# KTH-KRB

Kerberos 4 from KTH
For release 1.0.
1999

**Johan Danielsson**
**Assar Westerlund**
**last updated $Date: 1999/12/31 01:14:57 $**

Copyright (c) 1995-1999 Kungliga Tekniska Hgskolan (Royal Institute of Technology, Stockholm, Sweden). All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. Neither the name of the Institute nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE INSTITUTE AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE INSTITUTE OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

[注釈] 著作権 (c) 1995, 1996, 1997 Kungliga Tekniska Hgskolan (スウェーデン・ストックホルム王立技術研究所)。全ての権利は留保される。

ソースコードおよびバイナリ形式での再配布と利用は、変更の有無に関わらず、以下の条件を満たす限りにおいて認められる：

1. ソースコードの再配布には上記の著作権表示、この条件のリストならびに以下の放棄を含めること。

2. バイナリ形式での再配布には上記の著作権表示、この条件のリストならびに以下の放棄が提供される文書かつまたはその他の内容物の中に再生成されなければならない。

3. このソフトウェアの特徴や用途に言及するすべての宣伝材料は以下の告知を表示しなければならない：

この製品は Kunglia Tekniska Hgskolan と貢献者達により開発されたソフトウェアを含む。

4. 明文化された許可無しに当研究所の名前や貢献者達の名前を、このソフトウェアから派生した製品の裏書や販売促進のために使ってはならない。

このソフトウェアは当研究所と貢献者達により提供されたもの "そのまま (AS IS)" で、特定の目的の商用性及び適合性の暗黙の保証を含むがそれに限定されることなく、明示あるいは暗黙の保証は放棄される。直接的、

間接的、偶然、特別、代表的あるいは従属的な損害に (代用の物品あるいはサービスによる調達；効用、データあるいは利益の損失；あるいは業務の中断を含み、しかしそれらに制限されることなく) 対し、いかほど起ころうとも、いかなる責任の理論の上であれ、たとえ契約内、厳密責任、またはこのソフトウェアのいかなる使用により生じる民事犯 (不行き届きまたはその他によるものを含む) でも、たとえそのような損害の可能性について言及されていたとしても、当研究所または貢献者達は一切の責任を負わない。

Copyright (C) 1995 Eric Young (eay@mincom.oz.au) All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. All advertising materials mentioning features or use of this software must display the following acknowledgement: This product includes software developed by Eric Young (eay@mincom.oz.au)

THIS SOFTWARE IS PROVIDED BY ERIC YOUNG "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

[注釈] 著作権 (C) 1995 エリック・ヤング (eay@mincom.oz.au) 全ての権利は留保される。

ソースコードおよびバイナリ形式での再配布と利用は、変更の有無に関わらず、以下の条件を満たす限りにおいて認められる：

1. ソースコードの再配布には上記の著作権表示、この条件のリストならびに以下の放棄を含めること。

2. バイナリ形式での再配布には上記の著作権表示、この条件のリストならびに以下の放棄が提供される文書かつまたはその他の内容物の中に再生成されなければならない。

3. このソフトウェアの特徴や用途に言及するすべての宣伝材料は以下の告知を表示しなければならない：

この製品は Eric Young (eay@mincom.oz.au) により開発されたソフトウェアを含む。

このソフトウェアは ERIC YONG により提供されたもの "そのまま (AS IS)" で、特定の目的の商用性及び適合性の暗黙の保証を含むがそれに限定されることなく、明示あるいは暗黙の保証は放棄される。直接的、間接的、偶然、特別、代表的あるいは従属的な損害に (代用の物品あるいはサービスによる調達；効用、データあるいは利益の損失；あるいは業務の中断を含み、しかしそれらに制限されることなく) 対し、いかほど起ころうとも、いかなる責任の理論の上であれ、たとえ契約内、厳密責任、またはこのソフトウェアのいかなる使用により生じる民事犯 (不行き届きまたはその他によるものを含む) でも、たとえそのような損害の可能性について言及されていたとしても、著者または貢献者達は一切の責任を負わない。

Copyright (c) 1983, 1990 The Regents of the University of California. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. All advertising materials mentioning features or use of this software must display the following acknowledgement: This product includes software developed by the University of California, Berkeley and its contributors.

4. Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

[注釈] 著作権 (c) 1983, 1990 カリフォルニア大学本校。全ての権利は留保される。

ソースコードおよびバイナリ形式での再配布と利用は、変更の有無に関わらず、以下の条件を満たす限りにおいて認められる：

1. ソースコードの再配布には上記の著作権表示、この条件のリストならびに以下の放棄を含めること。

2. バイナリ形式での再配布には上記の著作権表示、この条件のリストならびに以下の放棄が提供される文書かつまたはその他の内容物の中に再生成されなければならない。

3. このソフトウェアの特徴や用途に言及するすべての宣伝材料は以下の告知を表示しなければならない：この製品はカリフォルニア大学バークレー校とその貢献者達により開発されたソフトウェアを含む。

4. 明文化された許可無しに本校の名前や貢献者達の名前を、このソフトウェアから派生した製品の裏書や販売促進のために使ってはならない。

このソフトウェアは本校と貢献者達により提供されたもの "そのまま (AS IS)" で、特定の目的の商用性及び適合性の暗黙の保証を含むがそれに限定されることなく、明示あるいは暗黙の保証は放棄される。直接的、間接的、偶然、特別、代表的あるいは従属的な損害に (代用の物品あるいはサービスによる調達；効用、データ

あるいは利益の損失；あるいは業務の中断を含み、しかしそれらに制限されることなく) 対し、いかほど起ころうとも、いかなる責任の理論の上であれ、たとえ契約内、厳密責任、またはこのソフトウェアのいかなる使用により生じる民事犯 (不行き届きまたはその他によるものを含む) でも、たとえそのような損害の可能性について言及されていたとしても、本校または貢献者達は一切の責任を負わない。

Copyright (C) 1990 by the Massachusetts Institute of Technology

Export of this software from the United States of America is assumed to require a specific license from the United States Government. It is the responsibility of any person or organization contemplating export to obtain such a license before exporting.

WITHIN THAT CONSTRAINT, permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation, and that the name of M.I.T. not be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission. M.I.T. makes no representations about the suitability of this software for any purpose. It is provided "as is" without express or implied warranty.

[注釈] 著作権 (C) 1990 マサチューセッツ工科大学 (M.I.T.) による

このソフトウェアのアメリカ合州国からの輸出には合州国政府からの特定のライセンスが必要と仮定されます。誰であれどんな組織であれ輸出を企てる者は輸出を行なう以前にこのような許可を得ることが義務づけられている。

拘束範囲, このソフトウェアとその文書を、如何なる目的でも、無料で、使用許諾書無しに、使用、複写、修正そして配布することへの許可を、上記の著作権表示がすべての複写に現れて、かつ上記の著作権表示とこの許諾表示が補助文書に現れることにより、かつ M.I.T. の名前が明文化された許可無しにソフトウェアの配布の宣伝や人気取りのために使われることなく提供される限りにおいて認められる。M.I.T. はこのソフトウェアのいかなる目的への適合性も表明はしない。擦り込みあるいは暗黙の保証はなく、"そのまま (as is)" で供給される。

Copyright 1987, 1989 by the Student Information Processing Board of the Massachusetts Institute of Technology

Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation, and that the names of M.I.T. and the M.I.T. S.I.P.B. not be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission. M.I.T. and the M.I.T. S.I.P.B. make no representations about the suitability of this software for any purpose. It is provided "as is" without express or implied warranty.

[注釈] 著作権 1987, 1989 マサチューセッツ工科大学 (M.I.T.) 学生情報処理掲示版 (S.I.P.B.) による

このソフトウェアとその文書を、如何なる目的でも、無料で、使用許諾書無しに、使用、複写、修正そして配布することへの許可を、上記の著作権表示がすべての複写に現れてかつ上記の著作権表示とこの許諾表示が補助文書に現れることにより、かつ M.I.T. と M.I.T. S.I.P.B. の名前が明文化された許可無しにソフトウェアの配布の宣伝や人気取りのために使われることなく提供される限りにおいて認められる。M.I.T. と M.I.T. S.I.P.B. はこのソフトウェアのいかなる目的への適合性も表明はしない。擦り込みあるいは暗黙の保証はなく、"そのまま (as is)" で供給される。

Copyright 1992 Simmule Turner and Rich Salz. All rights reserved.

This software is not subject to any license of the American Telephone and Telegraph Company or of the Regents of the University of California.

Permission is granted to anyone to use this software for any purpose on any computer system, and to alter it and redistribute it freely, subject to the following restrictions:

1. The authors are not responsible for the consequences of use of this software, no matter how awful, even if they arise from flaws in it.

2. The origin of this software must not be misrepresented, either by explicit claim or by omission. Since few users ever read sources, credits must appear in the documentation.

3. Altered versions must be plainly marked as such, and must not be misrepresented as being the original software. Since few users ever read sources, credits must appear in the documentation.

4. This notice may not be removed or altered.

[注釈] 著作権 1992 Simmule Turner と Rich Salz. 全ての権利は留保される。

このソフトウェアはアメリカ電信電話会社 (AT&T) あるいはカリフォルニア大学本校のいかなるライセンスの対象となるものではない。

このソフトウェアをいかなる会社のシステムのいかなる目的のために利用することへの許可を、以下の制限のもと、誰に対してでも認める。

1. 著者らは、このソフトウェアを使う上で起こるいかなることにも、もしそれがこのソフトウェアの欠陥によって起こることであっても、責任を持たない。

2. このソフトウェアの起源は、明確な主張によりあるいは省略によるものであっても誤表記されてはならない。ほとんどのユーザはソースを読むことが少ないので出典表記は文書の中に現れなければならない。

3. 改定版でも、基のソフトウェアと誤表記されることなく、このようにはっきりとマークされなければならない。ほとんどのユーザはソースを読むことが少ないので出典表記は文書の中に現れなければならない。

4. この印書きは削除されたり、置き換えられたりしてはならない。

# 1 Introduction

この文書は、スウェーデンのストックホルム王立技術研究所 (Kungliga Tekniska H$^s$*kolan*) から配布される *Kerberos4* について文書化を試みたものです。この配布版は *eBones* を基にしてますが、色々な改良がされています。このプログラムは、ごく最近の *UNIX* 互換システムであればどこでも動くはずです。

UNIX に加えて、以下のプラットホームでも部分的には動作します：

- OS/2、EMX が必要。
- Windows 95/NT、gnu-win32 が必要 (いくつかの魔法のライブラリーは Microsoft C でコンパイルしたほうが良い)。

基本的には、ANSI C コンパイラと、dbm ライブラリ ( サーバ側 ) と、 BSD ソケットのある POSIX 準拠のシステム上であれば動作するはずです。

ウェブページが `http://www.pdc.kth.se/kth-krb/` にありますのでご利用下さい。

## Bug reports

もし、プログラムを構築できなかったり、構築したプログラムが思うように動かなかったりした場合は、バグレポートを送って下さい。バグレポートの送り先は<kth-krb-bugs@pdc.kth.se>にして下さい。必ず、どういうマシンとオペーレティング・システム (バージョンも) とでプログラムを走らせたかと、何が起きてそれについてどのようなことが起こっていたと思うかと、それを再現するための例、および、その例を実行した時の出力を付けて下さい。それから、もし、その問題に対するパッチがあればそれも含むめた情報を下さい。すべてのパッチは `diff -u`か `diff -c`で作って下さい。バグレポートがより詳細であれば、その分、より早く我々もそれを再現し、理解し、修正することができます。

バグレポート以外の助言やコメントなども歓迎します。送り先は、<kth-krb@nada.kth.se> です。

# 2 What is Kerberos?

> Now this Cerberus had three heads of dogs,
> the tail of a dragon, and on his back the
> heads of all sorts of snakes.
> — Pseudo-Apollodorus Library 2.5.12

ケルベロス (Kerberos) はネットワーク上でユーザとサービスを認証するためのシステムです。そのシステムはネットワークが “安全でない” という仮定にたって築かれています。たとえば、ネットワークを越えて送られたデータは抜き取られたり、変更されたり、アドレスを偽らされたりすることがあり得ます。それゆえ、ネットワークを越えたデータを認証目的に使うことはできません。

ケルベロスは信頼できる第三者機関のサービス (a trusted third-party service) です。これは、ネットワーク上のすべての実体 (entities:ユーザとサービス、通常 プリンシパル *(principals)* と呼ばれる) により信頼される第三者機関 (ケルベロス・サーバ) があることを意味します。すべてのプリンシパルは、ケルベロス・サーバと秘密のパスワード (または鍵) を共有し、これでプリンシパルはケルベロス・サーバからのメッセージが本物であることを証明することが可能になります。こうして、ケルベロス・サーバ、ユーザおよびサービスを信任することで、お互いを認証し合うことができます。

## 2.1 Basic mechanism

ケルベロスの中で、プリンシパルは彼らが要請をする誰かに彼らであることを証明するために 切符 *(ticket)* を使います。次の例では、$A$ は認証交換のイニシエータ、通常はユーザで、そして、$B$ は $A$ が使いたいと願うサービスです。

特定のサービスのための切符を手に入れるために、$A$ は切符の依頼をケルベロスサーバに送ります。その依頼は基本的に $A$ の名前と $B$ の名前とを含んでいます。ケルベロス・サーバは $A$ と $B$ の両方が正当なプリンシパルであることを確認します。

プリンシパルの正当性が証明されると、$A$ と $B$ の名前、$A$ のネットワークアドレス ($A$<*addr*>)、現在の時刻 ($t$<*issue*>)、切符の寿命 (*life*)、および、秘密の セッション鍵($K$<*AB*>) を含むパケットを作ります。このパケットは $B$ の秘密鍵で暗号化されます ($K$<*B*>)。実際の切符 ($T$<*AB*>) は次のように見えます。

({$A$, $B$, $A$<*addr*>, $t$<*issue*>, *life*, $K$<*AB*>}$K$<*B*>)

$A$ への返信は、切符 ($T$<*AB*>)、$B$ の名前、現在の時刻、切符の寿命、および、セッション鍵で、すべて $A$ の秘密鍵 ({$B$, $t$<*issue*>, *life*, $K$<*AB*>, $T$<*AB*>}$K$<*A*>) で暗号化されます。$A$ は後で使うためにその返信を解読し保持します。

メッセージを $B$ へ送る前に、$A$ は $A$ の名前, $A$ のアドレス, 現行時刻, および $A$ に選ばれた “チェックサム” からなる認証子をつくります。それらは、すべて秘密のセッション鍵 ({$A$, $A$<*addr*>, $t$<*current*>, *checksum*}$K$<*AB*>) で暗号化されてます。これはケルベロス・サーバから $B$ によって受け取られる切符と供に送られます。受け取る際に、 $B$ は $B$ の秘密鍵でその切符の暗号を解読します。その切符には、認証子を暗号化したセッション鍵が含まれているので、ここで $B$ は暗号化された認証子を解読できるようになります。$A$ が本当に $A$ であることを証明するために、$B$ はここで切符の内容をその切符の認証子と比較しなければなりません。もしすべてが合致すれば、 $B$ はここで、$A$ が確かに認証されたと考えます。

## 2.2 Different attacks

### Impersonating A

一人の偽者 $C$ がネットワークを越えて転送される認証子と切符とを盗むことができ、それを使って $A$ に成りすますことができるとします。切符の中のアドレスと認証子が加えられて、この攻撃を達成することをより難

かしくしてます。$C$ が成功するためには、$A$ と同じマシンを使うか、パケットのソース・アドレスを偽る必要があります。認証子に時刻印を含めることにより、 $C$ が攻撃をするための十分な猶予は無くなります。

### Impersonating B

$C$ が $B$ のネットワーク・アドレスを偽装することができ、$A$ が $B$ (彼女) の信任状を送ると、$C$ はそれをただ単に証明するふりをすることができます。$C$ には $B$ (彼女) が $A$ に語りかけようとしていることがわかりません。

## 2.3 Defense strategies

サーバ側に リプレー・キャッシュを加えることは可能でしょう。考え方は、最後の数分間に送られた認証子を保存し、そうすると、誰かが既に使われたメッセージを再送しようとしたときに、$B$ はそれを検知することができます。これは、幾分か (性能に関してほとんど) 実用的でなく、Kerberos 4 には有りません； MIT Kerberos 5 は持っています。

$B$ を認証するために、$A$ は、セッション鍵にアクセスしようとしているのが $B$ であることを証明する何かを $B$ が 送り返すことを依頼するでしょう。この例が、$A$ が認証子の一部として送るチェックサムです。ある典型的な手順はチェックサムに 1 を加え、それをセッション鍵で暗号化して $A$ に送り返すことです。これは、相互認証 と呼ばれます。

セッション鍵は暗号学的チェックサムを $A$ と $B$ の間で送られるメッセージに加えるためにも使うことができます (メッセージ保全として知られます)。暗号化も加えられます (メッセージの内密性)。これがおそらくすべての場合において最良のアプローチです。

## 2.4 Further reading

ケルベロスについての元の論文は、1988 年の Jennifer Steiner, Clifford Neuman および Jeffrey I. Schiller による *Kerberos: An Authentication Service for Open Network Systems* です。

あまり技術的にならない記述は 1988 年の Bill Bryant による *Designing an Authentication System: a Dialogue in Four Scenes* です。

これらの他、幾つかの文書が我々のウェブページ上で見つけられます。

# 3 Installing programs

配布されているソースコードから構築するか、あるいは、もしマシンに合った実行モジュール (binaries) が用意されていればそれをインストールするかを選択できます。

もちろん、ソースからの構築をお奨めしますが、コンパイル済みの実行モジュールを使った方がより簡単です。マシンに合った実行モジュールが無かったり、特別な設定を行ないたいならば、ソースからコンパイルします。

## 3.1 Installing from source

このソフトウェアを構築するには、配布されたアーカイブを展開 (un-tar) して、`configure` スクリプトを走らせます。

正しくコンパイルするためには、`gcc` のような、ANSI C コンパイラが必要です。他のコンパイラでも動きますが、あまりにも、"ANSI 準拠" に設定し過ぎると、一部のコードでは、標準インクルード・ファイルがないために失敗するでしょう。

構築を個々のツリーで行なうには、ツリーの基のディレクトリで `configure` を走らせますが、Make が VPATH を正しく理解するようにする必要があります。GNU Make ならうまくできます。

全てが構築できたら (構築をする場所により数分からあるいはもっと長い時間を要します)、*`make install`*により (root で走らせて) 全てを '`/usr/athena`' にインストールできます。他の場所にインストールすることも可能ですが、あまりお奨めしません。もし、そうしたければ、`configure` で '`--prefix=/my/path`' を付けて走らせます。

既定 (default) の振舞を変えるためには、configure の際に以下のオプションを指定できます:

*`--enable-shared`*
: 共有バージョンの Kerberos ライブラリを作成します

*`--with-cracklib=`*dir
: パスワードの品質制御に cracklib を使います `kadmind`. このオプションにはパッチ `ftp://ftp.pdc.kth.se/pub/krb/src/cracklib.patch` の当たった cracklib が必要です。

*`--with-dictpath=`*dictpath
: これは cracklib が使うべきディレクトリです。

*`--with-socks=`*dir
: もし、ファイアウォールを通過させる必要がある場合に、SocksV5 プロトコル (*RFC 1928*) を使っていれば、socks-support で構築可能です。socks5 がインストールされているディレクトリを *dir* に指定してください。socks についてのもっと詳しい情報は、`http://www.socks.nec.com/` をご覧下さい。

*`--with-readline=`*dir
: 履歴 / 行編集を `ftp` や `kadmin`で可能にするには、どのバージョンの readline 使ってもできます。もし、readline をインストールされていても、configure でうまく見つけられないときにこのオプションを使います。configure は `libedit` も捜します。もし、ライブラリが見つからない時は、バンドルしてある `editline` を使うことになるでしょう。

*`--with-mailspool=`*dir
: 構成の過程で、そのマシンが受け取ったメールをどこに貯めておくかを特定しようとします。大抵は '`/usr/spool/mail`' か '`/var/mail`' です。もし、それがうまく機能しなかったり、一般的でないディレクトリにメールを貯めている場合は、このオプションを使って、メール・スプール・ディレクトリがある場所を示すことができます。このディレクトリは `popper` のみがアクセスし、`login` 時にメールをチェックします。

*`--with-hesiod=`*dir
: `push` で、Hesiod のサポートをします。このオプションにより、ヘシオド・ライブラリを使ってユーザのメールを郵便局に置こうとします。

*--with-mkey=file*
: 親 (master) 鍵をここに置きます。既定では、‘/.k’です。

*--with-db-dir=dir*
: ケルベロス・データベースが保存されている場所で、既定では ‘/var/kerberos’ です。

*--without-berkeley-db*
: もし、Berkeley DB がインストールされていたとしても、dbm の方が好ましい場合があります。もし、ケルベロスをすでに走らせていている場合は、dbm データベースから db への簡単な変換の方法が無いため (古いデータベースをダンプしてそれから、新しい実行モジュールでロードする必要があります)、このオプションは便利です。

*--without-afs-support*
: AFS んサポートをしません。

*--with-afsws=dir*
: AFS クライアントがインストールされている場所を指定します。既定では ‘/usr/afsws’ です。

*--enable-rxkad*
: rxkad ライブラリを構築します。

*--disable-dynamic-afs*
: AIX では実行時にロードされる共有ライブラリにより AFS のサポートが構成されます。このオプションではこれを不可能にし、静的なシステム呼出と結合します。これを行なうとカーネルの中に AFS を持たないマシン上では ( 例えば、起動時に AFS モジュールのロードに失敗したとき ) 、構築された実行モジュールをクラッシュさせます。

*--with-mips-api=api*
: このオプションにより Irix 上で異なるタイプの実行モジュールを作ることが可能になります。指定可能な値は *32*, *n32*, および *64* です。

*--enable-legacy-kdestroy*
: このオプションはコンパイル時に AFS トークンを破壊しない kdestroy を作ります。

*--disable-otp*
: OTP (see Chapter 5 [One-Time Passwords], page 23) ライブラリとプログラムを構築しないで、また、アプリケーション・プログラムでの OTP のサポートをしないようにします。

*--enable-match-subdomains*
: 通常は、ホスト ‘host.domain’ はレルムの ‘DOMAIN’ の一部と考えられます。このオプションにより、‘host.sub.domain’ や ‘host.sub1.sub2.domain’ などの書式のホストをレルムの ‘DOMAIN’ の一部と考えるようにできます。

*--enable-osfc2*
: OSF/1 上で 拡張 C2 セキュリティを使えるようにします。See Section 3.4.1 [Digital SIA], page 7.

*--disable-mmap*
: mmap システム呼出を使わないようにします。通常は、mmap があれば、configure はそれが唯一動くものとして使われていることを見つけます。このオプションは、それがどうしてもうまくゆかない時に使って下さい。

*--disable-cat-manpages*
: 整形済のオンライン・マニュアルをインストールしません。

## 3.2 Installing a binary distribution

実行モジュールの配布は、‘/usr/athena’にインストールされることを想定して、動かないもしれませんので他の場所にインストールすることは奨めしません。他の場所にインストールせざるを得ないときは、‘/usr/athena’ から、インストールされたディレクトリにシンボリックリンクを張ると良いでしょう。

## 3.3 Finishing the installation

インストールのとき root に UID 設定 (setuid) しなければならないプログラムは `su` だけです。

もし、`rlogin` や `rsh` が root に UID 設定 (setuid) されてしまうと、ケルベロス化されたプログラムがなんらかの理由で失敗したときのように、ケルベロスを使わないプロトコルに戻ってしまいます。古いプロトコルではセキュリティのために予約済のポートを使っているため、プログラムは root に UID 設定されていなくてはなりません。もし、こうすることが必要無ければ、UID 設定ビットを外すことを検討して下さい。

`login` は、いつも root が走らせるので、UID 設定される必要はありません (ユーザは `login` ではなく `su` を使うべきです)。root に UID 設定されないまま、ユーザが走らせたときは、何らかの手がかりとなる情報が表示されます。

ユーザが走らせることを意図したプログラムは '`/usr/athena/bin`' にあります。ユーザに '`/usr/athena/bin`' をパスに含めるよう知らせて下さい。あるいは、実行モジュールを適当な場所にコピー、または、シンボリック・リンクしてください。おそらく、使いたいプログラムは :

`kauth`/`kinit`, `klist`, `kdestroy`, `kpasswd`, `ftp`, `telnet`, `rcp`, `rsh`, `rlogin`, `su`, `rxtelnet`, `tenletxr`, `rxterm`, および `xnlock` でしょう。 もし AFS をお使いであれば、`afslog` と `pagsh` もまた便利です。管理者は、'`/usr/athena/sbin`' にある `kadmin` と `ksrvutil` を使うと便利です。

`telnetd` と `rlogind` は `login` が '`/usr/athena/bin`' (あるいは '`--prefix`' に指定したパス) にあることを前提にしています。もし、なんらかの理由で `login` を移動したければ、telnetd と rlogind を '`inetd.conf`' の中に設定するときに、 '`-L`' 切替で新たな場所を指定しなければならないでしょう。

システムの既定の `login` をケルベロス化された `login` と置き換えることも可能なはずです。しかしながら、あるシステムでは login が、我々の login ではしないような、ある非常に重要な (我々がいくらやろうとしてもできないような) トリックを演じることを仮定してます。ですから、マシンの置換えを全て行なう前に何がおこるかを試してみましょう。また、提供されている認証モジュール (see Section 3.4 [Authentication modules], page 7) の一つを使えることも試してみるべきです。

我々の使った `login` プログラムは、NetBSD の標準ログインプログラムの初期のころのもから派生したものです。それを沢山の奇怪なシステム (Solaris, SunOS, IRIX, AIX, その他) で使うために、他の多くのシステムの login の特徴をもって "改善" されました。そのうちの幾つかの特徴は本当に便利で、その他のシステムでも使いたくなるようなものです。

'`/etc/fbtab`'
'`/etc/logindevperm`'
: 特定の端末からユーザがログインするときにあるデバイスのオーナ変更 (chown) を許します。通常は誰かが /dev/console でログインしている時に、 '`/dev/mouse`', '`/dev/kbd`', その他のデバイスのオーナを変更するのに使います。

  '`/etc/fbtab`' は SunOS でのファイル名で最初に試されます。もし、その名前のファイルがなければ、Solaris でのファイル名 '`/etc/logindevperm`' が試されます。

'`/etc/environment`'
: このファイルは、ユーザがログインしたときにどの環境変数が設定されるべきかを指定します (AIX 流)。

'`/etc/default/login`'
: ほとんど '`/etc/environment`' と同じですが、System V 流です。

'`/etc/login.access`'
: 誰がどこからどの tty にログインできるかを制御するのに使われます (Wietse Venema より)。

それぞれのユーザは認定ファイル '`~`*user*`/.klogin`'を持つことができます。は、どのプリンシパルがそのユーザとしてログインできるかを決定します。IP と特権ポートを基にした認証でないことを除けば、'`~user/.rhosts`' と似ています。もし、このファイルが存在しなければ、'`user@LOCALREALM`' のユーザ自身のログインが許されます。それに追随するローカル・レルム (see Section 4.1.3 [Install the configuration files], page 10) も、ここで与えられます。もし、ファイルが存在すれば、ローカルのユーザ *user* としてログインすることが許される、他のプリンシパルが含まれるべきです。

ほとんどのデーモン ( rlogind,rshd, ftpd, telnetd, popper, kauthd, ならびに kxd ) は、このファイルと協議して

そのプリンシパルが、要求するサービスを受けることを許されるかを決定します。また、su でも、誰が root になることを許されているかについてのアクセス制御リスト (ACL) を維持する良い方法としても使われています。

'~root/.klogin' に次のもの :

```
nisse.root@FOO.SE
lisa.root@FOO.SE
```

が含まれることを仮定します。nisse と lisa というユーザは、どちらも root インスタンスのパスワードを入力することにより root に su することができます。 もし、'~root/.klogin' に含まれないユーザは su に失敗し、誰が su できるかについての普通の方針に戻ります。nisse と lisa は、自分たちの root インスタンスのチケットを持てば、たとえば、telnet で root 権限でログインすることができることもここに記しておきます。

## 3.4 Authentication modules

異なる認証機構を持つことの問題は幾つかのベンダーにより認識されてます、そして、幾つかの解決策が加えられてきました。ほとんどの場合は、これらの解決策ではある種の共有モジュールを動作時にロードして内包します。これらのシステムの幾つかのモジュールは 'lib/auth' で見つけられます。現時点では、Digital の SIA、 Solaris や Linux の PAM、 それと、 IRIX の ('lib/auth/afskauthlib' にある) login と xdm です。

### 3.4.1 Digital SIA

SIA モジュールをインストールする方法はどのバージョン OS の上で走らせるかに依存します。Tru64 5.0 には 'siacfg' という新しいコマンドがあり、この過程をとても単純にします。もし、このプログラムがあれば、ただ単につぎのように走らせるだけです:

```
siacfg -a KRB4 /usr/athena/lib/libsia_krb4.so
```

古いバージョン、あるいはすべてを自分の手で行いのであれば、以下のように行います ( ただし、Tru64 5.0 ではたしかめていません )。

- 'libsia_krb4.so' が '/usr/athena/lib' の下にあって、利用できることを確認して下さい。もし、'/usr/athena' がローカル・ディスク上になければ、それらは、'/usr/shlib' の下でも、あるいは別のある場所でも構いません。もしそうするのであれば、'krb4_matrix.conf' を編集して、新しい場所を反映させる必要があるでしょう (もし '/usr/athena' 以外にインストールした時は、この作業が必要となります。)。もし、共有ライブラリで構築すると、共有ファイル 'libkrb.so', 'libdes.so',+'libkadm.so', および 'libkafs.so' をローダがそれらを見つけられる ('/usr/shlib' のような) 場所にコピーしておかなくてはなりません。
- 'krb4_matrix.conf' を '/etc/sia' にコピーします (編集されてもかまいません)。
- 'security.patch' を '/sbin/init.d/security' に適用します。
- *rcmgr set SECURITY KRB4* と *rcmgr set KRB4_MATRIX_CONF krb4_matrix.conf* を実行して KRB4 セキュリティを行なうようにします。
- Compaq(Digital) の人はマシンをリブートするべきだと言うでしょうが、本当は必要ないはずです。通常は、*/sbin/init.d/security start* を走らせるだけです。(そして SIA を使う xdm のような任意のアプリケーションをリスタートさせます。)

ローカル・パスワードの ('root'のような) ユーザは安全にログインできるはずです。Compaq(Digital) の xdm を使う時、(xdm が環境を壊す (zaps) ので) 渡すべき 'KRBTKFILE' 環境変数が渡されません。その代わりに、'/usr/lib/X11/xdm/Xsession' の中に 'KRBTKFILE' を正しい値に設定し直さなくてはなりません。次のような行を加えます。

```
KRBTKFILE=/tmp/tkt‘id -u‘_‘ps -o ppid= -p $$‘; export KRBTKFILE
```

もし、CDE を使っているなら、dtlogin がどの追加環境をエキスポートするべきかを指定することを許します。‘KRBTKFILE’ をこのリストに加えるには、‘/usr/dt/config/Xconfig’ を編集して、‘exportList’ の定義を捜してください。おおよそ次のようになるでしょう。

```
Dtlogin.exportList:     KRBTKFILE
```

### Notes to users with Enhanced security

Digital の ‘ENHANCED’ (C2) セキュリティ、 およびケルベロスは二つのことなる問題を解決します。ローカルセキュリティを扱う C2 は、誰が何をできるかや監査、および、同様なことをする上でより良い制御を加えます。ケルベロスはネットワーク・セキュリティを取り扱います。

C2 セキュリティがケルベロスと供に動くようにするためには、以下のことをする必要があります。

- 全ての ‘krb4_matrix.conf’ の仕上がりを上記の方針で ‘krb4+c2_matrix.conf’ に置き換えます。
- “裏打うち”(vouching) を ‘既定’データベースで可能にする必要があります。これは、OSF C2 モジュールに SIA モジュールを信用させるので、C2 パスワードを与えること無しでログインができます。これをするには、‘edauth’ を使って */usr/tcb/bin/edauth -dd default* で既定の記載編修し、もし未だ無ければ ‘d_accept_alternate_vouching’機能を加えます。
- ローカル C2 パスワードを持って 無い それぞれのユーザには、パスワードの期限フィールドを零に設定するべきです。この設定は、それぞれのユーザ毎に行なうか、あるいは ‘既定の’ 表の中で行なえます。これを行なうには ‘edauth’ を使って ‘u_exp’ 機能を ‘u_exp#0’ に設定 ( あるいは変更 ) します。
- 出荷された ‘login’, ‘rcp’, および ‘rshd’ が、特別な C2 のマジック (無効にされたアカウントの色々な形式の確認のようなもの) を全くしないことに気づく必要もあるでしょう。そうして、もし、これらの機能を頼りたいならば、これらのプログラムを使うべきではありません。もし、‘--enable-osfc2’ で構成 (configure) をすれば、これらのプログラムを使えますが、しかし、ログイン UID を設定します。それでも: リスクを覚悟の上お使い下さい。

現状では、 ‘su’ が裏打うちフラグを受け付けませんので、思ったようには働きません。

また、ケロベロス化された ftp は C2 パスワードでは動きません。これは Digital の ftpd と我々のものを別のポートで使うことにより解決できます。

もしこれらの変更をすると、かなり確実に C2 システムの要求を完全には満たさ 無く なることを 忘れないで下さい。もし、C2 が欲しくても、誰かが無理矢理使わせようとしない限り、しばらくはあきらめて下さい。もし、それ以上できないくらいシステムをもっと安全にしてセキュリティを高めたければ、おそらくもっと安全なシステムを手に入れるでしょう。既にはっきりと分かるようなパスワードを送ったりはしてないでしょう。

## 3.4.2 IRIX

IRIX のサポートは Transarc の ‘afskauthlib.so’ と互換のモジュールです。このライブラリを使う全てのプログラムは動くはずで、‘login’ と ‘xdm’を含むはずです。

インターフェースについてはあまり文書化されていませんが、‘libkafs.so’, ‘libkrb.so’ および ‘libdes.so’ を ‘/usr/lib’ にコピーするか、‘afskauthlib.so’ を静的に結合して構築するようです。

ファイル ‘afskauthlib.so’ 自身は ‘/usr/vice/etc’ か ‘/usr/afsws/lib’ かあるいは、(それがある) 現行のディレクトリに置かれていても良いようです。

IRIX 6.4 以降のすべてのプログラムは (‘xdm’ と ‘login’ とを含みます)N32 オブジェクト・フォーマットですが、それより古いバージョンでは O32 オブジェクトです。それらが動くためには、‘afskauthlib.so’ ライブラリがそれをロードしようとしているプログラムと同じオブジェクト・フォーマットでなくてはなりません。このためには、既定の N32 に加えて O32 のための構成と構築をする必要があるということです。

これ以外には、設定ファイルはありませんので “まさに動く” はずです。

### 3.4.3 PAM

現時点で PAM モジュールは Linux と SunOS5(いわゆる Solaris) でしかテストしていません。おそらく、ほかのシステムでも働くでしょうが試したことはありません。

このモジュールを構築するには、引数 '--enable-shared' を configure にわたさなくてはなりません。なぜなら、PAM モジュールは共有ライブラリとしてロードされなくてはならないからです。共有ライブラリは、 PAM モジュールが正しい操作をするために、実行時においても利用可能でなくてはなりません。PAM モジュールの構成エラーを追跡するために 'debug' 引数を渡してもかまいません。

共有ライブラリを構築するのに対して、スタンドアローンの PAM モジュール、すなわち、Kerberos の共有ライブラリに依存しないモジュールを作ることも可能です。これを行うためには、アーカイブ (.a) ライブラリにポジション依存のコードが要求されます。以下のステップはスタンドアローンの PAM モジュールを構築するために使えます:

```
datan$ env CFLAGS=-fpic ./configure
datan$ make
datan$ cd lib/auth/pam
datan$ make
datan$ make install
```

configure に '--enable-shared' を渡すべきでないことに注意し、CFLAGS の設定に注意します。このモジュールは、たとえ Kerberos ライブラリがロードできなくても、機能するでしょう。

このモジュールを使うには次のようにすべきです :

- 'pam_krb4.so' が '/usr/athena/lib' で使えることを確認して下さい。ローカルディスク上にそれを置いておきたいと思うでしょうから、'/usr/athena' がローカルディスクで無い場合は、'/lib/security' がもっと良いかわりの場所となるでしょう。
- '/etc/pam.conf'に加える内容の参考例として、'krb4-1.0.2/lib/auth/pam/pam.conf.add' を見て下さい。

PAM モジュールにケルベロス・パスワードを変更するためのサポートは現在ありません。代わりに、kpasswd を使って下さい。

この PAM モジュールは、
ftp://ftp.dementia.org/pub/pamにある Derrick J Brashear <shadow@dementia.org> の Kerberos PAM モジュールに多大な影響を受けています。

# 4 How to set up a realm

> Who willed you? or whose will stands but mine?
> There's none protector of the realm but I.
> Break up the gates, I'll be your warrantize.
> Shall I be flouted thus by dunghill grooms?
> — King Henry VI, 6.1

## 4.1 How to set up the kerberos server

### 4.1.1 Choose a realm name

レルム (realm) は管理上の領域です。ケルベロス (kerberos) レルムは通常大文字で書かれた[1] インターネット (Internet) のドメイン名 (domain name) です。特別な理由が無い限り、レルムにインターネット・ドメイン名と同じ名前を付けて下さい。そうすると、あなたもあなたの回りの他の人々も楽になります。

### 4.1.2 Choose a kerberos server

ケルベロス・サーバ・プログラムを走らせるマシンを選定する必要があります。もし、ケルベロスのデータベースが在留するホストが開け渡されると、そのレルム全体が開け渡されてしまうことになります。それゆえ、このマシンはできる限り安全でなければなりません。できれば、ケルベロス以外のサービスを走らせるべきではありません。安全に配慮する管理者のみがコンソールにログインすることを許されるべきです。

このマシンには信頼性もなくてはなりません。もし、それがダウンすると、スレーブ・サーバを設置して (see Section 4.5 [Install a slave kerberos server], page 20) いない限り、全てののケルベロス化されたサービスが受けられなくなってしまいます。

ケルベロス・サーバを走らせるには、少しの CPU パワーと小さなディスクが必要なだけです。数百メガバイトの空きディスク空間を持つ古い PC で十分でしょう。ほとんどのディスク空間は様々な記録 (log) のために使えるでしょう。

### 4.1.3 Install the configuration files

二つの重要な構成ファイル (configration files) : '`/etc/krb.conf`' と '`/etc/krb.realms`' とがあります。

'`krb.conf`' ファイルではどのマシンが異なるレルムのサーバかを定義します。このファイルの書式は :

```
THIS.REALM
SUPP.LOCAL.REALM
THIS.REALM              kerberos.this.realm admin server
THIS.REALM              kerberos-1.this.realm
SUPP.LOCAL.REALM        kerberos.supp.local.realm admin server
ANOTHER.REALM           kerberos.another.realm
```

最初の行にはローカル・レルムの名前を定義します。その次からの数行はオプションで、副次的なローカル・レルムの名前です。このファイルの残りの行は、知っているすべてのレルムのケルベロス・サーバとデータベース・アドミニストレーション・サーバの名前を定義しています。4 行目のように書けば、このレルムのケルベロス・スレーブ・サーバを幾つでも定義できます。クライアントは '`krb.conf`' に列挙されているサーバの順番で接触を試みます。

[1] レルム名に小文字を使うと妙にうまくゆきません。これは、本当はすべきなのですが、まだ直されていません。

最初の記載文の 'admin server' 句は、これがマスタ・サーバ (パスワードの変更のような、データベースを更新しようとするときに接触するもの。) ということです。これはそれぞれのレルム毎に一意にすべきです。

元の MIT Kerberos4 では (他のほとんどにおいてと同様に)、サーバの指定はホスト名の形をとるだけです。(ファイアウォールの内側のような) 限られた場所でケルベロス・サーバを設置するために、既定ポート (750) 以外のポートと UDP 以外のプロトコルのサポートが加えられました。

現在、記載の正式な構文は '[*proto*/]*host*[:*port*]'です。*proto* は 'UDP' か 'TCP' か 'HTTP', かで、*port* は問いかけるポートです。*proto* の既定値は 'UDP' で *port* には、'kerberos-iv' が '/etc/services' の中に定義されていればそれを、でなれば、750 を既定値とします。もし *proto* が 'HTTP' であれば、既定ポートは 80 になります。'HTTP' の記載は URL 書式でも指定できます。

1. もし、レルムの SRV-レコード (*RFC 2052*) を持っていればそれが使えます。このレコードは '`kerberos-iv.`*protocol*`.`*REALM*' の形式で、*proto* は 'udp' か 'TCP' かあるいは 'HTTP' となるべきです。(注記：現在の実装ではどちらのサーバに話かけるかを決めるための優先度や重みについての制限は行ないません。)
2. もし、SRV-レコードが無ければ、同ドメインのための TXT-レコードを見つけようと試みます。そのレコードの内容は '`krb.conf`' の中のホスト指定の書式と同じにすべきです。(注意:もしネーム・サーバが SRV レコードをサポートしなければ、これが一時的な解決策となります。クライアントは SRV レコードでも十分動くはずで、もし、ネーム・サーバがそれらをサポートするのであれば、そちらの方が好まれます。)
3. もし、適切なケルベロス・サーバが見つからなければ、UDP でサービス '`kerberos-iv`' に問いかけようとし、750 を落ち着き先のポートとして、(マスタ・サーバと過程される) '`kerberos.`*REALM*' へ、それから、'`kerberos-1.`*REALM*'、そして、'`kerberos-2.`*REALM*'の順に試します。

SRV レコードは BIND 4.9.5T2A 以降でサポートされています。ゾーン・ファイルの記述例は次のようになります：

```
kerberos-iv.udp.foo.se.  1M IN SRV  1 0 750 kerberos-1.foo.se.
kerberos-iv.udp.foo.se.  1M IN SRV  0 0 750 kerberos.foo.se.
```

ケルベロス・マスタ・サーバを指す CNAME '`kerberos.`*REALM*' を加えることを、強くお奨めします。

'`krb.realms`' ファイルは特定のホストがどのレルムに属するかを見つけ出すのに使われます。このファイルの例は以下の通りです：

```
this.realm              THIS.REALM
.this.realm             THIS.REALM
foo.com                 SOME.OTHER.REALM
www.foo.com             A.STRANGE.REALM
.foo.com                FOO.REALM
```

点 ('.') から始まる記載はドメイン名として、点で始まらない記載はホスト名として解釈されます。最初に合致した記載が使われます。'`this.realm`' の記載は '`this.realm`' という名前のホストがある場合にのみ必要です。

もし、合致するレルムが '`krb.realms`' の中に見つからなければ、正しいレルムのために DNS で捜します。たとえば、もしホスト '`a.b.c`' を捜しているとすると、最初は '`krb4-realm.a.b.c`' が試され、そして '`krb4-realm.b.c`' がという具合になります。その記載は次のように、レルム名を含む TXT レコードにするべきです。

```
krb4-realm.pdc.kth.se.  7200    TXT     "NADA.KTH.SE"
```

もし、これがドメイン名に無ければ、最初の部分が大文字のものが試されます。

もとの簡素なケルベロスでは、サーバとレルムを得るための垢抜けた方法を持たないので、一般的には '`krb.conf`' と '`krb.realms`' をいつも最新にしておくことが良い考えです。

これらの一般に使うファイルに加えて、'`/etc/krb.extra`' は、通常は使わないものを内包します。それは、'*variable* = *value*' の組から成り、空行と (#) で始まる行は無視されます。

現在、定義されている変数を以下に示します。ブーリアン / フラッグの変数は 'true' または 'yes' もしくはゼロ以外の正数です。

'kdc_timeout'
: KDC からの答を待つ秒単位の時間 (既定値は 4 秒)。

'kdc_timesync'
: このフラグは最初の切符を得る時に KDC への時間微分の保存を可能にします。この微分は後で現在の時間を計算するのに使います。これはマシンのクロックが動いてない時の助けとなり得ます。

'firewall_address'
: ファイアウォールの内側からファイアウォールの外のホストに繋ぐときに見る IP アドレスです。もし、これが指定されていればプログラムは、'reverse_lsb_test' のための値を計算しようとします。

'krb4_proxy'
: HTTP からチケットを得る時に使用するプロクシを指定します。既定では直接 KDC に話かけます。

'krb_default_tkt_root'
: チケット・ファイルの既定の前置添字です。既定では '/tmp/tkt' です。通常はそれの後置添字として uid と tty が付け加えられます。

'krb_default_keyfile'
: サーバの鍵が保存されるファイルの場所を示します。既定では '/etc/srvtab' となります。

'nat_in_use'
: クライアントが Network Address Translator (NAT) の後ろにあることを指定するフラッグです。

'reverse_lsb_test'
: 通信しているホストの順番を計算するのに使われる、krb_mk_safe, krb_rd_safe, krb_mk_priv, および krb_rd_priv によるテスト順番をひっくり返します。このテストはファイアウォールを使う時に問題を起こし得ます。

### 4.1.4 Updating /etc/services

'services.append' の内容ををを '/etc/services' または NIS-map に追加あるいは混合するべきです。出荷時設定の使わないケルベロスのポート定義は、混乱の可能性を避けるために削除して下さい。

ほとんどのプログラムは '/etc/services' にポート番号が見つからなければ、既定のポートとなるでしょうが、いずれにしろそこにあった方が便利です。

### 4.1.5 Install the kerberos server

既にケルベロス・サーバを走らせるマシンとレルム名を選んでいるはずです。そのマシンはケルベロス・サーバをインストールする前に可能な限り安全にしてあるはずです (see Section 4.1.2 [Choose a kerberos server], page 10)。ここでの例では、ケルベロス・サーバを 'FOO.SE' というレルムで 'hemlig.foo.se' と呼ばれるマシンにインストールすることにします。

### 4.1.6 Setup the server

ケルベロス・サーバのコンソールに root でログインします。'/usr/athena/bin' と '/usr/athena/sbin' をコマンド・パスに加えます。データベースを保存するためのディレクトリ '/var/kerberos' を作り (*mkdir /var/kerberos*)、*kdb_init*を走らせます。:

```
hemlig# mkdir /var/kerberos
hemlig# kdb_init
Realm name [default  FOO.SE ]:
You will be prompted for the database Master Password.
It is important that you NOT FORGET this password.

Enter Kerberos master password:
Verifying password
Enter Kerberos master password:
```

もし、構成ファイルを正しく設定していたならば、*kdb_init* は正しいレルムを既定値として選ぶはずです。でなければ、(うまい) 推定をします。マスタ・パスワードを入力します。

このパスワードはディスク上のケルベロス・データベースを暗号化するためと新しいランダム・キーを生成するためにだけ使われます。それを覚えておく必要はありません。*kstash* を走らせる時にもう一度打ち込むだけです。長くてでたらめなものを選びましょう。そして、同じパスワードを使って *kstash* を走らせます。:

```
hemlig# kstash

Enter Kerberos master password:

Current Kerberos master key version is 1.

Master key entered.  BEWARE!
Wrote master key to /.k
```

同じマスタ・パスワードを入力すると、それはファイル '/.k' に保存され、そして、ケルベロス・サーバは必要になるとそれを読み出します。もしかすると、ディスクが潰れたり、スレーブ・サーバを設定したい思うでしょうから、マスタ・パスワードを書いたものを封筒に入れて封をして安全なところにおきましょう。

kdb_init データベースを最小の記載で初期化します:

'krbtgt.*REALM*'
: ケルベロス・サーバへの認証をするために使う鍵。

'changepw.kerberos'
: 管理サーバへの認証をするために使う鍵。例えば、ユーザの追加、パスワードんの変更など。

'default'
: この記載は新しい項目が加えられた時にコピーされます。新しい記載に持たせたい値、特に有効期限がここに入ります。

'K.M'
: これはマスタ鍵で、暗号化しないで '/.k' の中に保存されているマスタ鍵がこのデータベースに対し正しいかを証明するためだけに使われます。

kstash は、マスタ・パスワードを読んで、ファイル '/.k'に書き込むだけです。これによって、マスタ・パスワードを入力することなく、ケルベロス・サーバを起動することを可能にします。このファイル ('/.k') は、それが存在する "安全な" マシン上の root ユーザにのみ読み出しが可能です。

### 4.1.7 Add a few important principals

これで、最小のプリンシパルを含んだケルベロスのデータベースができました。次のステップではプリンシパルを更にもう少し加え、そして、きちんと動くことをテストして、それから、ケルベロス・サーバのコンソールを使わないでレルムの管理をすることができます。*kdb_edit*を使ってケルベロス・サーバのデータベースを編集します。

kdb_edit はデータベースの編集に際し、立ち上げ (bootstrapping) と後戻り (fall-back) 機構のために意図されたものです。普通の目的には、kadmin プログラムを使います (see Section 4.1.12 [Add users to the database], page 16)。

次の例は 'nisse.admin' をケルベロス・データベースの中に加えるところを示します。このプリンシパルはケルベロス・データベースを管理する時に 'nisse' によって使われます。あとで 'nisse' のための正常なプリンシパルがつくられます。'nisse' と 'password' とを自分のユーザ名とパスワードとに置き換えて下さい。

```
hemlig# kdb_edit -n
Opening database...
Current Kerberos master key version is 1.

Master key entered.  BEWARE!
Previous or default values are in [brackets] ,
enter return to leave the same, or new value.

Principal name: <nisse>
Instance: <admin>

<Not found>, Create [y] ? <>

Principal: nisse, Instance: admin, kdc_key_ver: 1
New Password: <password>
Verifying password
New Password: <password>

Principal's new key version = 1
Expiration date (enter yyyy-mm-dd) [ 2000-01-01 ] ? <>
Max ticket lifetime (*5 minutes) [ 255 ] ? <>
Attributes [ 0 ] ? <>
Edit O.K.
Principal name: <>
```

kdb_edit は "Principal name" のプロンプトに *return* を打つまで繰り返します。これで、nisse を管理者として登録しました。

### 4.1.8 Start the server

```
hemlig# /usr/athena/libexec/kerberos &
Kerberos server starting
Sleep forever on error
Log file is /var/log/kerberos.log
Current Kerberos master key version is 1.

Master key entered.  BEWARE!

Current Kerberos master key version is 1
Local realm: FOO.SE
```

### 4.1.9 Try to get tickets

遂に、追加したプリンシパルとサーバが正しく動くことを証明することができます。

```
hemlig# kinit
eBones International (hemlig.foo.se)
Kerberos Initialization
Kerberos name: <nisse.admin>
Password: <password>
```

もし、kinit からなにもエラーを受け取らなければ、全て動いています。(でなければ、Section 6.3 [Common error messages], page 27 を御覧下さい。) klist を使えば、kinit で得られた切符を検証できます：

```
hemlig# klist
Ticket file:    /tmp/tkt0
Principal:      nisse.admin@FOO.SE

Issued           Expires          Principal
May 24 21:06:03  May 25 07:06:03  krbtgt.FOO.SE@FOO.SE
```

### 4.1.10 Create initial ACL for the admin server

管理サーバ kadmind は、一連のファイルを使って誰が何の操作をする権利を持つかを決定します。それらのファイルは：

‘admin_acl.add’, ‘admin_acl.get’, ‘admin_acl.del’, および ‘admin_acl.mod’です。‘nisse.admin@FOO.SE’をその内容に加えるには次のようにします:

```
hemlig# echo "nisse.admin@FOO.SE" >> /var/kerberos/admin_acl.add
hemlig# echo "nisse.admin@FOO.SE" >> /var/kerberos/admin_acl.get
hemlig# echo "nisse.admin@FOO.SE" >> /var/kerberos/admin_acl.mod
hemlig# echo "nisse.admin@FOO.SE" >> /var/kerberos/admin_acl.del
```

後で、管理者権限をさらに多くのユーザを加えたければ、それぞれの管理用プリンシパルを作ってそれらを管理サーバの ACL に加えることを忘れないで下さい。

### 4.1.11 Start the admin server

```
hemlig# /usr/athena/libexec/kadmind &
KADM Server KADM0.0A initializing
Please do not use 'kill -9' to kill this job, use a
regular kill instead

Current Kerberos master key version is 1.

Master key entered.  BEWARE!
```

### 4.1.12 Add users to the database

kadmin クライアントを使ってユーザをデータベースに加えます：

```
hemlig# kadmin -p nisse.admin -m
Welcome to the Kerberos Administration Program, version 2
Type "help" if you need it.
admin:  <add nisse>
Admin password: <nisse.admin's password>
Maximum ticket lifetime?  (255)  [Forever]
Attributes?  [0x00]
Expiration date (enter yyyy-mm-dd) ?  [Sat Jan  1 05:59:00 2000]
Password for nisse:
Verifying password Password for nisse:
nisse added to database.
```

他のユーザも加えたければ同様に行ないます。次に、ユーザがデータベースの中にあることを検証し、そのユーザのデータベースの記載を確認します。

```
admin:  <get nisse>
Info in Database for nisse.:
Max Life: 255 (Forever)   Exp Date: Sat Jan  1 05:59:59 2000

Attribs: 00  key: 0 0
admin:  <^D>
Cleaning up and exiting.
```

### 4.1.13 Automate the startup of the servers

スタートアップ・スクリプト ('/etc/rc' あるいはそれと同等なもの) にケルベロス・サーバと管理サーバを始動するために使われる行を加えます。

## 4.2 Install the client programs

ケルベロス・クライアントのマシンをつくるのはほんの数段階です。最初に構成ファイルのケルベロス・サーバの変更をする必要があるでしょう。(see Section 4.1.3 [Install the configuration files], page 10 および see Section 4.1.4 [Install the /etc/services], page 12。) さらに、'/usr/athena/bin' にあるプログラムを利用可能にする必要があります。これは、ユーザのパスに、それを加えるか、シンボリック・リンクにするか、あるいは、プログラムをコピーするかによってできます。

さらに、なんらかの方法で、クライアント上の時刻がケルベロス・サーバの時刻と同期していることを証明するべきでしょう。同調するサーバとクライアントとの両者の間で、許される時間の差は最大 5 分です。

時刻を同期させるのに良い方法の一つとして NTP (Network Time Protocol) があります。http://www.eecis.udel.ed
を御覧下さい。

もし、root-アクセス権を持たないマシンでクライアント・プログラムを走らせる必要があるとしても、お望み通り、実行モジュールを使用するだけで構成を行なう必要はありません。特殊な使われ方は上記にて言及してます。(see Section 4.1.3 [Install the configuration files], page 10). もし、この場合に該当せず、'krb.conf' や 'krb.realms' を持つ必要があれば、それらを選んだディレクトリにコピーして、環境変数 *KRBCONFDIR* をそのディレクトリを指すように設定します。

クライアントの機能をテストするには、kinit プログラムを走らせます：

```
foo$ kinit
eBones International (foo.foo.se)
Kerberos Initialization
Kerberos name: <nisse>
Password: <password>

foo$ klist
Ticket file:    /tmp/tkt4711
Principal:      nisse@FOO.SE

Issued           Expires          Principal
May 24 21:06:03  May 25 07:06:03  krbtgt.FOO.SE@FOO.SE
```

## 4.3 Install the kerberised services

これらには、rsh, rlogin, telnet, ftp, rxtelnetなどが含まれます。

まず、前節で述べられた段階に従ってクライアントを作り、その操作の検証を行ないます。次に、新しいデーモンを使うために、'inetd.conf' を変更します。ファイル'etc/inetd.conf.changes' を見て、'inetd.conf' に施すお奨めを御覧下さい。

この時点では、どのマシンで何のサービスを走らせるかを決めているはずです。

### 4.3.1 rsh, rlogin, and rcp

ここでは、ケルベロス化されたバージョンと "旧式 (old style)" バージョンが存在します。異なるバージョンは異なるポート番号を使いますので、何にも無しか、一方か、あるいは、両方を選べます。もし、強いて "旧式" の r* サービスを使いたくなければ、プログラムにそのポートへの接続を単に拒むかわりに、"Remote host requires Kerberos authentication" のテキストを出力させることができます。これは、'-v' オプションで有効となります。ケルベロス化されたサービスには暗号化と非暗号化のバージョンが存在します。暗号化サービスには "e" が名前の前に付き、プログラムは '-x' を暗号化を示唆するオプションとして受け付けます。

われわれの推奨は、ケルベロス化されたサービスのみを使い、古いポートには説明のためのメッセージを与えるものです。

### 4.3.2 telnet

telnet サービスはいつも同じポートを使って、どの認証方法が使われるべきかを交渉します。`telnetd` プログラムが "-a user" オプション を持てば、ケルベロス化されて認証された接続のみ許します。もし、それを含まれなければ、明示文 (clear text) パスワードの使用に後戻ります。明らかな理由で、このオプションを有効にすることを推奨します。もし、ワンタイム・パスワードを (see Chapter 5 [One-Time Passwords], page 23) を使いたければ、"-a otp" オプションで OTPs かケルベロス化による接続を許すでしょう。

### 4.3.3 ftp

ftp サービスは telnet と同様に指定された一つのポートで動きます。既定では、ケルベロス認証された接続を許します。以下のオプションで、付加的なレベルを指定できます。

`-a otp` ワンタイム・パスワード (see Chapter 5 [One-Time Passwords], page 23) を許します。

`-a ftp` 匿名ログイン ("ftp" または "anonymous" ユーザにて) を許します。

`-a safe` 後方互換のため `-a ftp` と同じです。

`-a plain` クリアテキストのパスワードを許します。

`-a none` `-a ftp -a plain` と同じです。

`-a user` これらも後方互換の理由で何もしません。

匿名 (anonymous) ftp を走らせるとき、それをどのように設定するかを説明する `ftpd` についてのオンライン・マニュアルを読むべきです。

### 4.3.4 pop

ポストオフィス・プロトコル (POP) はメールをメール・ハブから取り込むのに使われます。

`popper` プログラムは、標準 POP3 プロトコルとケルベロス化された KPOP を実装してます。ケルベロス版のプロトコルで走らせるには '`-k`' を使って下さい。このサービスはメール・ハブだけで走らせるべきです。

### 4.3.5 kx

`kx` はケルベロス認証されてかつ暗号化された接続の上で X を走らせることを許します。このプログラムは `rxtelnet`、 `tenletxr`、 および、 `rxterm` により使われます。

もし、X ライブラリが UNIX-ソケット を使うことを許されないような妙な類のオペレーティング・システムをお持ちの場合は、'`-t`' オプションを `kxd` に指定する必要があります。それ以外は、デーモンを '`inetd.conf`' に加えるだけで十分なはずです。

### 4.3.6 kauth

このサービスは切符 (ticket) をリモート・ホスト上に作ることを許します。これを可能にするには、'`inetd.conf`' に関係する行を挿入するだけです。

## 4.4 srvtabs

全てのユーザはケルベロス・サーバで登録されたパスワードを持っている必要があり、同様に全てのサービスはケルベロス・サーバとの共有鍵を持つ必要があります。そのサービス鍵は、通常は '/etc/srvtab' と呼ばれるファイルに保存されます。このファイルは鍵が暴かれるの避けるために root 以外の誰にも読まれないようにすべきです。ケルベロス・データベースの中のこのプリンシパルの名前は通常はサービス名とホスト名です。このようなプリンシパルの例は、'pop.*hostname*' と 'rcmd.*hostname*' です。(rcmd は "remote command" に由来します。) もっともよく使われる srvtab の幾つかの型と、それらを使うプログラムをここに列挙します。

rcmd.*hostname*
: rsh, rcp, rlogin, telnet, kauth, su, kx

rcmd.kerberos
: kprop

pop.*hostname*
: popper, movemail, push

sample.*hostname*
: sample_server, simple_server

changepw.kerberos
: kadmin, kpasswd

krbtgt.*realm*
: kerberos (not stored in any srvtab)

ftp.*hostname*
: ftp (also tries with rcmd.*hostname*)

zephyr.zephyr
: Zephyr

afs or afs.*cellname*
: Andrew File System

これらの鍵を作るためには、ksrvutil プログラムを使うでしょう。これを行なうには以下のようにします：

```
bar# ksrvutil -p nisse.admin get
Name [rcmd]: <>
Instance [bar]: <>
Realm [FOO.SE]: <>
Is this correct? (y,n) [y] <>
Add more keys? (y,n) [n] <>
Password for nisse.admin@FOO.SE: <nisse.admin's password>
Written rcmd.bar
rcmd.bar@FOO.SE
Old keyfile in /etc/srvtab.old.
```

### 4.4.1 Complete test of the kerberised services

あるマシン ('foo') 上で切符を入手して、ケルベロス化サービスで次のマシン ('bar') にログインするために使います。もしうまく行けば、そのテストは以下のように見えます:

```
foo$ kinit nisse
eBones International (foo.foo.se)
Kerberos Initialization for "nisse"
Password: <nisse's password>
foo$ klist
Ticket file:    /tmp/tkt4711
Principal:      nisse@FOO.SE

Issued           Expires          Principal
May 30 13:48:03  May 30 23:48:03  krbtgt.FOO.SE@FOO.SE
foo$ telnet bar
Trying 17.17.17.17...
Connected to bar.foo.se
Escape character is '^]'.
[ Trying mutual KERBEROS4 ... ]
[ Kerberos V4 accepts you ]
[ Kerberos V4 challenge successful ]
bar$
```

全てがうまく動いていることを確かめるのに、rsh, rcp, rlogin, rlogin -x やその他のコマンドで試してみることもできます。

## 4.5 Install a slave kerberos server

マスタ・サーバが落ちた場合のことを考えて、少なくとも 1 つ以上のバックアップ (スレーブ)・サーバを持つことは望ましいことです。このようなスレーブ・サーバを好きな数だけ持つことは可能ですが、通常は余計な重複をさせるために三つ以上も持つことはありません。

最初に良いサーバ・マシンを選定します。see Section 4.1.2 [Choose a kerberos server], page 10。マスターで 'rcmd.kerberos' (注記、文字通り "kerberos" とすべきです) プリンシパルを ('ksrvutil get' を使って) 加えます。マスター上で走る kprop プログラムは、スレーブ・サーバ上で走っている kpropd デーモンに認証を求めるときにこれを使います。スレーブ上の kpropd はマスターからの認証接続のために 'rcmd.hostname' 鍵を使うでしょう。それゆえ、スレーブは srvtab にこれのための鍵を持つ必要があり、そして、もちろん、サーバーとして活動するために十分の構成ファイルを持つ必要もあります。これをどのように行なうかは Section 4.3 [Install the kerberised services], page 17 を参照して下さい。

総括すると、マスターは 'rcmd.kerberos' のための鍵を持ち、そして、スレーブは 'rcmd.hostname' の鍵を持つはずです。

スレーブはマスターに使ったものと同じ親鍵を必要とするでしょう。

マスタ・サーバでは、ケルベロス・スレーブ・サーバのホスト名を含むファイル、例えば '/var/kerberos/slaves'、を作ります。

kpropd をスレーブ・サーバ上で、 'kpropd -i' で、開始します。

マスタ・サーバ上では、データベースのダンプを作って、それを伝播します。

```
foo# kdb_util slave_dump /var/kerberos/slave_dump
foo# kprop
```

これで、データベースのコピーをスレーブ・サーバ上に持つはずです。このことは、スレーブサーバ上で 'kdb_util dump *file*' を発行して、元のマスタ・サーバ上のファイルと比較することによって証明することができます。記載が同じ順番にはないことに注意して下さい。

一例として、この手続きを一時間毎にクローンにより走らされる正規のスクリプトにより自動化するべきです。

マスターとスレーブは同じデータベースの複製を使うので、両者は同じ親鍵をを使う必要があります。kstash で、親鍵をスレーブに加えます。(see Section 4.1.6 [Set up the server], page 12)

ケルベロス・サーバをスレーブで開始するには、まず最初にマスタ鍵をマスタ・サーバよりコピーしなければなりません。このためには、マスタ・パスワードを覚えておいて、'kstash' コマンドを発行するか、あるいは、単に鍵ファイルをマスタ・サーバからコピーすることにより可能です。もし、ファイルをコピーする場合は、安全な媒体で、ネットワークを通さないでコピーすることを忘れないで下さい。良い方法はフロッピーか紙に入れることです。始末し易いので紙の方が良いでしょう。

スレーブ・サーバ上では、ケルベロス・サーバは '-s' オプションで開始されるべきです。これは、たとえば最後のマスタ・サーバからのアップデートからの時間など、正常性の確認を可能にします。

データベースへの変更は全てマスタ上の kadmind により行なわれ、スレーブに伝えられますので、スレーブ上では kadmind を走らせる必要は ありません。

最後にスレーブ・サーバに '/etc/krb.conf' を加えます。クライアントはそのファイルに指定された順番でサーバに問い合わせることになります。

Section 4.1.3 [Install the configuration files], page 10 を御覧になって、CNAME をスレーブサーバに追加することを検討して下さい。

## 4.6 Cross-realm functionality

自分のレルム 'MY.REALM' にいると仮定して、どのようにして他のレルム 'OTHER.REALM' にあるサーバに認証を求めますか？ 'MY.REALM' での正当な切符を持つことにより、そのレルムにてケルベロス化されたサービスと通信することを許します。しかしながら、他のレルムのコンピュータはあなたのレルムのケルベロス・サーバと秘密鍵を共有しません。

二つのレルムの間でお互いに信用しあえる共有鍵を加えることが可能です。telnet のようなクライアント・プログラムが相手のコンピュータが異なるレルムにあることを見つけると、ローカル・ケルベロス・サーバから、他のレルムへの発券許可証 (TGT) を得ようと試みるでしょう。そして、発券許可証さえあれば、他のレルムのケルベロス・サーバからサービス券を取得できるでしょう。

この機能を与えるためには、それぞれのレルムのプリンシパルを相互に加えなければなりません。プリンシパルは、'MY.REALM' では 'krbtgt.OTHER.REALM' で、そして、'OTHER.REALM' では 'krbtgt.MY.REALM' のはずです。その異なる 2 つのプリンシパルは同じ ( そして同じ鍵バージョン番号の ) 鍵を持つべきです。この鍵を安全に転送することを忘れないで下さい。これが要求されることのすべてです。

```
blubb$ klist
Ticket file:    /tmp/tkt3008
Principal:      joda@NADA.KTH.SE

  Issued           Expires          Principal
Jun  7 02:26:23  Jun  7 12:26:23  krbtgt.NADA.KTH.SE@NADA.KTH.SE
blubb$ telnet agat.e.kth.se
Trying 130.237.48.12...
Connected to agat.e.kth.se.
Escape character is '^]'.
[ Trying mutual KERBEROS4 ... ]
[ Kerberos V4 accepts you ]
[ Kerberos V4 challenge successful ]
Last login: Sun Jun  2 20:51:50 from emma.pdc.kth.se

agat$ exit
Connection closed by foreign host.
blubb$ klist
Ticket file:    /tmp/tkt3008
Principal:      joda@NADA.KTH.SE

  Issued           Expires          Principal
Jun  7 02:26:23  Jun  7 12:26:23  krbtgt.NADA.KTH.SE@NADA.KTH.SE
Jun  7 02:26:50  Jun  7 12:26:50  krbtgt.E.KTH.SE@NADA.KTH.SE
Jun  7 02:26:51  Jun  7 12:26:51  rcmd.agat@E.KTH.SE
```

# 5 One-Time Passwords

このパッケージには ワンタイム・パスワード (OTP) を使うためのサポートもあります。特に、`login`, `ftpd` それと `popper` でのサポートがあります。

## 5.1 What are one time passwords?

ワンタイム・パスワード (OTP) は、その名前のとおり、一回だけ使えるパスワードです (使い捨てパスワードとも言います)。これは、誰かがネットワークを盗聴して盗んだパスワードを使おうとしても使えないことを意味します。

このパッケージの OTP は *RFC 1938* をサポートします。この標準はよく知られている S/Key と後方互換でもあります。これを生成するプログラムは、HP 48 のものからクレイのものまで全てのマシンにわたりたくさんあります。

## 5.2 When to use one time passwords?

なぜ、ケルベロスの代わりに OTP を使いたがるのでしょうか？ OTP の利点は コンピュータの操作が必要無いことです。パスワードのリストをプリントアウトして持って行くことができますし、計算機か携帯型コンピュータを使って生成することもできます。

裏を返せば、OTP は透過型攻撃に対してのみあなたを守ります。最初の接続でのみ認証が行なわれます。それ以後は、誰でもあなたのセッションを盗聴できますので、OTP での接続を通して (パスワードのような) 重要なデータは送ったり見たりするべきではありません。また、侵入者が、あなたの TCP セッションかつまたは紹介データを、途中で横取りするような攻撃については無防備です。言い替えると、OTP では、最初の認証のみを提供しますが、データの保全性も内密性もありません。

OTP はタプル (*seed*, *sequence number*, *pass-phrase*) からジェネレートされます。種 (seed) と通番 (sequence number) は挑戦状 (*challenge*) の一部として表示されますので、それに対応したパスワードを生成するか、予め生成しておいたリストより取り出します。

最後に、OTP は簡単でどこでも使えますが、ケルベロスがするように、全ての脅威に対して保護されるわけではありません。ケルベロスが使えない時に使ってください。

## 5.3 Configuring OTPs

### Initializing

OTP を初期化するには `otp` プログラムを使います。このプログラムはこのホスト上で現在のパスワード (ここでは 100 番目) とそれに関係する種 (‘`foobar`’) を手元のファイルに記載します。

```
datan:>otp 100 foobar
Pass-phrase: <pass-phrase>
Verifying password Pass-phrase: <pass-phrase>
```

### Generating

それらのリストを出力するには `otpprint` と呼ばれるプログラムがあります。

```
datan:>otpprint 100 foobar
Pass-phrase: <pass-phrase>
91: SLAM BUY SUP DUSK SKY BEST
92: DEEM SIGH ROB RASH JUG MAT
93: DUET FISK HERS AREA TOLL SUP
94: WOW RAIN LEAK SARA MARK WING
95: COG YELL MILK CART ABE BAWL
96: GROW SILK GIST OMEN CAM ANNE
97: JAG QUAD NUT BEAT BHOY MAGI
98: ADAM USED GENE NIP EYE SIS
99: MY SUNG HERO AT DASH RAKE
100: CORN KNIT BOTH TOGO SOUL BOG
```

## Using the OTPs

自分用に初期化された一連のワン-タイム・パスワードの一つを使おうとする時は、使用されるアルゴリズムと通番とそして種を挑戦状とともに受け取ります。これらを生成器に入力するか、リストから対応するパスワードを見つけて下さい。

```
login: assar
assar's [ otp-md5 99 foobar ] Password: <MY SUNG HERO AT DASH RAKE>
```

パスワードの通番は `otp` に与えた番号より 1 だけ少ない数から始まり、それを使う毎に 1 づつ減って行きます。誰もあなたのパスワードを盗んで使っていないことを確かめるためには、現在の通番が何番かをいつも見ているべきです。番号が 0 に達したら、新しいパスワードのセットを入手する必要があります。

一度パスワードの綴を初期化してしまったら、上記のような挑戦状を受け取ったところで、どのようなパスワード・プロンプトに対してでも使うことができます。

## Configuring servers

`ftpd`, `telnetd`, および `popper` は、ケルベロス認証でない接続の時に、ワン-タイム・パスワードを要求するように設定することができます。オンライン・マニュアルで、これらのプログラムの現在のオプションを確認して下さい。

# 6 Resolving frequent problems

## 6.1 Problems compiling Kerberos

多くのコンパイラは ANSI 準拠にするためにスイッチを要します。kth4 は ANSI C で書かれていますので、使われるコンパイラの名前と ANSI 準拠にするために要求されるスイッチを指定する必要があります。これは、*env* コマンドを使って configure を走らせると最も簡単にできます。具体的に HP-UX の下で附属のコンパイラを使って構築するには次のように行ないます。

```
datan$ env CC="cc -Ae" ./configure
```

一般的には *gcc* が使えます。次の組合せでも配布パッケージをうまくコンパイルできたことが証明されています。

'HP-UX' *cc -Ae*

'Digital UNIX'
*cc -std1*

'AIX' *xlc*

'Solaris 2.x'
*cc* (unbundled one)

'IRIX' *cc*

### Linux の問題

RedHat4.2 の下で libc 関数の gethostby*() は時々コアダンプします。もしこれらの問題にぶつかったら、ファイル '/etc/nsswitch.conf' のホストの記載が以下に示す行よりも複雑になっていないかを確認して下さい。

```
hosts: files dns
```

あるシステムは krb4 を正しく構築するのに必要な '/usr/include/ndbm.h' が無くなっています。'ndbm.h.Linux' をソース配布パッケージの隣に置いてあります。

ある Linux パッケージでは動かない 'libdb' の報告があります。もしそれが起こったら、configure を走らせるときに *--without-berkeley-db* を使って下さい。

### SunOS 5 (aka Solaris 2) problems

共有ライブラリを構築する際に GNU gcc/ld の組合せを使うときは、RUN_PATH 環境変数を/usr/athena/lib (対象 libdir) に設定する方が良いでしょう。そうしないと、LD_LIBRARY_PATH を 実行時に設定しない限り、PAM モジュールは動きません。

### HP-UX の問題

共有ライブラリの '/usr/lib/libndbm.sl' がすべてのシステムに存在するわけではありません。この問題にとって更に悪いことは、静的結合のためのアーカイブ版も無いことです。それゆえ、バイナリを構築するときは、"本当に移植性のある" GNU gdbm または Berkeley DB を最初にインストールしおいて、そのライブラリに対して結合を行なえることを確認してからにします。

### Cray の問題

Cray 上で *rlogind* は `forkpty()` が移植されるまでは動きません。それまでは、*telnetd* を使って下さい。

## IRIX problems

IRIX には 3 種類の異なる ABI:(Application Binary Interface) があり、それぞれ、古い 32 ビットのインターフェース ( O32、あるいは、ただの 32 として知られる) と、新しい 32 ビットのインターフェース (N32)、および、64 ビットのインターフェース (64) です。O32 と N32 は、どちらも 32 ビットですが、それぞれ異なる呼出規定となっており、並びの制約についても同様です。IRIX 6.4 からは、N32 フォーマットが既定となっています。ABI はコンパイル時に選択しますが、'`--with-mips-abi`' 構成オプションで指定できます。このオプションに与えることのできる引数は、'o32', 'n32', あるいは, '64' で、N32 が既定です。3 種類の異なる ABI:s のためのライブラリは、通常、それぞれ異なるディレクトリ ('`lib`', '`lib32`', あるいは '`lib64`') にインストールされます。

GCC は、異なる ABI では、既知の問題を抱えています。古い GCC では O32 だけを扱えますが、新しい GCC では N32 と 64 を扱えて O32 は扱えず、ある版の GCC においては、N32 の構造体並びが壊れています。

この異なる ABI による混乱のため、ある問題が引き起こされます。例えば、'`afskauthlib.so`' ライブラリは '`xdm`' や '`login`' と同じ ABI を使う必要があります。最も簡単な方法は、'`/usr/bin/X11/xdm`' で走る '`file`' がどの ABI を使うのかを確認することです。

もう一つの問題は、Transarc の AFS を走らせようとする場合で、64-bit ABI をサポートしないため、64 ビットのアプリケーションからのトークンを入手することができません。もし、どうしてもしなくてはならないのでしたら、この機能を提供するためのカーネル・モジュールが `ftp://ftp.pdc.kth.se/home/joda/irix-afs64.tar.gz` にあります。

### AIX の問題

*gcc* version 2.7.2.* には、最適化をし過ぎると '`appl/telnet/telnetd/sys_term.c`' を間違えてコンパイルするバグがあります (もしかしたら '`appl/bsd/forkpty.c`' も)。

*xlc* プリプロセッサの幾つかの版では、(文書に書かれていない)'`-qnolm`'オプションを認識しません。もし、このオプションがそのプリプロセッサに ('`/etc/ibmcxx.cfg`' 構成ファイルを経由したりして) 渡されれば、構成 (configure) は失敗するでしょう。

この解決には構成ファイルからそのオプションを全体的に、あるいは、プリプロセッサについてのみ取り除くことです：

```
$ cp /etc/ibmcxx.cfg /tmp
$ed /tmp/ibmcxx.cfg
8328
/nolm
        options   = -D_AIX,-D_AIX32,-D_AIX41,-D_AIX43,-D_IBMR2,-D_POWER,-bpT:0x1000000
s/,-qnolm//p
        options   = -D_AIX,-D_AIX32,-D_AIX41,-D_AIX43,-D_IBMR2,-D_POWER,-bpT:0x1000000
w
8321
q
$ env CC=xlc CPP="xlc -E -F/tmp/ibmcxx.cfg" configure
```

AFS 3.4 version 5.38 for AIX 4.3 にはバグがあり、幾つかの場合にカーネルをパニックに陥れます。*login* にはこれに対する対処もありますが、この問題は他のプログラムにも及ぶかも知れません。この問題は、version 5.55 で直ったようです。

### C2 の問題

パスワードを確認するプログラムは 'passwd', OTP およびケルベロス・パスワードで動きます。もし、C2 セキュリティ(もしくはその他のパスワード・データベース) を使うならば、通常パスワードはどこかわからない所に保持します。もし、ケルベロスを C2 セキュリティで使いたいならば、どのような変更が必要かを考えなければならないでしょう。Section 3.4.1 [Digital SIA], page 7 を見て、Digital の SIA と C2 セキュリティについての議論も見て下さい。

## 6.2 Problems with firewalls

ファイアウォールはファイアウォールのある側から来るある型のパケットが反対側に来るのを分別して締め出します。ファイアウォールはある種の問題をケルベロスのように解決すると仮定されます ( 基本的には認可されないネットワークの使用を阻みます )。違いは、ケルベロスが利用者を認証しようとするのに対し、ファイアウォールはネットワークを '安全' な内側と '安全でない' 外側とに分けることです。

ファイアウォールを使う人々は、UDP が部分的に '安全でない' プロトコルを使っているという理由から、UDP は安全でないと考えます。ケルベロスは既定では UDP を使ってパケットを送ったり受け取ったりするので、ケルベロスとファイアウォールとが一緒ではあまりうまく動きません。

ファイアウォールの後ろ側からケルベロスを試そうとしている兆候は、切符を全く入手できません (`kinit` が悪名高い '`Can't send request`' エラーメッセージで終了します)。

これらの問題を解決するには幾つかの方法があります :

- ファイアウォールの管理者を説き伏せて、入ってくるパケットのために UDP ポートの 750 か 88 を開けてもらうことです。大抵これはややこしくなります。
- ファイアウォールの管理者を説き伏せて、出てゆくパケットのために UDP ポートの 750 か 88 を開けてもらうことです。これはもっと簡単にできるでしょう。もしかするともう可能になっているかもしれません。
- ある標準でないポートを越える TCP 接続を使いましょう。これには、ケルベロス・サーバの管理者を説き伏せてこのポートでの接続を可能にしてもらいます。
- HTTP を使って切符を得ます。ウェブによるものがほとんど無数に人気があるので、多くのファイアウォールが HTTP ポートを開いているか、あるいは HTTP プロクシを持っています。

最後の 2 つの方法は攻撃的と考えられるでしょう ( '正しい' 型のデータをそれぞれのポートに送るのではないので )。おそらく最善を尽くしてファイアウォールの管理者と議論することになるでしょう。

KDC と通信するときに他のプロトコルをどのように使うかについての情報については、Section 4.1.3 [Install the configuration files], page 10 を見て下さい

ファイアウォールは '内側' のアドレスを隠すことがよくありますので、全てのパケットがファイアウォールから来ているように見えます。クライアントのホストのアドレスが切符の中にコード化されてますので、これがトラブルを起こします。もし、切符を使おうとすれば '`Incorrect network address`' のようなエラーメッセージを受けとります。この問題の起きる理由は、話かけようとしているサーバが KDC から送られたのと異なるアドレスを見るためです。もし、この種のトラブルが起きたときの解決への最も近道は、おそらく切符を得るのに他の機構をためすことでしょう。管理者を説き伏せることができれば、サーバに異なる 2 つのアドレスを '`/etc/krb.equiv`' ファイルに加えてもらうべきです。

## 6.3 Common error messages

もっと不明瞭なエラーメッセージに幾つか出会うかも知れません :

'`Time is out of bounds`'
: あなたのマシンの時間が、ケルベロス・サーバか、あるいは、ログインしようとしているマシンかのどちらかの時間と異なります。もし、このことが明確でなければ、全ての時間は UTC で比較されることを思い出して下さい。

UNIX システム上では普通は `telnet machine daytime` を行なうことにより局所時間を調べることができます。この時間 (再び、普通はキーワード) はタイムゾーン (time-zone) と夏時間 (daylight savings) の補正を行なっています。もし、あなたの時刻を同期させ続けることが難しければ、NTP( Section 4.2 [Install the client programs], page 17 での議論も参照下さい) のような時刻合わせシステムの利用を考えて下さい。

'`Ticket issue date too far in the future`'
ケルベロス・サーバの時間がアプリケーション・サーバの時間より 5 分以上進んでいます。

'`Can't decode authenticator`'
これは、ケルベロス・サーバのサービス鍵が特定マシン上のサービス鍵ファイルと合っていないことを意味します。あるいは:

- サーバが要請に合致するサービス鍵を見つけられない
- サービス鍵 (あるいはバージョン番号) がパケットを暗号化した鍵にマッチしない

'`Incorrect network address`'
切符の中のアドレスがリクエストを送ったアドレスと一致しません。これは、物理的あるいは論理的に二つ以上のネットワークアドレスを持つシステムで起こります。サーバの '`/etc/krb.equiv`' に同等と考えられるべきアドレスを載せることができます。

プログラマへの注意: あるサーバは '`*`' を '`krb_rd_req`' へのインスタンスとして渡すべきではありません。'`k_getsockinst`' によるインスタンスについては、どのインターフェースでリクエストが受け取られたかをはっきりさせるように試みるべきです。

もし、コンピュータのアドレスを変えると、持っている切符は失効させられます。最も簡単にこれを直す方法は、新しいアドレスで新しい切符を得ることです。

'`Message integrity error`'
次のようにパケットが壊れています:

- 長さがパケットのサイズに合っていないか、あるいは、
- チェックサムがパケットの中身と合っていません。

'`Can't send request`'
ケルベロス・サーバと接触する上で幾つかの問題があります。サーバがダウンしているか、あるいは、間違ったポートを使っています ('`/etc/services`' の '`kerberos-iv`' の記載と比べてみてください)。クライアントが話かけるべきケルベロス・サーバを勘違いしているかもしれません ('`/etc/krb.conf`' と '`/etc/krb.realms`'を確認して下さい)。

ケルベロス・サーバと接触できない別の理由は、ケルベロスのパケットを通すことを許さないファイアウォールの後ろにいるからでしょう。これを可能にする解決方法は上記のファイアウォールの節を見て下さい。

'`kerberos: socket: Unable to open socket...`'
ケルベロス・サーバが各インターフェースのためにソケットを開かなければなりません。もし、沢山の仮想インターフェースがあるマシンを持っているとすると、ファイル記述子を使い果たす危険性があります。もしそれが起きるとこのエラーメッセージを受け取ります。

'`ftp: User foo access denied`'
通常これは、ユーザのシェルが '`/etc/shells`' に記述されていないために起こります。'`/etc/shells`' を持たないバージョンのシステムであっても、*ftpd* がチェックを行なうことに注意して下さい。

'`Generic kerberos error`'
これは、汎用的に受け取るエラーメッセージです。

## 6.4 Is Kerberos year 2000 safe?

Yes.

もう少し長い回答をすると、別に何も壊れるものはないと考えています。プロトコルそのものはテキスト形式の時間刻印は使っていないし、元の MIT コードにあった 2 桁の年号の問題も直されています (このほとんどはログファイルの問題でした)。FTP クライアントには、(既に取得したファイルより新しいファイルだけを持ってくる) newer コマンドにバグがありました。

Y2K 問題というわけではないのですが、 もう一つあるといえば、古いプリンシパルの失効日付です。MIT のコードでは幾つかの新しいプリンシパルの失効日を 1999-12-31 にしてますので、そうした日付のものがないか、データベースを確認しておいたほうが良いでしょう。

ところが、Y2038 問題は何かが全く違います (しかしながら、おそらく著者達はそのころまでには引退してるでしょう。察するに、野苺がどこか素敵な暖かい場所で育っていることでしょう。)

# Appendix A Acknowledgments

この版は、元々 MIT アテネ・プロジェクト (Athena project) の人々が書いたコードを基に、開発されています。Kerberos 4 patch-level 9 は暗号化関数とその呼出しを取り除かれて、US から “Bones” 版として輸出されました。Eric Young は彼自身が持っている libdes でその呼出しを復元し、それが “eBones” 版となりました。

“rcmd” プログラム群は、元々カリフォルニア大学バークレー校 (UCB) で開発されましたが、その後 FreeBSD や NetBSD などのプロジェクトで改良されました。

バークレーでは `ftp`, `ftpd`, `telnet` および `telnetd` も書かれました。`telnet` と `telnetd` の認証と暗号化のコードは David Borman(当時、Cray Research, Inc.) により加えられました。暗号化のコードは、これが輸出される時には削除され、その後 Juha Eskelinen, `<esc@magic.fi>` により再び付加されました。

`popper` も最初はバークレーのプログラムでした。

`login` も源は同じですが、オランダの Eindhoven University of Technology の Wietse Venema が書いたコードを受け入れてます。

`movemail` は (少なくとも部分的には) Jonathan Kamens`<jik@security.ov.com>` により書かれ、© 1986, 1991, 1992, 1993, 1994 Free Software Foundation, Inc. が著作権を有します。

`xnlock` は、元々 1985 年 Dan Heller が sunview のために書きました。その後 X 版も 1990 年に彼が書きました。

‘`libroken`’ の幾つかの関数もバークレーから NetBSD/FreeBSD を経て来たものです。

AIX の AFS モジュールのために動的読み込みを取り扱うコードは © 1992 HELIOS Software GmbH 30159 Hannover, Germany. が著作権を有します。

`editline` は Simmule Turner と Rich Salz が書きました。

バグ修正とコーディングは以下の方々の貢献によります:

Derrick J Brashear
`<shadow@dementia.org>`

Anders Gertz
`<gertz@lysator.liu.se>`

Dejan Ilic `<svedja@lysator.liu.se>`

Kent Engstrm
`<kent@lysator.liu.se>`

Simon Josefsson
`<jas@pdc.kth.se>`

Robert Malmgren
`<rom@incolumitas.se>`

Fredrik Ljungberg
`<flag@astrogator.se>`

Joakim Fallsj
`jfa@pobox.se`

Lars Malinowsky
`<lama@pdc.kth.se>`

Fabien Coelho
`<coelho@cri.ensmp.fr>`

Chris Chiappa
<griffon+@cmu.edu>

Gregory S. Stark
<gsstark@mit.edu>

Love Hrnquist-strand
<lha@stacken.kth.se>

Daniel Staaf
<d96-dst@nada.kth.se>

Magnus Ahltorp
<map@stacken.kth.se>

Robert Burgess
<rb@stacken.kth.se>

Lars Arvestad
<arve@nada.kth.se>

Jrgen Wahlsten
<wahlsten@pathfinder.com>

Daniel Staaf
<d96-dst@nada.kth.se>

R Lindsay Todd
<toddr@rpi.edu>

ke Sandgren
<ake@cs.umu.se>

Thomas Nystrm
<thn@stacken.kth.se>

それから、この記載に洩れている方はどうかご容赦下さい。

Ian Marsh <ianm@sics.se> はこのテキストから英語の悪い言い回しをとり除きました。

Ilja Hallberg <iha@incolumitas.se> が文書を清書するのを手伝ってくれています。

この作業は、部分的には、SUNET と KTH の並列コンピュータセンター (Centre for Parallel Computers) の支援によりなされました。

Windows 95/NT への移植は KTH のコンピュータ会議室 (Computer Council) の支援で Jrgen Karlsson <d93-jka@nada.kth.se> によりなされました。

全てのバグは我々自身により取り除かれました。

# Index


## .

`.klogin` ... 6

## A

`admin_acl.add` ... 15
`admin_acl.del` ... 15
`admin_acl.get` ... 15
`admin_acl.mod` ... 15
AFS ... 26
`afslog` ... 6
authentication ... 2

## B

Berkeley DB ... 5
Bones ... 30

## C

C2 ... 27
confidentiality ... 3
cracklib ... 4

## D

`default/login` ... 6

## E

eBones ... 30
`environment` ... 6

## F

`fbtab` ... 6
firewall ... 4, 27
firewall_address ... 12
`ftp` ... 6, 17, 18
`ftpd` ... 7

## G

GCC ... 25, 26

## H

Hesiod ... 4
HP-UX ... 25
HTTP ... 27

## I

`inetd.conf` ... 17
integrity ... 3
IRIX ... 26

## K

`kadmin` ... 6, 16
`kadmind` ... 4, 16
`kauth` ... 6, 19
`kauthd` ... 7
`kdb_edit` ... 13
`kdb_init` ... 12
kdc_timeout ... 12
kdc_timesync ... 12
`kdestroy` ... 6
`kerberos` ... 10, 15
`kinit` ... 6, 15
`klist` ... 6, 15
`kpasswd` ... 6
`kprop` ... 20
`kpropd` ... 20
`krb.conf` ... 10, 17
`krb.extra` ... 11
`krb.realms` ... 10, 17
krb_default_keyfile ... 12
krb_default_tkt_root ... 12
krb4_proxy ... 12
KRBCONFDIR ... 17
`ksrvutil` ... 6, 19
`kstash` ... 13
`kx` ... 18
`kxd` ... 7, 18

## L

Linux ... 25
`login` ... 4, 6
`login.access` ... 6
`logindevperm` ... 6

## M

master server ... 11

## N

NAT ... 12
nat_in_use ... 12
Network Address Translator ... 12
NTP ... 17

## O

One time passwords ... 23
`otp` ... 23
OTP ... 23
`otpprint` ... 23

## P

pagsh ..... 6
popper ..... 4, 7, 18
push ..... 4

## R

rc ..... 17
rcp ..... 6, 18
readline ..... 4
realm ..... 10
replay cache ..... 3
reverse_lsb_test ..... 12
rlogin ..... 6, 17, 18
rlogind ..... 6, 7
rsh ..... 6, 17, 18
rshd ..... 7
rxtelnet ..... 6, 17
rxterm ..... 6

## S

S/Key ..... 23
services ..... 12
session key ..... 2
socks ..... 4
srvtab ..... 19
su ..... 6, 7
SunOS 5 ..... 25
supplementary local realms ..... 10

## T

telnet ..... 6, 17, 18
telnetd ..... 6, 7, 18
tenletxr ..... 6

## X

xnlock ..... 6

## Y

Year 2000 ..... 28

# Table of Contents


**1 Introduction ..... 1**

**2 What is Kerberos? ..... 2**

2.1 Basic mechanism ..... 2
2.2 Different attacks ..... 2
2.3 Defense strategies ..... 3
2.4 Further reading ..... 3

**3 Installing programs ..... 4**

3.1 Installing from source ..... 4
3.2 Installing a binary distribution ..... 5
3.3 Finishing the installation ..... 6
3.4 Authentication modules ..... 7
3.4.1 Digital SIA ..... 7
3.4.2 IRIX ..... 8
3.4.3 PAM ..... 9

**4 How to set up a realm ..... 10**

4.1 How to set up the kerberos server ..... 10
4.1.1 Choose a realm name ..... 10
4.1.2 Choose a kerberos server ..... 10
4.1.3 Install the configuration files ..... 10
4.1.4 Updating /etc/services ..... 12
4.1.5 Install the kerberos server ..... 12
4.1.6 Setup the server ..... 12
4.1.7 Add a few important principals ..... 13
4.1.8 Start the server ..... 15
4.1.9 Try to get tickets ..... 15
4.1.10 Create initial ACL for the admin server ..... 15
4.1.11 Start the admin server ..... 16
4.1.12 Add users to the database ..... 16
4.1.13 Automate the startup of the servers ..... 16
4.2 Install the client programs ..... 17
4.3 Install the kerberised services ..... 17
4.3.1 rsh, rlogin, and rcp ..... 17
4.3.2 telnet ..... 18
4.3.3 ftp ..... 18
4.3.4 pop ..... 18
4.3.5 kx ..... 18
4.3.6 kauth ..... 18
4.4 srvtabs ..... 19
4.4.1 Complete test of the kerberised services ..... 19
4.5 Install a slave kerberos server ..... 20
4.6 Cross-realm functionality ..... 21

**5 One-Time Passwords ..... 23**

5.1 What are one time passwords? ..... 23
5.2 When to use one time passwords? ..... 23
5.3 Configuring OTPs ..... 23

**6 Resolving frequent problems ..... 25**
6.1 Problems compiling Kerberos ..... 25
6.2 Problems with firewalls ..... 27
6.3 Common error messages ..... 27
6.4 Is Kerberos year 2000 safe? ..... 28

**Appendix A Acknowledgments ..... 30**

**Index ..... 32**