız.

3.21. Arşivimi bir internet tarayıcısı ile sorunsuz görüntüleyebildiğim halde, `svn checkout` ile arşive erişmeye çalıştığımda "301 Moved Permanently" şeklinde bir hata alıyorum. Sorun neden kaynaklı olabilir?

Aldığınız hata, `httpd.conf` dosyanızın yanlış ayarlandığı anlamına geliyor. Bu hata genellikle Subversion'ın sanal konumunun aynı anda birbirinden farklı iki alanda tanımlanmasından doğmaktadır.

Örneğin, eğer arşivinizi `<Location /www/foo>` şeklinde belirttiyseniz ve `DocumentRoot` değişkeniniz de `/www` dizinine ayarlıysa başınız dertte demektir. `/www/foo/bar` dizinine herhangi bir istek geldiğinde Apache `/foo/bar` isimli *gerçek* bir dosyayı `DocumentRoot` altında tanımlı `/www` altında mı, yoksa `mod_dav_svn` için belirttiğimiz `/www/foo` için `/bar` altında mı bulması gerektiğine karar veremiyor. Ve genellikle böyle durumlarda ortaya çıkan "Moved Permanently" (kalıcı olarak kaldırılmıştır) hatasını düşüyor.

Çözümü için arşiv olarak belirttiğiniz konumun, başka bir tanımlamanın konumu ile kesişmediğine dikkat edin.

3.22. Bir dosyanın eski bir sürümüne `svn` ile bakmaya çalıştığımda "path not found" hatası alıyorum. Bunun sebebi ne olabilir?

Subversion'ın diğer güzel bir özelliği ise, arşivin kopyalama ve yeniden adlandırma işlemlerini anlayıp, eski bağlantıları koruması. Örneğin, `/trunk` dizinini `/branches/mybranch` şeklinde kopyalarsanız, arşiv, 'brach'deki herbir dosyanın `/trunk`'da bir öncelinin olduğunu bilir. **`svn log --verbose`** komutunu çalıştırırsanız kopyalama işleminin geçmiş kayıtlarını listeleyip, yeniden adlandırmayı görebilirsiniz:

```
r7932 | joe | 2003-12-03 17:54:02 -0600 (Wed, 03 Dec 2003) | 1 line
Changed paths:
   A /branches/mybranch (from /trunk:7931)
```

Ne yazık ki, arşiv tüm bu kopyalama ve yeniden adlandırma işlemlerinden haberdar olduğu halde, istemciler değildir. **`svn diff`**, **`svn merge`** ve **`svn cat`** gibi komutlar yeniden adlandırmaları anlayıp, takip etme yetisine sahip oldukları halde, bunu yapamazlar. Bu özellik post–1.0 sürümüne eklenmek üzere programlanmıştır. (Bkz. 1093. hata bildirimi[B73]) Mesela, **`svn diff`** ile `/branches/mybranch/foo.c` dosyasının iki eski sürümünü karşılaştırmak istediğinizde, yeniden adladırmadan dolayı, komut otomatik olarak aslında iki eski `/trunk/foo.c` sürümünün karşılaştırılması gerektiğini anlamayacaktır. Bunun yerine, 'branch' yolunun bir önceki revizyonda yer almadığı şeklinde bir hata alacaksınız.

Tüm bu çeşit problemler için kendiniz biraz koşturmak zorunda kalacaksınız. Bu nedenle, kendiniz yeniden adlandırmalardan haberdar olup, bunları **`svn log --verbose`** ile öğrenip açıkça **`svn`** istemcinize belirtmelisiniz. Örneğin,

```
$ svn diff -r 1000:2000 http://host/repos/branches/mybranch/foo.c
svn: Filesystem has no item
svn: '/branches/mybranch/foo.c' not found in the repository at revision 1000
```

komutu yerine

```
$ svn diff -r1000:2000 http://host/repos/trunk/foo.c
...
```

komutunu vermelisiniz.

3.23. HTTP Digest auth neden çalışmıyor olabilir?

Bu büyük ihtimalle Apache HTTP sunucusunun 2.0.48 ve öncesi sürümleri için geçerli olan, yaması mevcut, bir hatasından kaynaklanıyor. (Bkz. http://nagoya.apache.org/bugzilla/show_bug.cgi?id=25040) http://subversion.tigris.org/issues/show_bug.cgi?id=1608 adresine de bulgularınızın uyup uymadığını karşılaştırmak için göz atabilirsiniz.

3.24. AIX üzerinde `xlc` ile derlemeye çalıştığımda hata alıyorum. Neden kaynaklı olabilir?

Ortam değişkeni olan `CFLAGS`'a `-qlanglvl=extended` seçeneğini eklerseniz derleme işlemi **`xlc`** için biraz daha esnekleşip, sorunsuz gerçekleşecektir. Ayrıntılar için http://svn.haxx.se/dev/archive-«2004-«01/0922.shtml adresindeki iletiye ve bu ileti ile ilgili iletilere bakabilirsiniz.

3.25. Arşivden bir dizini (`-N` seçeneği ile) devirli olmayacak şekilde çalışma dizinime çektikten sonra, bazı alt dizinlerin görünmesini istiyorum. Fakat `svn up subdir` ile bunu yapamıyorum.

Bkz. İleti #695[B77]. **`svn checkout -N`** komutunun şu anki uyarlamasında bir takım eksikler mevcut. Bu ise eksik dosyaların olduğu arşiv kopyanızda, "tamsızlık" olarak sonuçlanıyor. Subversion geliştiricilerin hiçbiri böyle bir şeye bağımlılık duymasa da, görünürde bir çok CVS kullancısı bu paradigmaya bağımlı. Şu an itibari ile işleyiş yönteminizi değiştirmekten başka bir şansınız yok: Tüm arşivi kopyaladıktan sonra, arşivi istediğiniz dizinler kapsanacak şekilde daraltabilirsiniz.

3.26. `mod_dav_svn`'yi Windows üzerinde Apache ile çalıştırmaya çalıştığımda, `mod_dav_svn.so` dosyasını doğru yer olan `\Apache\modules` altına koyduğum halde, Apache'den modülü bulamadığına dair hata alıyorum.

Bu hata iletisi böyle bir durum için biraz yanıltıcı. Daha çok Apache `mod_dav_svn.so` tarafından gerek duyulan bir kaç DLL'nin yüklenememesi ile ilgili. Eğer Apache bir servis olarak çalışıyorsa, normal bir kullanıcı ile aynı `PATH` değişkenine sahip olmayacaktır. `libdb4*.dll, libeay32.dll` ve `ssleay32.dll` dosyalarının `\Apache\bin` ya da `\Apache\modules` altında olduğundan emin olun. Eğer belirtilen dizinlerin birinde değillerse, bir kopyalarına Subversion'ın kurulum dizini altından ulaşabilirsiniz.

Eğer bu sorununuzu hala çözmezse, Dependency Walker[B78] gibi bir araç kullanarak `mod_dav_svn.so`'nun çözülmemiş bağımlılıklarını görebilirsiniz.

3.27. Neden arşiv askılarım çalışmıyor. Dışarıdan program çağrıları gerektiği halde hiç de öyle bir çağrı yolluyor gözükmüyorlar.

Arşiv askılarından program çalıştırmaya çalışırken programların tam yollarını girmeyi deneyiniz. Subversion askıları çalıştırdığında, bulundukları ortam, Subversion'ı başlatan kullanıcınınki ile aynıdır ki bu, siz aynı programları elle çalıştırmayı denediğinizde kullandığınız ortamdan farklı bir ortam olabilir. Özel olarak: Unix üzerinde `$PATH`, Windows üzerinde `%PATH%`, beklediğiniz dizinlere sahip olmuyor olabilir.

3.28. `--diff-cmd` seçeneği ile `-u` seçeneğini kullandığımda neden uyarı alıyorum? `--extensions` seçeneği ile bunu etkisiz kılmaya çalıştığım halde olmuyor.

Dışarıdan bir **`diff`** uygulaması kullandığınızda, Subversion komut satırını gerçekten karmaşık bir hale sokar. İlk olarak belirtilen `--diff-cmd`'dir. Ardından `--extensions` (boşsa yoksayılır), `--extensions` belirtilmezse (ya da " şeklinde belirtilirse) `-u` gelir. Üçüncü ve dördüncüde, Subversion bir `-L` ekledikten sonra, ilk dosyanın adını girer (Örneğin: `project_issues.html (revision 11209)`). Beşinci ve altıncı alanlarda yeniden bir `-L` ve ikinci dosya adı... Yedinci ve sekizinciler ise birinci ve ikinci dosya isimleri... (Örneğin: `.svn/text-base/project_issues.html.svn-base` ve `.svn/tmp/project_issues.html.tmp`.)

Eğer seçtiğiniz **`diff`** uygulaması bu seçenekleri desteklemiyorsa, ufak bir kapsayıcı betik yazarak, bu seçenekleri göz ardı ettikten sonra asıl **`diff`** uygulamanızı çalıştırabilirsiniz.

Uyarı: Subversion çağrılan **diff** uygulamasının işlemesi için gönderdiği dosyaları değiştirmesini beklemez ve böyle bir durumda çalışma kopyanız içinden çıkılmaz bir hal alabilir.

Daha fazla bilgi için: İleti #2044[B79]

3.29. N'olamaz! Subversion istemcim parolalarımı düz bir metin dosyasında tuttuyor.

Sakin olup, derin bir nefes alın ilk önce. *(Canım dur daha karpuz kesecez :P)*

Her şeyden önce, kaydedilmiş parolalarınızın bulunduğu dizinin (Unix sitemlerde genellikle `~/.subversion/auth/` altındadır) haklarının 700, yani orada sadece sizin okuma hakkınızın olduğuna dikkat ediniz. Diskteki veriyi koruması için işletim sisteminize güvenin.

Eğer bu sizi gerçekten rahatsız ediyorsa, parola saklama özelliğini tamamen kapatabilirsiniz. Bunun için, svn 1.0 istemcisinde, ayar dosyanıza `store-auth-creds = no` satırını eklemeniz yeterli. svn 1.1 ve üstü istemcilerde ise daha özel bir alanı işaret eden `store-passwords = no` satırını kullanabilirsiniz. (Son durumda sunucu sertifikaları halen kaydedilmeye devam edecek yalnız.)

3.30. "*svn: bdb: call implies an access method which is inconsistent with previous calls*" hatası alıyorum. Bunu nasıl düzeltebilirim?

Berkeley DB 4.1'in arasıra kararsız davrandığı oluyor – 4.0 ve 4.2 sürümleri daha iyi. Bu hata mesajı 4.1'i genelde tek bir nedenden dolayı oluşan bir hatasının belirtisi.

Sorunun nedeni Subversion BDB–tarzı arşivi oluşturan tablolardan birinin veritabanı biçim alanında bozulma oluşması. Nedeni bilinmemekle birlikte, `btree` türünden `recno` türüne geçişi sağlayan tablolardan birini neredeyse her zaman kopyalar. En basitinden kurtarma işlemi için gerekli adımlar aşağıda anlatılmıştır – eğer bunlar başarılı olmazsa, Subversion Kullancıları E–Posta Listesi <users (at) subversion.tigris.org> ile temasa geçebilirsiniz.

- Hiçbir işlemin arşive erişmeye çalışmadığına emin olun.
- Şimdi, bir zip ya da tar dosyasına *arşivinizin yedeğini alın*.
- Arşivin `db` alt dizinine geçin.
- `rm __db.* log.*`
- **`db_dump -p -r copies > copies.dump`**
- `copies.dump` dosyasını bir metin düzenleyici yardımı ile açın. En tepeye yakın kısımdaki `type=recno` ibaresini `type=btree` ile değiştirin ve `re_len=` ile başlayan satırı silin.
- **`rm copies`**
- **`db_load copies < copies.dump`**
- **`svnadmin dump .. > ../../my-recovered.svndump`**
- Şimdi yeni bir arşiv oluşturun, daha demin oluşturduğumuz arşiv dökümünü (`my-recovered.svndump`) yükleyin ve gerekli askı betikleri ile ayar dosyalarını yeni arşive kopyalayın. Arşivdeki en yüksek revizyon numarasının olması gereken değer olduğuna emin olun.

3.31. `hotbackup` ile arşivimin yedeğini almaya çalıştığımda, `svnadmin` 2GB'tan büyük dosyalarda hata veriyor.

APR'nin Apache 2.0.x ve Subversion 1.x ile kullanılan önceki 0.9 dalındaki sürümleri 2GB'tan büyük dosyaların kopyalanmasına izin vermemekte. **`svnadmin hotcopy`** sorununu çözen bir düzeltme APR'ye 0.9.5+, Apache'ye de 2.0.50+ sürümlerinde eklendi. Bu düzeltme tüm platformlarda çalışmıyor, fakat Linux'da çalışıyor.

3.32. Henüz yeni onayladığım bir dosya için ilgili günlük kayıtlarını göremiyorum. Neden olabilir?

Farzedin ki, **svn checkout** ile içinde bir `foo.c` dosyası bulunan bir arşivin 7. revizyonunu (r7) çalışma kopyanız olarak indirdiniz. Dosya üzerinde değişiklikler yapıp ve bunları onayladınız. Bu noktada iki şey olur:

- Sunucudaki arşiv r8'e yükselir.
- Elinizdeki çalışma kopyasında sadece `foo.c` r8'e yükselir, diğer dosyalar r7'de kalır.

Şimdi *karma revizyonlu çalışma kopyası* denen şeye sahibiz. Sadece bir tek dosya r8'de, fakat diğer bütün dosyalar onaylanana ya da **svn update** denene kadar r7'de kalır.

```
$ svn -v status
   7        7 nesscg       .
   8        8 nesscg       foo.c
```

Eğer **svn log** komutunu hiçbir argüman vermeden çalıştırırsanız, komut şu anki dizinin günlük bilgisini ekrana döker. (Yukarıdaki listede '.' ile geçen.) Şu an bulunduğunuz dizin de hala r7'de olduğu için, r8 için günlük bilgisini göremezsiniz.

Son günlük bilgierini görmek için, şunları yapabilirsiniz:

1. **svn log -rHEAD**
2. **svn log URL** (URL, arşivin URL'sini gösteriyor.)
3. Sadece bir dosyanın günlük bilgisini görmek için: **svn log foo.c**
4. Tüm arşivinizi yükseltin ve haliyle tüm dosyalar r8'e geçeceğinden **svn log** komutunu kullanın.

4. Geliştirici Soruları

4.1. Bellek diski ('ram disk') üzerinde nasıl bütünlük sınaması yapabilirim?

Bkz. `http://svn.haxx.se/dev/archive-«2003-«02/0068.shtml`

5. Atıflar

5.1. Subversion tarafından kullanılan tüm HTTP yöntemleri neler?

Aşağıdaki e–posta her şeyi söylemektedir. Yazarın da altını çizdiği gibi, Subversion aslında henüz tüm WebDAV/DeltaV yöntemlerini kullanmamaktadır, ama muhtemelen günün birinde bu olacak. Bu yüzden eğer bir vekil sunucu kuruyorsanız, bunların hepsine izin verebilirsiniz:

```
From: Nuutti Kotivuori <naked (at) iki.fi>
Subject: Re: list of HTTP messages used by svn?
To: "Hamilton Link" <helink (at) sandia.gov>
Cc: <dev (at) subversion.tigris.org>
Date: Sat, 10 Aug 2002 13:51:52 +0300

Hamilton Link şunları yazmış:
>Önerebileceğiniz svn kullandığı tüm HTTP yöntemlerinin listesini
```

```
>bulabileceğim bir adres var mı? İlgili belgelerde
>(project_faq.html ve INSTALL dosyası) bulabildiğim kadarı ile,
>svn'nin kullandığı yöntemlerin listesi en azından şöyle:
>
>GET, PROPFIND, REPORT, OPTIONS, MERGE, MKACTIVITY, and CHECKOUT
>
>Fakat bunlar benim sadece özel olarak bulduklarım ve hepsinin
>kullanıldığından da tam olarak emin değilim ve pek de hakkında
>fikir yürütme yanlısı değilim.
>
>Eğer elimde tam bir liste olsaydı, şirketin vekil sunucularına
>bakan yöneticilere birçok kez yerine bir kere giderdim ve
>sorunsuz bir çıkış için gerekli değişiklikleri onlara bildirip
>proxy'de kesintisiz bir svn desteğine sahip olurdum.

http://www.webdav.org/deltav/WWW10/deltav-«intro.htm

Listenin bir kopyası burada:

HTTP/1.1: GET, HEAD, POST, PUT, DELETE, OPTIONS, TRACE, CONNECT

WebDAV: LOCK, UNLOCK, PROPFIND, PROPPATCH, COPY, MOVE, MKCOL

DeltaV: CHECKIN, CHECKOUT, UNCHECKOUT, VERSION-CONTROL, REPORT,
UPDATE, LABEL, MERGE, MKWORKSPACE, BASELINE-CONTROL, MKACTIVITY

Subversion bunların dışında ne başka bir yöntem kullanmatadır, ne de
bunların hepsini tam olarak kullanmaktadır. Fakat tüm WebDAV/DeltaV
desteğine sahip olmak diğer bir kaç rastgele seçimden daha iyi. Eğer
ayarlara sahip geçerli vekil sunucunuz Squid ise, o büyük ihtimalle
HTTP/1.1 ve WebDAV için gerekli her şeye sahiptir - ve ona sadece
DeltaV eklentilerini eklemeniz yeterli.

Bu listeyi şirketinizin vekil sunucu yöneticisine verebilir ve eğer
isterse daha fazla bilgi için RFC'leri kontrol edebileceğinin
açıklamasını yaparsınız.
```

5.2. 'bikeshed' de nedir?

Poul–Henning Kamp'ın freebsd–hackers'a gönderdiği iletiye bakınız:
http://www.freebsd.org/doc/en_US.ISO8859-«1/books/faq/misc.html#BIKESHED-«PAINTING

5.3. 'baton' tam olarak nedir?.

Subversion'ın kaynak kodu boyunca bir çok noktada 'baton'a bağ vardır. Bunlar işlevlere bağlam sağlayan **`void *`** veri yapılarıdır. Diğer API'lerde bu daha çok **`void *ctx`** ya da **`void *userdata`** şeklinde geçer. Subversion geliştiricileri, etrafta çok fazla dolandıkları için bu tür veri yapılarına 'baton' demeyi seçmişlerdir.

5.4. Arşiv kesilmiş ('wedged') denerek aslında ne kastediliyor?

Kesilmiş ('wedged') arşiv

Bir Subversion arşivi, çalışma ve depolama olmak üzere, birbirinden farklı iki iç bölümden oluşur. Kesilmiş bir arşiv, çalışma bölümü herhangi bir nedenden dolayı ulaşılamayan, fakat depo bölümü hala sağlam haldeki arşivdir. Bunun için, kesilmiş bir arşivde veri kaybı olmaz. Fakat arşivin ulaşılabilir kılınması için çalışma bölümünün düzeltilmesi gereklidir. Ayrıntılı bilgi için *buraya bakabilirsiniz* (sayfa: 22).

Zarar görmüş (‘corrupted’) arşiv
: Zarar görmüş bir arşiv ise, bahsedilen depolama bölümünde hasar oluşmuş bir arşivdir ve bu nedenle arşivde belli bir veri kaybı gözlenir.

Eric S. Raymond’un Argo Dosyasındaki (‘Jargon File’) ‘wedged’ tanımına bakabilirsiniz[B85].

Notlar

Belge içinde dipnotlar ve dış bağlantılar varsa, bunlarla ilgili bilgiler bulundukları sayfanın sonunda dipnot olarak verilmeyip, hepsi toplu olarak burada listelenmiş olacaktır.

(B2) http://subversion.tigris.org/

(B3) http://www.debian.org/social_contract#guidelines

(B4) http://svn.collab.net/repos/svn/trunk/HACKING

(B5) http://apr.apache.org/

(B6) http://www.sleepycat.com/

(B7) http://svnbook.red-«bean.com/html-«chunk/ch06.html

(B8) http://svnbook.red-«bean.com/html-«chunk/ch06.html

(B10) http://svn.collab.net/repos/svn/trunk

(B11) http://svn.collab.net/repos/svn/trunk/www

(B12) http://svn.collab.net/repos/svn/trunk/www/logo/subversion_logo.eps

(B13) http://svn.collab.net/repos/svn/trunk/www/logo/subversion_logo.ai

(B15) http://www.freenode.net/

(B17) http://svnbook.red-«bean.com/

(B19) http://svn.collab.net/repos/cvs2svn/trunk/README

(B20) http://public.perforce.com/public/revml/index.html

(B21) http://search.cpan.org/perldoc?VCP::Dest::svk

(B23) http://yazicivo.fateback.com/doc/project_links.html

(B25) http://svnbook.red-«bean.com/html-«chunk/ch06.html

(B28) http://www.zip.com.au/%7Eroca/ttssh.html

(B29) http://www.sleepycat.com/docs/ref/env/remote.html

(B30) http://web.mit.edu/ghudson/info/fsfs

(B31) http://nfs.sourceforge.net/nfs-«howto/server.html

(B33) http://subversion.tigris.org/servlets/ReadMsg?list=users\&msgNo=15194

(B34) http://www.sleepycat.com/docs/api_c/env_set_flags.html#DB_LOG_AUTOREMOVE

(B35) http://svnbook.red-«bean.com/html-«chunk/ch06s05.html

(B36) http://svnbook.red-«bean.com/svnbook-«1.1/ch05.html#svn-«ch-«5-«sect-«1.2.A

(B37) http://subversion.tigris.org/issues/show_bug.cgi?id=516

(B38) http://svnbook.red-«bean.com/html-«chunk/ch05.html

(B39) http://svnbook.red-«bean.com/svnbook/ch05.html#svn-«ch-«5-«sect-«1.2

(B40) http://svnbook.red-«bean.com/svnbook/ch05s02.html#svn-«ch-«5-«sect-«2.1

(B41) http://svn.collab.net/repos/svn/trunk/HACKING

(B42) http://svn.collab.net/repos/svn/trunk/HACKING

(B43) http://subversion.tigris.org/issues/show_bug.cgi?id=1328

(B45) http://tortoisesvn.tigris.org/

(B47) http://tortoisesvn.tigris.org/download.html

(B48) http://www.google.com/search?q=%22ssh-«agent%22

(B52) http://svnbook.red-«bean.com/svnbook-«1.1/ch07s02.html#svn-«ch-«7-«sect-«2.4

(B54) http://subversion.tigris.org/issues/show_bug.cgi?id=1974

(B55) http://svnbook.red-«bean.com/en/1.1/ch05s02.html#svn-«ch-«5-«sect-«2.1

(B56) http://svnbook.red-«bean.com/en/1.0/apc.html

(B59) http://svnbook.red-«bean.com/svnbook/ch02s03.html#svn-«ch-«2-«sect-«3.4

(B61) http://subversion.tigris.org/db42-«support-«patch.txt

(B62) http://svnbook.red-«bean.com/html-«chunk/ch02s03.html#svn-«ch-«2-«sidebar-«1

(B69) http://ethereal.ntop.org/

(B71) http://svnbook.red-«bean.com/book.html#svn-«ch-«5-«sect-«4

(B72) http://svn.haxx.se/dev/archive-«2003-«10/0136.shtml

(B73) http://subversion.tigris.org/issues/show_bug.cgi?id=1093

(B77) http://subversion.tigris.org/issues/show_bug.cgi?id=695

(B78) http://www.dependencywalker.com/

(B79) http://subversion.tigris.org/issues/show_bug.cgi?id=2044

(B85) http://catb.org/~esr/jargon/html/W/wedged.html

Bu dosya (subversion-sss.pdf), belgenin XML biçiminin TEXLive ve belgeler-xsl paketlerindeki araçlar kullanılarak PDF biçimine dönüştürülmesiyle elde edilmiştir.

20 Ocak 2007