Linux on System z

How to Set up a Terminal Server Environment on z/VM June 2009

Linux Kernel 2.6 – Development stream

SC34-2596-00

Linux on System z

How to Set up a Terminal Server Environment on z/VM June 2009

Linux Kernel 2.6 – Development stream

SC34-2596-00

Note

Before using this information and the product it supports, read the information in “Notices” on page 53.

First Edition – (June 2009)

This edition applies to the Linux on System z Development stream, s390-tools version 1.8.1, and to all subsequent releases and modifications until otherwise indicated in new editions.

© Copyright International Business Machines Corporation 2009.
US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

© Copyright IBM Corp. 2009

About this publication

This document describes how to set up a Linux® instance as a terminal server for a virtual Linux server farm on z/VM®. The terminal server uses Inter-User Communications Vehicle (IUCV) communications to access terminals on other Linux guest operating systems in the environment. Through the terminal server, you can access terminals on Linux instances that are not connected to an Internet Protocol (IP) network.

In this book, System z® is taken to include IBM® System z10™, System z9®, and zSeries® mainframes in 64- and 31-bit mode.

You can find the latest version of this document on developerWorks® at:
`www.ibm.com/developerworks/linux/linux390/documentation_dev.html`

Who should read this document

This document is intended for Linux administrators and system programmers in charge of a virtual Linux server farm that runs under z/VM.

How this document is organized

Chapter 1, “Introduction,” on page 1 provides an overview of the elements of a terminal server environment.

Chapter 2, “Requirements,” on page 5 tells you what you need to set up a terminal server environment.

Chapter 3, “Security,” on page 7 explains the control points you can use to protect your terminal server environment.

Chapter 4, “Setting up a terminal server,” on page 11 gives step-by-step instructions for setup tasks on the terminal server.

Chapter 5, “Setting up the target systems,” on page 17 gives step-by-step instructions for setup tasks on target systems.

Chapter 6, “Working with the terminal server,” on page 25 shows how to establish terminal sessions through the terminal server.

Chapter 7, “Scenarios,” on page 29 illustrates how the different elements of a terminal server environment interact in a particular context.

Appendix A, “Command reference,” on page 37 provides a reference for the most important commands used to set up, start, and access terminal devices.

Appendix B, “ts-shell user authorization file syntax,” on page 49 explains the syntax of a configuration file that authorizes Linux users on the terminal server to connect to specific target systems.

Appendix C, “Creating files with lists of z/VM user IDs,” on page 51 describes a convenient method you might want to use to create lists of z/VM user IDs.

© Copyright IBM Corp. 2009

Where to get more information

For information about z/VM guest virtual machine definitions, see *z/VM CP Planning and Administration*, SC24-6083.

For information about z/VM IUCV, see *z/VM CP Planning and Administration*, SC24-6083 and *z/VM CP Programming Services*, SC24-6084.

For information about the z/VM IUCV HVC device driver, see the chapter about console devices in *Device Drivers, Features, and Commands*, SC33-8411. You can obtain the latest version of this book on developerWorks at

`www.ibm.com/developerworks/linux/linux390/development_documentation.html`

See also the man pages for iucvtty, iucvconn, hvc_iucv, chiucvallow, and ts-shell.

Chapter 1. Introduction

A *terminal server* is a Linux instance that provides access to terminal devices on other Linux instances, called *target systems* in this document. The terminal server and all target systems run as guest operating systems of the same z/VM instance. Terminal server and target systems are connected through the z/VM Inter-User Communication Vehicle (IUCV). From the terminal server, administrators can access terminal devices on target systems without requiring direct TCP/IP connections to the target systems.

You can use a terminal server to:

- Increase availability by providing emergency access to target systems if the primary network for these systems fails.
- Heighten security by separating user networks from administrator networks or by isolating sensitive Linux instances from IP networks.
- Simplify systems administration by providing a central access point to target systems.

The environment

Figure 1 shows an overview of a terminal server environment with a terminal server and multiple target systems.

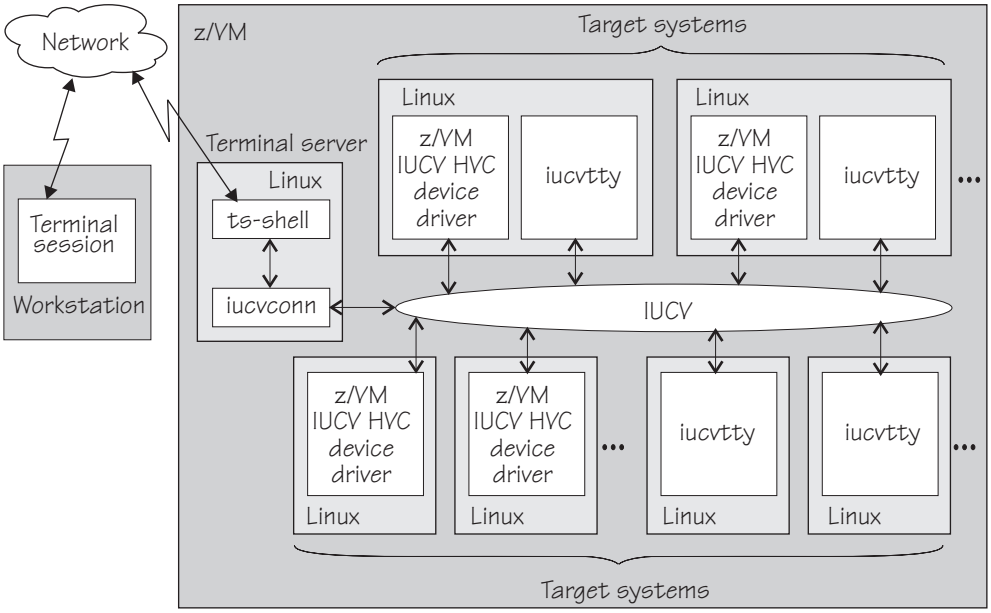

Figure 1. Terminal server environment

To access a terminal device on a target system, administrators first open a terminal session on a workstation and log in to a special terminal shell, the ts-shell, on the terminal server. The terminal shell uses the iucvconn program that can access terminal devices on target systems through z/VM IUCV connections.

Linux on System z supports two types of terminal devices that can be accessed through z/VM IUCV.

- Terminal devices provided by the iucvtty program.

© Copyright IBM Corp. 2009

For simplicity, these terminal devices are referred to as *iucvtty instances* in this document.

- Terminal devices provided by the z/VM IUCV hypervisor console (HVC) device driver.

  For simplicity, these terminal devices are referred to as *HVC terminal devices* in this document.

Both types of devices can be present on the same Linux instance and there can be multiple instances of each type. Each instance of a terminal device is accessed through a separate z/VM IUCV connection.

iucvtty instances

Several iucvtty instances can run to provide multiple terminal devices. The instances are distinguished by a terminal ID that is set when an iucvtty instance is started.

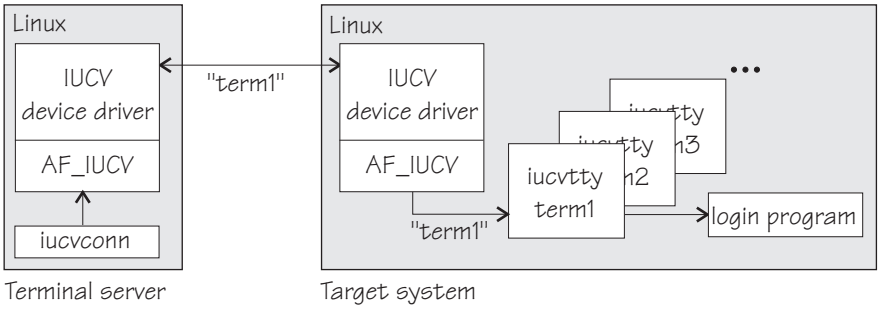

Figure 2. Login through iucvtty instances

Connection requests are created with the iucvconn program on the terminal server. A request includes the z/VM user ID of the target z/VM guest virtual machine and a terminal ID. After successfully connecting to the target system, a communication path is established to the iucvtty instance with the specified terminal ID.

An inittab entry or an Upstart job file associates the iucvtty instance with a login program.

HVC terminal devices

The z/VM IUCV HVC device driver is a kernel module and uses device nodes to enable HVC terminal devices to communicate with getty and login programs.

There can be up to 8 HVC terminal devices, hvc0 to hvc7. hvc0 can be activated to receive Linux kernel messages. The terminal IDs for HVC terminal devices match the device names with a leading “lnx”. For example, the terminal ID for `hvc0` is `lnxhvc0`.

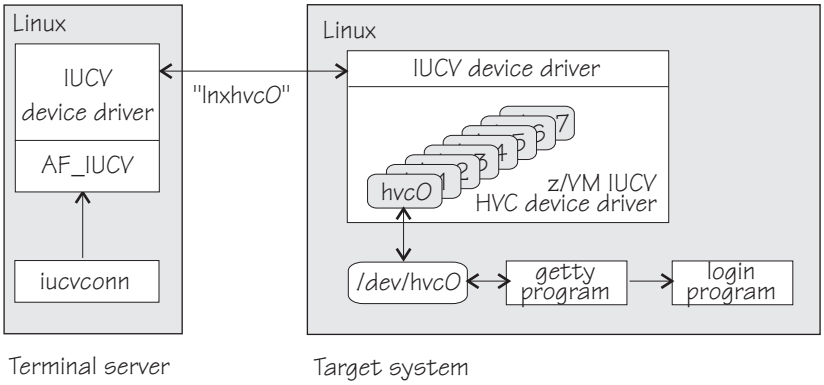

Figure 3. Login through HVC terminal devices

Connection requests are created with the iucvconn program on the terminal server. A request includes the z/VM user ID of the target z/VM guest virtual machine and the terminal ID of an HVC terminal device. The z/VM IUCV HVC device driver maps the terminal ID to the corresponding terminal device.

An inittab entry or an Upstart job file associates the HVC terminal device with a getty program and a login program.

Using iucvconn_on_login

As an alternative to giving terminal server users access to ts-shell on the terminal server you can configure Linux to start the iucvconn_on_login script when a user establishes an SSH session with the terminal server.

The iucvconn_on_login script immediately calls iucvconn and connects the user to a target system. The iucvconn_on_login user cannot perform any actions on the terminal server. Depending on how the terminal device on the target system is set up, a successful login to the terminal server is immediately followed by a login prompt for the target system.

For each target system to be reached through iucvconn_on_login, you must create a specific Linux user on the terminal server. The user name of this Linux user must match the z/VM user ID that identifies the target system. The terminal ID on the target system is specified as a parameter when establishing the SSH session to the terminal server.

See “Basic iucvconn_on_login scenario” on page 34 for an example.

Chapter 2. Requirements

This section lists the requirements for z/VM, terminal servers, and target systems in a terminal server environment.

Linux kernel and s390-tools

You need:

- Linux kernel 2.6.29 with the May 8 2009 Development stream code drop on developerWorks or Linux kernel 2.6.30 or later
- The s390-tools package version 1.8.1 or later

z/VM

To set up a terminal server environment you need z/VM 5.2 or later.

Terminal server

For the terminal server you need a Linux instance with:

- The IUCV device driver and AF_IUCV address family support (as separate modules or compiled into the kernel)
- The iucvconn program (from s390-tools)
- The ts-shell program (from s390-tools)
- Perl (Version 5 or later)

Optional additions:

- iucvconn_on_login (from s390-tools)
- scriptreplay (from the util-linux package)
- Command completion (Perl CPAN module Term::ReadLine::Perl or Term::ReadLine::Gnu)

If the s390-tools package is not included in your distribution, you can obtain it from `www.ibm.com/developerworks/linux/linux390/s390-tools.html`. The required programs are included as of version 1.8.1.

If the util-linux package is not included in your distribution, you can obtain it from `www.kernel.org/pub/linux/utils/util-linux/`.

If Perl is not included in your distribution, you can obtain it from `www.perl.org/`.

If the Comprehensive Perl Archive Network (CPAN) modules are not included in your Perl installation or provided as packages with your distribution, you can obtain them from `www.cpan.org/`.

Target system

For a target system you need a Linux instance with:

- The IUCV device driver (as a separate module or compiled into the kernel)

To support HVC terminal devices you also need:

- The z/VM IUCV HVC device driver (compiled into the kernel)

© Copyright IBM Corp. 2009

To support iucvtty instances you also need:

- The AF_IUCV address family support (as a separate module or compiled into the kernel)
- The iucvtty program (from s390-tools)

Optional addition:

- The chiucvallow program (from s390-tools)

If the s390-tools package is not included in your distribution, you can obtain it from `www.ibm.com/developerworks/linux/linux390/s390-tools.html`. The required programs are included as of version 1.8.1.

Chapter 3. Security

Access to Linux is typically controlled by an authentication program, for example, a login program. In a terminal server setup, you can also use additional security mechanisms:

- z/VM IUCV authorizations to control which IUCV connections are possible
- Restrictions on the terminal server to only allow connections to specific target systems
- Restrictions for the terminal devices on the target systems, to only allow access from specific z/VM guest virtual machines

How you set up security depends on the specific needs of your installation. This section describes the available control points. Chapter 7, “Scenarios,” on page 29 illustrates how you can combine the various possibilities into a working environment.

IUCV security on z/VM

You configure the IUCV connection between the terminal server and the target systems through IUCV statements in the z/VM user directory. An IUCV statement for one of the communication peers is sufficient to permit a particular connection. Depending on your needs you can use different strategies.

Permit any IUCV connection to a target system

The following statement in the user entry for a target system permits any other z/VM guest virtual machine to establish an IUCV connection to the target system. This permission also applies to z/VM guest virtual machines without IUCV statements in their own z/VM directory entry.

```
IUCV ALLOW
```

Omit this statement from the z/VM directory entry of your target systems unless you want to grant a general permission to all other z/VM guest virtual machines.

Permit the terminal server to connect to specific z/VM guest virtual machines

Through IUCV statements in the z/VM directory entry of the terminal server, you can explicitly specify the target systems to which the terminal server can establish an IUCV connection.

Example: These statements allow connections to the z/VM guest virtual machines with the z/VM user IDs LXGUEST1, LXGUEST2, LXGUEST7, and LXGUEST9.

```
IUCV LXGUEST1
IUCV LXGUEST2
IUCV LXGUEST7
IUCV LXGUEST9
```

With such explicit statements, you can avoid permitting IUCV connections that are not required or intended.

© Copyright IBM Corp. 2009

Permit the terminal server to connect to any z/VM guest virtual machine

If you regard the z/VM guest virtual machine of your terminal server as a trusted system, you can permit it to connect to all other z/VM guest virtual machines on the z/VM instance. You can grant this general permission with the following IUCV statement in the z/VM user directory entry for the z/VM guest virtual machine of the terminal server:

```
IUCV ANY
```

With this statement, a user on the terminal server can connect to all z/VM guest virtual machines on the same z/VM instance, including all target systems.

This general permission for the terminal server relieves you from updating the z/VM directory each time a new target system is added. The disadvantage is that general users on the terminal server can establish IUCV connections not only to all target systems, but also to all other z/VM guest virtual machines.

These concerns are addressed by a special shell that limits user actions on the terminal server, see “ts-shell” on page 9.

Security on the terminal server

This section summarizes some of the general security measures you might want to consider for your terminal server. It also introduces the ts-shell program and the iucvconn_on_login script both of which fence actions that are not directly related to connecting to target systems from terminal server users.

General security limiting access to the terminal server

Provide general security measures as you would for any sensitive system. For example consider the following measures:

Workload and users

It is good practice to use a dedicated system as the terminal server with no unnecessary users defined.

Physical access

Physical access to mainframe systems is tightly restricted in most installations. If you configure the network connection to the terminal server as a private network that can only be accessed from one or more workstations within a controlled physical area, you can also use physical access restrictions to protect your terminal server.

Hardening Linux

It is good practice to limit access to the Linux system to what is required. Do not install or load any modules that you do not need and switch off all daemons and processes that you do not need. To find out which processes are accessible at network sockets enter:

```
[root]# netstat -lptu
```

Firewall

Consider protecting your terminal server through a firewall.

ts-shell

You can set up the terminal server such that particular users always log in to ts-shell. The only functions available on ts-shell are commands that directly relate to establishing connections to target systems. Other functions on the terminal server are fenced from ts-shell users.

ts-shell can be configured to only permit connections to specific target systems, for ts-shell itself and for individual users.

iucvconn_on_login script

You can set up the terminal server such that particular users always log in to the iucvconn_on_login script. An iucvconn_on_login user logs in to Linux on the terminal server with a user ID that matches the z/VM user ID of a target system. After a successful login to the terminal server, the user is immediately prompted to log in to the target system. No action is possible on the terminal server.

Auditing

You can set up ts-shell to create transcripts of terminal sessions with target systems and store the transcripts on the terminal server.

The iucvconn_on_login script as included in s390-tools does not create session transcripts. If needed, you can modify the script to create session transcripts.

Logging

The ts-shell program and the iucvconn_on_login script both use the **iucvconn** command to connect to target systems. The **iucvconn** command logs all connection requests to syslog.

Security on the target system

This section describes extra security measures and considerations for the target systems.

Limiting access to terminal devices

You can limit the z/VM guest virtual machines from which connection requests are accepted for HVC terminal devices and individually for each iucvtty instance.

Enabling root logins

Whether direct root logins are permitted on terminal devices depends on the login program used. For example, the default login program for iucvtty instances and HVC terminal devices, `/bin/login` restricts root logins. Root logins are allowed only on devices for which a device node is listed in `/etc/securetty`.

To enable direct root logins on HVC terminal devices that use `/bin/login` you can add the respective device nodes to `/etc/securetty`.

Because iucvtty instances use pseudo terminal devices with dynamically assigned device nodes, enabling root logins on iucvtty instances that use `/bin/login` constitutes a potential security exposure. If you need root access through an iucvtty instance, log in as a general user and then change to root, for example, with the **su** command.

For security risks associated with other login programs, see the documentation for the login program.

Logging

All access requests to an iucvtty instance are logged to syslog.

All refused attempts to access an iucvtty instance or an HVC terminal device are logged to syslog.

Summary

Figure 4 summarizes the security barriers that a user must negotiate in a terminal server environment to gain access to a terminal device on a target system.

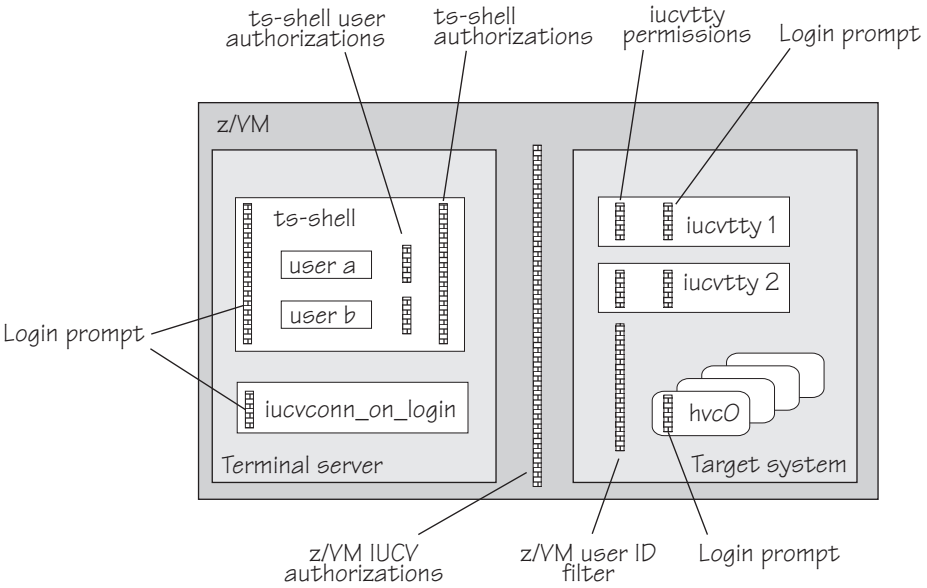

Figure 4. Security barriers in a terminal server environment - overview

For example, a ts-shell user first must log in to the terminal server and pass an SSH authentication. A connection request to an iucvtty instance is granted only if all the following apply:

- The user is authorized to connect to the target system.
- ts-shell is authorized to connect to the target system.
- The z/VM IUCV authorizations of the terminal server and the target system allow the IUCV connection between the two z/VM guest virtual machines.
- The iucvtty instance permits connections from the terminal server.

Once the connection is established, the user is prompted to log in and authenticate at the target system.

The only difference when connecting to an HVC terminal device is that there are no individual permissions. All HVC terminal devices use the same z/VM user ID filter to accept or reject a connection request.

For iucvconn_on_login users, the only security check on the terminal server is the authentication when logging in. The IUCV authorization and the checks on the target system are the same as for ts-shell users.

Chapter 4. Setting up a terminal server

This section describes the tasks you typically need to perform to set up a terminal server.

Setting up the z/VM guest virtual machine

The z/VM guest virtual machine for the terminal server requires:

- Sufficient storage (memory) for your Linux distribution.
- A network connection.
- Persistent disk space for session transcripts.

Figure 5 shows a typical directory entry for the z/VM virtual machine of a terminal server.

```
USER LXTS     XSECRETX 768M 1G G
* General statements
   IPL 0150
   CPU 00 BASE
   CPU 01
   MACH ESA 8
* IUCV authorization
   IUCV ANY
   OPTION MAXCONN 128
* Generic device statements
   CONSOLE 0009 3215 T
   SPOOL 000C 2540 READER *
   SPOOL 000D 2540 PUNCH A
   SPOOL 000E 1403 A
* Network connection
   NICDEF 7000 TYPE QDIO LAN SYSTEM VSWITCH1
* MiniDisks for Linux system and CMS A-disk
   MDISK 0150 3390 0001 3318 LXDASD1 MR
   MDISK 0151 3390 0001 1000 LXDASD2 MR
   MDISK 0191 3390 3000 0032 MDDASD  MR
```

Figure 5. Sample directory entry for a terminal server

The statements in this sample have the following meaning:

USER
: defines a z/VM user ID (LXTS), an initial password (XSECRETX), assigns 768 MB storage (memory) that, if required, can be expanded to 1 GB, and grants general user privileges (G).

IPL
: specifies the boot device for Linux.

CPU
: defines one or more virtual CPUs.

MACH ESA 8
: specifies a standard value for the machine architecture and the maximum number of CPUs that can be defined.

© Copyright IBM Corp. 2009

IUCV
allows the z/VM guest virtual machine to start an IUCV connection to any other z/VM guest virtual machine. See “IUCV security on z/VM” on page 7 for alternatives.

For more complete information about z/VM IUCV see *z/VM CP Programming Services*, SC24-6084 and *z/VM CP Planning and Administration*, SC24-6083.

OPTION MAXCONN
limits the number of concurrent IUCV connections to 128. If omitted, the limit defaults to 64, the maximum value for OPTION MAXCON is 65 535.

CONSOLE
specifies standard value for the z/VM console device.

SPOOL
specifies a standard value for the z/VM spool file queues.

NICDEF
specifies a virtual switch. The network device you use depends on your installation. For example, you can also use appropriate statements to specify HiperSockets™ or Open System Adapter (OSA) devices. See *z/VM Connectivity*, SC24-6080 for more information.

MDISK
Assigns read/write disk space for Linux and other data. The amount of disk space you require depends chiefly on the extend to which you want to create session transcripts.

For more information about z/VM user directory entries, see the chapter about the z/VM user directory in *z/VM CP Planning and Administration*, SC24-6083.

Installing the s390-tools package

For the Linux instance of the terminal server you need several components from the s390-tools package. If the s390-tools package is not included in your distribution, you can obtain it from `www.ibm.com/developerworks/linux/linux390/s390-tools.html`. The required programs are included as of version 1.8.1.

Installing the s390-tools package:

- Creates a directory `/etc/iucvterm` with configuration files for ts-shell
- Installs the iucvconn program
- Installs ts-shell
- Makes a copy of the iucvconn_on_login script available to you

If you install the s390-tools package as an RPM, the installation process might also:

- Make ts-shell an eligible login shell by adding it to `/etc/shells`
- Create a user group `ts-shell`
- Make the configuration files in `/etc/iucvterm` writable for user root and readable for the `ts-shell` user group
- Create a directory `/var/log/ts-shell` for session transcripts
- Make `/var/log/ts-shell` writable for the `ts-shell` user group and for user root

Setting up ts-shell

Before you begin: You need root authority to perform the tasks in this section.

The ts-shell program observes general and user-specific authorizations for connecting to target systems. You can also create session transcripts for sessions that are established with ts-shell.

Making ts-shell an eligible login shell

Before you begin: If you install the s390-tools package as an RPM, the installation process might perform this task for you.

To make ts-shell an eligible login shell add it to `/etc/shells`, for example, by entering the following command:

```
[root]# echo "/usr/bin/ts-shell" >> /etc/shells
```

Creating a user group with permissions for the ts-shell configuration files

Before you begin: If you install the s390-tools package as an RPM, the installation process might perform this task for you.

Perform the following steps to set the permissions for the ts-shell configuration files:

1. Create a user group for all ts-shell users.

   ```
   [root]# groupadd -r ts-shell
   ```

2. Make `ts-shell` the group for the configuration files.

   ```
   [root]# chgrp -R ts-shell /etc/iucvterm
   ```

3. Set the access permissions for the directory with the configuration files.

   ```
   [root]# chmod 0750 /etc/iucvterm
   ```

   This command makes the `/etc/iucvterm` directory writable for user root and readable for the `ts-shell` user group.

Restricting target system connections for ts-shell

Before you begin: By default ts-shell is permitted to connect to all target systems. Skip this task if you do not want to restrict this permission to specific target systems.

Perform the following steps to permit connections from ts-shell:

1. With your preferred editor, open `/etc/iucvterm/ts-shell.conf`.
2. Find the line

   ```
   ts-systems = /etc/iucvterm/unrestricted.conf
   ```

   and change it to

   ```
   ts-systems = /etc/iucvterm/ts-systems.conf
   ```

3. With your preferred editor, open `/etc/iucvterm/ts-systems.conf`.
4. List the z/VM user IDs, each on a separate line, of all target systems to which you want to permit connections.

   Example: A file to permit connections to LXGUEST1, LXGUEST3, LXGUEST5, LXGUEST7, and LXGUEST9 could read:

```
LXGUEST1
LXGUEST3
LXGUEST5
LXGUEST7
LXGUEST9
```

Tips:

- Lists of z/VM user IDs can be extensive. If you have access to the z/VM user directory, see Appendix C, “Creating files with lists of z/VM user IDs,” on page 51 for a convenient method of obtaining a list.
- You can permit connections to any target system by keeping the default configuration file `unrestricted.conf` or with a single entry, `[*ALL*]` in `ts-systems.conf`.

5. Save and close the configuration file.

Creating a user for ts-shell

Perform the following steps to create a user for ts-shell:

1. Add a new user with ts-shell as the login shell to user group ts-shell.

   Example:

   ```
   [root]# useradd -s /usr/bin/ts-shell -G ts-shell alice
   ```

2. **Optional:** You might want to add the user to additional user groups to manage access to target systems (see Appendix B, “ts-shell user authorization file syntax,” on page 49).
3. Set an initial password for the new user and force the new user to change the password at the initial login.

   Example:

   ```
   [root]# passwd alice
   ...
   [root]# chage alice
   ```

Grant authorizations to ts-shell users

This section describes how to authorize specific ts-shell users to connect to specific target systems. A user can connect to a target system for which both the user and ts-shell itself is authorized (see “Restricting target system connections for ts-shell” on page 13).

Perform the following steps to specify the target systems, specific ts-shell users are authorized to connect to:

1. With your preferred editor, open `/etc/iucvterm/ts-authorization.conf`.
2. Specify the authorization statements for your users and user groups (see Appendix B, “ts-shell user authorization file syntax,” on page 49).

   Tip: The s390-tools package includes a sample user authorization file. The location is similar to `/usr/share/doc/packages/s390-tools-<version>/ts-shell/authorization-sample.conf`. The value of *<version>* and whether `/packages` is present or absent in the path depend on your distribution.

3. Save and close the configuration file.

Configuring session transcripts

Before you begin: If you install the s390-tools package as an RPM, the installation process might perform steps 1 on page 15 to 3 on page 15 of this task for you.

This section describes how to configure session transcripts for specific target systems. Skip this section if you do not want to create session transcripts.

Perform the following steps to configure session transcripts:

1. Create a directory, `/var/log/ts-shell`, for the session transcripts.

   ```
   [root]# mkdir /var/log/ts-shell
   ```

2. Change the group for the new directory to the `ts-shell` group:

   ```
   [root]# chown root:ts-shell /var/log/ts-shell
   ```

3. Set the access permissions for the directory, and future subdirectories, to which the session transcripts are written:

   ```
   [root]# chmod 2770  /var/log/ts-shell
   ```

4. With your preferred editor, open `/etc/iucvterm/ts-audit-systems.conf`.
5. List the z/VM user IDs, each on a separate line, of all target systems for which session transcripts are to be created. The list entries are interpreted as uppercase and, therefore, not case sensitive.

   Example: A file that configures session transcripts for the target systems LXGUEST0 through LXGUEST4 could read:

   ```
   lxguest0
   lxguest1
   lxguest2
   lxguest3
   lxguest4
   ```

   Tips:

   - Lists of z/VM user IDs can be extensive. If you have access to the z/VM user directory, see Appendix C, “Creating files with lists of z/VM user IDs,” on page 51 for a convenient method of obtaining a list.
   - You can configure session transcripts for all target system with a single entry, `[*ALL*]`.

6. Save and close the configuration file.

Installing scriptreplay

You need scriptreplay if you want to replay terminal sessions from session transcripts.

The scriptreplay utility is included in the util-linux package. To find out if scriptreplay is installed on your Linux instance enter:

```
[root]# which scriptreplay
```

If scriptreplay is not included in your Linux distribution, you can obtain it from `www.kernel.org/pub/linux/utils/util-linux/`.

Setting up iucvconn_on_login

Before you begin: You need root authority to perform the tasks in this section.

You can set up the iucvconn_on_login script as an alternative to or in addition to ts-shell. The iucvconn_on_login script connects each user to one specific target system.

Setting up the script

Perform the following steps to set up iucvconn_on_login:

1. Copy the script from the s390-tools package documentation to `/usr/bin`. The path depends on your distribution and might or might not include a packages directory or version information for the s390-tools package. For example, enter:

```
[root]# cp /usr/share/doc/packages/s390-tools-1.8.1/ts-shell/iucvconn_on_login /usr/bin
```

2. Make the script executable.

```
[root]# chmod +x /usr/bin/iucvconn_on_login
```

3. Add the script to `/etc/shells`.

```
[root]# echo "/usr/bin/iucvconn_on_login" >> /etc/shells
```

Creating a user for iucvconn_on_login

Each target system to which you want to connect with iucvconn_on_login requires a separate Linux user on the terminal server. The user ID must match the z/VM user ID of the target system.

Perform the following steps to create a user for iucvconn_on_login:

1. Add a new user with iucvconn_on_login as the login shell. For example, to add a user for accessing a terminal device on `lxguest1`, enter:

```
[root]# useradd -s /usr/bin/iucvconn_on_login lxguest1
```

2. Set an initial password for the new user and force the new user to change the password at the initial login.

   Example:

```
[root]# passwd lxguest1
...
[root]# chage lxguest1
```

If you are using an external security manager for your z/VM system, for example, Resource Access Control Facility (RACF®), you can set up Linux to use the external security manager for authentication. See *Security on z/VM*, SG24-7471 for more information.

Modifying iucvconn_on_login for session transcripts

By default, no session transcripts are created for sessions that are established with the iucvconn_on_login script. If required, you can modify the script to create session transcripts. When modifying the script see “iucvconn - start terminal connection” on page 40 for the required **iucvconn** options.

The iucvconn_on_login user must have write access to the directory to which session logs are written.

Chapter 5. Setting up the target systems

Perform the tasks in this section for each target system.

The typical approach for handling a large number of target systems is to first configure a small number of systems that serve as templates and then use cloning techniques to create similar target systems. Cloning and other techniques for propagating configuration actions to numerous target systems are not covered in this document.

The descriptions in the following sections describe how to configure a target system through an SSH session. It is assumed that a TCP/IP connection is available when configuring the target system. After the configuration is completed, the target system can be accessed without an active TCP/IP connection.

Setting up the z/VM guest virtual machine

The specifications for the z/VM guest virtual machine entirely depend on the Linux instance and the applications that run on it.

If the necessary permissions for allowing an IUCV connection are in place for the terminal server, no additional statements are required for the target system (see “Setting up the z/VM guest virtual machine” on page 11).

If you do not want to use IUCV authorizations for the terminal server, add the following statement to the z/VM directory entry for your target system:

```
IUCV ALLOW
```

Be aware that this statement allows all z/VM guest virtual machines in the same z/VM instance to establish an IUCV connection to your target system.

Setting up iucvtty instances

Before you begin: You need root authority to perform the tasks in this section.

Installing iucvtty

The iucvtty program is part of the s390-tools package. If the s390-tools package is not included in your distribution, you can obtain it from `www.ibm.com/developerworks/linux/linux390/s390-tools.html`. The required programs are included as of version 1.8.1.

Enabling user logins

Depending on your distribution, you need an Upstart job file or an entry in `/etc/inittab` to facilitate user logins on a terminal device.

A full discussion of inittab entries or Upstart job files for starting login programs is beyond the scope of this document. This section highlights some of the issues you should be aware of and provides typical examples that you can use as a starting point. For more details see the inittab and events man pages.

For the syntax of the iucvtty program see “iucvtty - allow remote logins over z/VM IUCV” on page 42.

© Copyright IBM Corp. 2009

Examples for logins using inittab

This section shows examples of inittab entries that enable user logins. For corresponding Upstart examples see “Examples for logins using Upstart.”

Each inittab entry starts with an identifier that is unique within inittab. For more details see the man page for the inittab file.

- This inittab entry enables user logins on the iucvtty instance with terminal ID `lxterm1` with `/bin/login`:

  ```
  i1:2345:respawn:/usr/bin/iucvtty lxterm1
  ```

- This inittab entry enables user logins on the iucvtty instance with terminal ID `slnxterm` in single user mode. Instead of `/bin/login`, the default login program, the `/sbin/sulogin` login program is used.

  ```
  i1:S:once:/usr/bin/iucvtty slnxterm -- /sbin/sulogin
  ```

Examples for logins using Upstart

This section shows examples of Upstart job files that enable user logins. For corresponding inittab examples see “Examples for logins using inittab.”

You can use names of your choice for the file names of your Upstart job files. The directory where you must place the file depends on your distribution.

- This Upstart job file enables user logins on the iucvtty instance with terminal ID `lxterm1` with `/bin/login`:

  ```
  start on runlevel [2345]
  stop on runlevel [01]
  respawn
  exec /usr/bin/iucvtty lxterm1
  ```

- This Upstart job file enables user logins on the iucvtty instance with terminal ID `slnxterm` in single user mode. Instead of `/bin/login`, the default login program, the `/sbin/sulogin` login program is used.

  ```
  start on runlevel S
  stop on runlevel
  exec /usr/bin/iucvtty slnxterm -- /sbin/sulogin
  ```

Setting up HVC devices

Before you begin: You need root authority to perform the tasks in this section.

Specifying the number of HVC terminal devices

Use the `hvc_iucv` kernel parameter to specify the number of HVC terminal devices to be present.

hvc_iucv kernel parameter syntax

```
►►──hvc_iucv=<no>──────────────────────────────────────────────────►◄
```

<no> is an integer in the range 1 to 8 and specifies the number of terminal devices. The default for `hvc_iucv` depends on your distribution.

Activating hvc0 to receive Linux kernel messages

By default, the line-mode terminal device `ttyS0` is activated to receive Linux kernel messages and also is used as the preferred console. Use the `console` kernel parameter to also activate `hvc0` to receive Linux kernel messages. Of the HVC terminal devices, only `hvc0` can receive Linux kernel messages.

console kernel parameter syntax

```
console=hvc0
```

You can specify multiple console statements, each activating a terminal device to receive Linux kernel messages. The last console statement specifies the preferred console. If the following is the only console statement in the Linux kernel parameter string, `hvc0` is activated to receive Linux kernel messages and also becomes the preferred console:

```
console=hvc0
```

If you want to keep `ttyS0` as the preferred console, you need a second console statement:

```
console=hvc0 consloe=ttyS0
```

For more information about the console kernel parameter see *Device Drivers, Features, and Commands*, SC33-8411.

Restricting access to HVC devices

You can set a filter that restricts which z/VM guest virtual machines can connect to the z/VM IUCV HVC device driver and access HVC terminal devices. The same filter applies to all HVC terminal devices. If no filter is active, there are no restriction for accessing the HVC terminal devices.

The filter specifies the z/VM user IDs that are allowed to access the HVC terminal devices. Requests from all other z/VM user IDs are rejected. Be aware that the filter also applies to local connections. If an active filter does not include the z/VM user ID of the target system itself, local connections are refused.

Setting an initial z/VM user ID filter

You set the initial filter through the `hvc_iucv_allow` kernel parameter. Specify the z/VM user IDs that are allowed to connect to your HVC terminal devices as a comma-separated list.

hvc_iucv_allow kernel parameter syntax

```
                  ┌─,──────────────┐
                  ▼                │
hvc_iucv_allow=────<z/VM user ID>──┴──
```

Example: To accept requests from TERMSRV1 and TERMSRV2 specify:

```
hvc_iucv_allow=termsrv1,termsrv2
```

Displaying the current z/VM user ID filter

Use the **lsiucvallow** command to display the current z/VM user ID filter.

Example:

```
$ lsiucvallow
TERMSRV1
TERMSRV2
```

Creating a z/VM user ID filter file

You can specify a z/VM user ID filter as a filter file. Use your preferred text editor to create the filter file. The file lists the z/VM user IDs to be allowed to access the HVC terminal devices.

A valid filter file:

- Specifies each z/VM user ID on a separate line, with no white space before or after the z/VM user ID.
- Contains z/VM user IDs that all consist of up to eight alphanumeric characters or underscores (_).
- Contains no more than 500 z/VM user IDs.
- Can include empty lines and comment lines that start with a number sign (#).
- Does not exceed 4096 bytes.

Example: A filter file `/etc/iucvterm/ts-filters/filterb` might have the following content:

```
# Primary terminal server
termsrv1
# Backup terminal server
# termsrv2
# Replacement for backup terminal server termsrv2
termsrv3
```

"Changing the z/VM user ID filter with an editor" describes how to make the filter in a file the current filter.

Tip: You might want to list numerous z/VM user IDs in a filter file. If you have access to the z/VM user directory, see Appendix C, "Creating files with lists of z/VM user IDs," on page 51 for a convenient method of obtaining a list.

Changing the z/VM user ID filter with an editor

You can base the new z/VM user ID filter on the current filter or on specifications from a filter file.

Perform these steps to change the z/VM user ID filter:

1. Open a filter with the **chiucvallow** command.
   - Open the current filter:

     ```
     [root]# chiucvallow -e
     ```

   - Alternatively, open a filter file:

     ```
     [root]# chiucvallow -e <filter>
     ```

     where *<filter>* is the file path.

2. Use the editor to make any changes to the filter. **chiucvallow** opens the filter with vi unless you specify an alternative editor with the EDITOR environment variable.
3. Save your changes and close the editor. **chiucvallow** validates the new filter and replaces the current filter.

Replacing the current z/VM user ID filter

Enter a command of this form to replace the current z/VM user ID filter with a filter defined by a filter file:

```
[root]# chiucvallow -s <filter>
```

where *<filter>* specifies the filter file. **chiucvallow** first validates the new filter and then replaces the current filter. If necessary, use `chiucvallow -e <filter>` to correct verification errors. You can use `chiucvallow -V <filter>` to just validate the specifications in the filter file without replacing the current filter.

Example:

```
[root]# chiucvallow -s /etc/ts-filters/filterb
```

Tip: You can replace the filter as part of the boot process, for example as part of an init script (for example, rc.local or boot.local). This can be a useful alternative to specifying a filter with the kernel parameters, especially if the filter is extensive.

Revoking access restrictions

You can revoke access restrictions to the HVC terminal devices by clearing the z/VM user ID filter.

To clear the filter enter:

```
[root]# chiucvallow -c
```

Permitting root logins

The default login program for HVC terminal devices, `/bin/login`, restricts root logins. Root logins are allowed only on devices that are listed in `/etc/securetty`.

To permit root logins on an HVC terminal device add a separate line that specifies the device node for the device, omitting the leading `/dev/`. For example, to include `/dev/hvc0` specify `hvc0`.

See the securetty man page for more information. For other login programs see the respective documentation.

Enabling user logins

Depending on your distribution, you need an Upstart job file or an entry in `/etc/inittab` to facilitate user logins on a terminal device.

A full discussion of inittab entries or Upstart job files for starting login programs is beyond the scope of this document. This section highlights some of the issues you should be aware of and provides typical examples that you can use as a starting point.

Setting the terminal capabilities

You must set the terminal name, of the HVC terminal devices to a suitable value to obtain correct terminal output on the terminal emulator of your workstation. The terminal name indicates the capabilities of the terminal device. Examples for terminal names are `linux`, `dumb`, `xterm`, or `vt220`. You set the terminal name with the `TERM` environment variable.

Some getty programs accept the terminal name as a parameter and set the TERM environment variable accordingly at startup. For other getty programs you have to explicitly set the variable after the terminal session has been established, for example by entering the following command:

```
# export TERM=xterm
```

The value of the TERM variable is specific for each established terminal session and different sessions might use different values.

If `xterm` does not result in properly displayed terminal output, find out the setting for the terminal emulator on your workstation and set the `TERM` environment variable on the target system accordingly.

The iucvtty program automatically sets the `TERM` environment variable to a suitable value for you.

Examples for logins using inittab

This section shows examples of inittab entries that enable user logins. For corresponding Upstart examples see “Examples for logins using Upstart.”

Each inittab entry starts with an identifier that is unique within inittab. For more details see the man page for the inittab file.

- This inittab entry enables user logins on terminal device `hvc1` with mingetty.

  ```
  h1:2345:respawn:/sbin/mingetty --noclear hvc1
  ```

  With mingetty you must explicitly export the TERM environment variable as explained in “Setting the terminal capabilities.”
- This inittab entry enables user logins on terminal device `hvc2` with agetty and sets the TERM environment variable to `xterm` at startup.

  ```
  h2:2345:respawn:/sbin/agetty -L 9600 hvc2 xterm
  ```

  With agetty, you can specify the value to be set for the TERM environment variable as a parameter.
- This inittab entry enables user logins in single user mode on terminal device `hvc0`. Instead of `/bin/login`, the default login program, the `/sbin/sulogin` login program is used.

  ```
  h0:S:once:/sbin/sulogin hvc0
  ```

  The `/sbin/sulogin` login program requires a login by user root (see “Permitting root logins” on page 21).

Examples for logins using Upstart

This section shows examples of Upstart job files that enable user logins. For corresponding inittab examples see “Examples for logins using inittab.” You can use names of your choice for the file names of your Upstart job files. The directory where you must place the file depends on your distribution.

- This Upstart job file enables user logins on terminal device `hvc1` with mingetty.

```
start on runlevel [2345]
stop on runlevel [01]
respawn
exec /sbin/mingetty --noclear hvc1
```

With mingetty you must explicitly export the TERM environment variable as explained in “Setting the terminal capabilities” on page 22.

- This Upstart job file enables user logins on terminal device `hvc2` with agetty and sets the TERM environment variable to `xterm` at startup.

```
start on runlevel [2345]
stop on runlevel [01]
respawn
exec /sbin/agetty -L 9600 hvc2 xterm
```

With agetty, you can specify the value to be set for the TERM environment variable as a parameter.

- This Upstart job file enables user logins in single user mode on terminal device `hvc0`. Instead of `/bin/login`, the default login program, the `/sbin/sulogin` login program is used.

```
start on runlevel S
stop on runlevel
exec /sbin/sulogin hvc0
```

The `/sbin/sulogin` login program requires a login by user root (see “Permitting root logins” on page 21).

Chapter 6. Working with the terminal server

This section describes how users can access a terminal device on a target system from the terminal server. Which method is available to a particular user depends on how the user has been set up.

- A ts-shell user (see “Creating a user for ts-shell” on page 14) uses the **connect** command on ts-shell.
- An iucvconn_on_login user (see “Creating a user for iucvconn_on_login” on page 16) logs on to the terminal server and is automatically connected to the target system.
- A general Linux user on the terminal server uses the **iucvconn** command.

This section also describes how to work with session transcripts and how to identify log entries that pertain to terminal server activities.

Accessing a terminal device from ts-shell

This topic applies to users who log in to ts-shell on the terminal server (see “Creating a user for ts-shell” on page 14).

As a ts-shell user, perform the following steps to access a terminal device:

1. Log in to ts-shell on the terminal server.
2. **Optional:** Confirm that you are authorized to connect to the intended target system by entering the **list** command. The command lists all target systems for which you are authorized with a pager. Close the pager to return to ts-shell.

   Example:

   ```
   alice@ts-shell> list
   LXGUEST1
   LXGUEST3
   LXGUEST5
   LXGUEST7
   LXGUEST9
   ```

3. Connect to the target system and access the terminal device by entering a command of this form:

   ```
   alice@ts-shell> connect <vm_guest> <terminal_id>
   ```

   where:

   <vm_guest>
   specifies the z/VM user ID where the target Linux instance runs.

   <terminal_id>
   optionally identifies the terminal device.

   For HVC terminal devices the terminal IDs are lnxhvc*n*, where *n* is an integer in the range 0 through 7. The terminal ID for an iucvtty instance is set in the start command for the instance. See “iucvtty - allow remote logins over z/VM IUCV” on page 42 over.

   If omitted, a default terminal ID is used. Initially, the default is lnxhvc0. You can change the default for the terminal ID by entering a command of this form:

© Copyright IBM Corp. 2009

```
alice@ts-shell> terminal <terminal_id>
```

where *<terminal_id>* is the new default. To display the current default enter:

```
alice@ts-shell> terminal
```

The default applies to an individual ts-shell session only. It is not persistent across logins.

Example:

```
alice@ts-shell> connect lxguest1 lnxterm1
```

Result: Depending on how the terminal device on the target system has been set up, you are prompted to log in to the terminal.

Tip: If you have Perl ReadLine installed, you can press the Tab key to complete command names, terminal IDs, and z/VM guest IDs.

Accessing a terminal device using iucvconn_on_login

This topic applies to users who log in to iucvconn_on_login on the terminal server (see “Creating a user for iucvconn_on_login” on page 16).

The iucvconn_on_login program is designed to connect a specific terminal server user to a terminal device on a specific target system. Use the z/VM user ID of the target system as the user ID for opening an SSH session with the terminal server. Depending on how the terminal device has been set up on the target system you are then prompted to log in.

To establish a connection enter a command of this form from a command prompt on your workstation:

```
$ ssh -t <guest_id>@<terminal_server> <terminal_id>
```

where:

<guest_id>
: is the z/VM user ID that identifies the target system.

<terminal_server>
: is the host name or IP address of the terminal server.

<terminal_id>
: identifies the terminal device on the target system. If omitted, `lnxhvc0` is used.

Example:

```
$ ssh -t lxguest1@termsrv.example.net
...
lxguest1@termsrv.example.net's password:
iucvconn_on_login: Connecting to lxguest1 (terminal ID: lxterm1)

login: ...
...
[lxguest1]$ exit
logout
Connection to lxguest1 closed.
$
```

See “Basic iucvconn_on_login scenario” on page 34 for more details of this example.

Accessing a terminal device with iucvconn

Linux users with access to a regular shell (for example, bash) on the terminal server can use the **iucvconn** command to establish a terminal session with a target system. The **iucvconn** command is not directly available to ts-shell users or iucvconn_on_login users.

See “iucvconn - start terminal connection” on page 40 or the **iucvconn** man page for details.

Working with HVC terminal devices

Output that is written by Linux while the terminal session for an HVC terminal device is closed is not displayed. Therefore, a newly opened terminal window is always blank. For most applications, like login or shell prompts, it is sufficient to press Enter to obtain a new prompt.

You can also call the magic sysrequest functions from the hvc0 terminal device if it is present and has been activated to receive Linux kernel messages. To call the magic sysrequest functions from hvc0 enter the single character Ctrl+o followed by the character for the particular function. See `Documentation/sysrq.txt` in the Linux source tree for the available magic sysrequest functions.

Your distribution might not have enabled all of the listed functions. For information about enabling magic sysrequest functions see *Device Drivers, Features, and Commands*, SC33-8411 and the hvc_iucv man page.

Security hint: Always end sessions with HVC terminal devices by explicitly logging off (for example, type “exit” and press Enter). If logging off results in a new login prompt, press Control and Underscore (Ctrl+_) then press d to close the login window. Simply closing the terminal window for a hvc0 terminal device that has been activated for Linux kernel messages leaves the device active and the terminal session can be reopened without a login.

Working with session transcripts

Before you begin: To be able to work with session transcripts:

- You must be a regular user on the terminal server. ts-shell users and iucvconn_on_login users cannot work with session transcripts.
- You must have read access to `/var/log/ts-shell` where ts-shell creates the session transcripts.

Within `/var/log/ts-shell` there is a subdirectory for each user who has conducted a terminal session for which a transcript has been created.

The raw terminal data stream is written to a file within the directory for the respective user with a name of the format:

<vm_guest>_<YY-MM-DD-hhmmss>

where *<vm_guest>* is the z/VM user ID that identifies the target system and *<YY-MM-DD-hhmmss>* is a time stamp that indicates when the session was started.

The complete transcript includes two additional files:

*<vm_guest>_<YY-MM-DD-hhmmss>***.timing**
: with timing information about the session.

*<vm_guest>_<YY-MM-DD-hhmmss>***.info**
: with additional terminal session information.

The file with extension `.info` is a human readable text file. The transcript file without an extension and the file with extension `.timing` are intended for replaying a session. See the scriptreplay man page for details.

Consider a cron job to perform housekeeping and purge obsolete transcripts according to your audit policies.

Inspecting the logs

Events related to the terminal server are logged to syslog on both the terminal server itself and on the target systems. In particular, the iucvtty program and the z/VM IUCV HVC device driver log refused IUCV connection attempts.

In addition, unsuccessful login attempts are logged to `/var/log/secure` by the login program. These log records include the involved terminal IDs.

To find relevant entries on the terminal server examine `/var/log/secure`. For example, enter:

```
[root]# grep "iucvconn" /var/log/secure
May 25 10:42:42 termsrv1 iucvconn[27340]: Established connection to lxguest1/lxterm1 for user alice (uid=503)
May 25 10:44:13 termsrv1 iucvconn[27342]: Established connection to lxguest1/lnxhvc0 for user alice (uid=503)
May 25 10:52:42 termsrv1 iucvconn[27358]: Established connection to lxguest3/lxterm1 for user alice (uid=503)
May 25 11:38:09 termsrv1 iucvconn[27522]: Established connection to linux00/lnxhvc0 for user bob (uid=505)
May 25 12:01:34 termsrv1 iucvconn[27589]: Established connection to lxguest1/lxterm1 for user lxguest1 (uid=507)
```

To find relevant entries on a target system examine `/var/log/secure` for iucvtty instances. For example, enter:

```
[root]# grep "iucvtty" /var/log/secure
May 25 10:38:57 lxguest3 iucvtty[23618]: Listening on terminal ID: lxterm1, using pts device: /dev/pts/10
May 25 10:52:42 lxguest3 iucvtty[23618]: Accepted client connection from termsrv1
May 25 11:13:19 lxguest3 iucvtty[23621]: Listening on terminal ID: lxterm1, using pts device: /dev/pts/10
[root]# grep "login: LOGIN ON pts" /var/log/secure
May 25 10:53:08 lxguest3 login: LOGIN ON pts/10 BY alice FROM termsrv1
```

To find relevant entries on a target system examine `/var/log/secure` and `/var/log/messages` for HVC terminal devices. For example, enter:

```
[root]# grep "LOGIN ON hvc" /var/log/secure
May 25 10:44:22 lxguest1 login: ROOT LOGIN ON hvc0
[root]# grep "hvc_iucv" /var/log/messages
May 25 13:44:16 lxguest1 kernel: hvc_iucv.09cae6: A connection request from z/VM user ID LXGUEST7 was refused
```

Chapter 7. Scenarios

This section contains scenarios that show how the different components of a terminal server environment work together.

Basic scenario

This basic scenario assumes:

- A z/VM guest virtual machine TERMSRV1 has been set up as a terminal server. In particular:
  - Linux with the s390-tools package has been installed, `ts-shell` is listed in `/etc/shells` and a user group `ts-shell` is in place.
  - The directory entry for the terminal server includes the IUCV ANY statement that permits IUCV connections to any other z/VM virtual machine within the z/VM instance.
- A z/VM guest virtual machine LXGUEST1 has been set up on the same z/VM instance. In particular:
  - Linux with the s390-tools package has been installed
  - The Linux distribution uses inittab.

The steps in the scenario show how to set up a ts-shell user `alice` on the terminal server with access to three terminal devices on the target system: an iucvtty instance `lxterm1` and two HVC terminal devices `hvc0` and `hvc1` (see Figure 6).

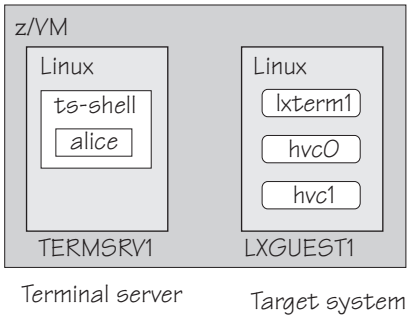

Figure 6. Basic scenario

Setting up the terminal server

Perform these steps to set up the terminal server:

1. As user root, log in to Linux on the terminal server.
2. Add and set up user `alice`.

```
[root]# useradd -s /usr/bin/ts-shell -G ts-shell alice
[root]# passwd alice
...
[root]# chage alice
```

3. Permit user `alice` to connect to LXGUEST1 by opening `/etc/iucvterm/ts-authorization.conf` and adding the following line:

```
alice = list:lxguest1
```

© Copyright IBM Corp. 2009

Setting up the target system

Perform these steps to set up the target system:

1. Log on to z/VM guest virtual machine LXGUEST1.
2. IPL Linux with the kernel parameters `hvc_iucv=2` and `hvc_iucv_allow=termsrv1` to obtain two HVC terminal devices and to allow connections from TERMSRV1 only.
3. As user root, establish an SSH session with the target system.
4. Confirm that the HVC terminal devices are accessible only through connections from TERMSRV1.

```
[root]# lsiucvallow
TERMSRV1
```

5. Add lines to inittab to allow user logins on the three terminal devices. For example, add these lines:

```
i1:2345:respawn:/usr/bin/iucvtty -a TERMSRV1 lxterm1
h0:2345:respawn:/sbin/mingetty --noclear hvc0
h1:2345:respawn:/sbin/mingetty --noclear hvc1
```

6. Instruct init to reexamine `/etc/inittab`:

```
[root]# init q
```

Establishing terminal sessions

User `alice` can now log in to ts-shell on the terminal server and access the terminal devices on LXGUEST1.

Accessing `lxterm1`:

```
alice@ts-shell> connect lxguest1 lxterm1
ts-shell: Connecting to lxguest1 (terminal identifier: lxterm1)...
login as:
...
[LXGUEST1]$
...
[LXGUEST1]$ exit
ts-shell: Connection ended
alice@ts-shell>
```

Accessing `hvc0`, using the default setting for the terminal ID:

```
alice@ts-shell> terminal
lnxhvc0
alice@ts-shell> connect lxguest1
ts-shell: Connecting to lxguest1 (terminal identifier: lnxhvc0)...
login as:
...
[LXGUEST1]$ export TERM=xterm
...
[LXGUEST1]$ exit
ts-shell: Connection ended
alice@ts-shell>
```

If exiting the terminal session at the target system results in a renewed login prompt to the target system, you might have to press Control and Underscore (Ctrl+_) then press d to disconnect and return to the ts-shell (see also “Security hint” on page 27).

Extended scenario

This scenario extends “Basic scenario” on page 29:

- There is now a backup terminal server TERMSRV2. TERMSRV1 and TERMSRV2 must both be permitted to connect to all target systems.
- In addition to LXGUEST1, there are additional target systems: LXGUEST0, LXGUEST2 through LXGUEST9, and LINUX00 through LINUX99.
- User `alice` is responsible for LXGUEST1, LXGUEST3, LXGUEST5, LXGUEST7, and LXGUEST9
- There is an additional ts-shell user, `bob`, who is responsible for LINUX00 through LINUX99, LXGUEST0, LXGUEST2, LXGUEST4, LXGUEST6, and LXGUEST8.
- `ts-shell` is to be permitted to connect to the target systems only.
- Session transcripts are to be created for LXGUEST0 through LXGUEST4.

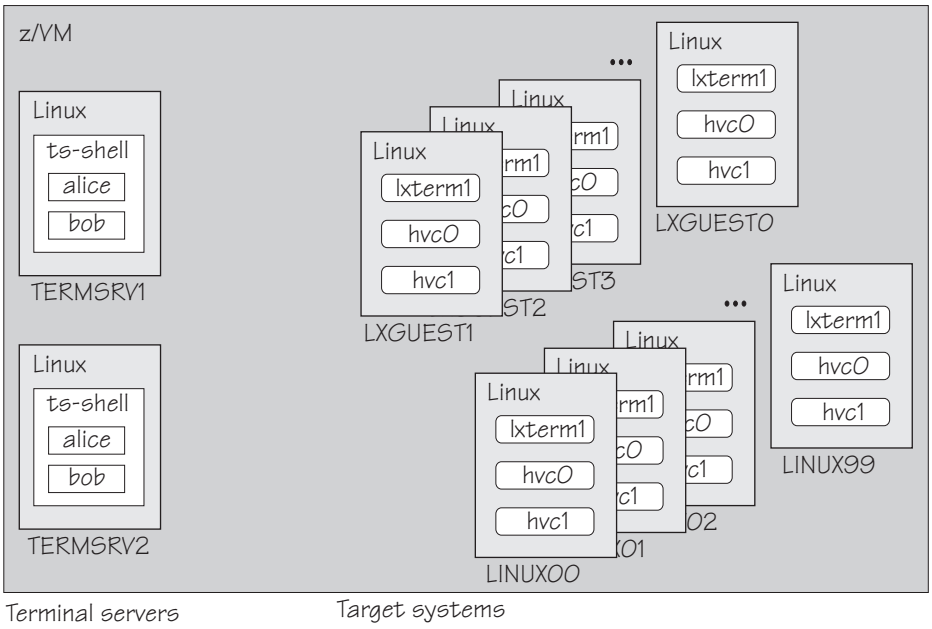

Figure 7. Extended scenario

This scenario assumes that the terminal servers and target systems are set up as described in “Basic scenario” on page 29.

Extending the terminal server configuration

Perform these steps for each terminal server:

1. As user root, log in to Linux on the terminal server.
2. Add and set up user `bob`.

```
[root]# useradd -s /usr/bin/ts-shell -G ts-shell bob
[root]# passwd bob
...
[root]# chage bob
```

3. Grant user permission by changing the content of `/etc/iucvterm/ts-authorization.conf` to:

```
alice = list:lxguest1,lxguest3,lxguest5,lxguest7,lxguest9
bob = regex:lxguest[02468]
bob = regex:^linux[0-9]{2}$
```

4. With your preferred editor, open `/etc/iucvterm/ts-shell.conf`.
5. Find the line

   ```
   ts-systems = /etc/iucvterm/unrestricted.conf
   ```

   and change it to

   ```
   ts-systems = /etc/iucvterm/ts-systems.conf
   ```

6. In `/etc/iucvterm/ts-systems.conf` list the z/VM user IDs of all target systems, each z/VM user ID on a separate line.

   ```
   [root]# echo lxguest{0..9}|tr ' ' '\n' > /etc/iucvterm/ts-systems.conf
   [root]# echo linux0{0..9}|tr ' ' '\n' >> /etc/iucvterm/ts-systems.conf
   [root]# echo linux{10..99}|tr ' ' '\n' >> /etc/iucvterm/ts-systems.conf
   ```

7. Ensure that `/etc/iucvterm/ts-systems.conf` is readable by members of the `ts-shell` user group.
8. If not already present as a result of installing s390-tools, set up a directory, `/var/log/ts-shell`, for the session transcripts.

   ```
   [root]# mkdir /var/log/ts-shell
   [root]# chown root:ts-shell /var/log/ts-shell
   [root]# chmod 2770  /var/log/ts-shell
   ```

9. Configure session transcripts for LXGUEST0 through LXGUEST4 by adding the following lines to `/etc/iucvterm/ts-audit-systems.conf`:

   ```
   lxguest0
   lxguest1
   lxguest2
   lxguest3
   lxguest4
   ```

Extending the target system configuration

Perform these steps for each target system:

1. Log on to the z/VM guest virtual machine for the target system.
2. IPL Linux with the kernel parameters `hvc_iucv=2` and `hvc_iucv=allow_termsrv1,termsrv2` to obtain two HVC terminal devices and to allow connections from both TERMSRV1 and TERMSRV2.
3. Log in to Linux as user root.
4. Confirm that the HVC terminal devices are accessible only through connections from TERMSRV1.

   ```
   [root]# lsiucvallow
   TERMSRV1
   TERMSRV2
   ```

5. Modify the inittab entry for `lxterm1` to allow connections from both terminal servers TERMSRV1 and TERMSRV2. For example, change

   ```
   i1:2345:respawn:/usr/bin/iucvtty -a TERMSRV1 lxterm1
   ```

   to

   ```
   i1:2345:respawn:/usr/bin/iucvtty -a TERMSRV[12] lxterm1
   ```

6. Instruct init to reexamine `/etc/inittab`:

```
[root]# init q
```

Establishing terminal sessions

User `alice` can now log in to ts-shell on the terminal servers and access the terminal devices on LXGUEST1, LXGUEST3, LXGUEST5, LXGUEST7, and LXGUEST9.

User `bob` can now log in to ts-shell on the terminal servers and access the terminal devices on LINUX00 through LINUX99, LXGUEST0, LXGUEST2, LXGUEST4, LXGUEST6, and LXGUEST8.

User `alice` accessing `lxterm1` on LXGUEST3:

```
alice@ts-shell> connect lxguest3 lxterm1
ts-shell: Connecting to lxguest3 (terminal identifier: lxterm1)...
login as:
...
[LXGUEST3]$
...
[LXGUEST3]$ exit
ts-shell: Connection ended
alice@ts-shell>
```

An attempt by user `bob` to access `lxterm1` on LXGUEST3 is rejected:

```
bob@ts-shell> connect lxguest3 lxterm1
ts-shell: You are not authorized to connect to lxguest3
bob@ts-shell>
```

User `bob` accessing `hvc0` on LINUX00:

```
bob@ts-shell> connect linux00 lnxhvc0
ts-shell: Connecting to linux00 (terminal identifier: lnxhvc0)...
login as:
...
[LINUX00]$ export TERM=xterm
...

[LINUX00]$ exit
ts-shell: Connection ended
bob@ts-shell>
```

If exiting the terminal session at the target system results in a renewed login prompt to the target system, you might have to press Control and Underscore (Ctrl+_) then press d to disconnect and return to the ts-shell (see also “Security hint” on page 27).

Locating the session transcripts

Session transcripts have been configured for terminal sessions with LXGUEST0 through LXGUEST4. These transcripts are located in subdirectories of `/var/log/ts-shell`.

To show who has established terminal sessions with these systems, enter:

```
$ ls /var/log/ts-shell
alice
```

To show the transcripts for the sessions established by user `alice` enter:

```
$ ls /var/log/ts-shell/alice
lxguest3_09-05-25-105242
lxguest3_09-05-25-105242.info
lxguest3_09-05-25-105242.timing
```

Basic iucvconn_on_login scenario

This simple scenario illustrates the use of iucvconn_on_login and extends “Basic scenario” on page 29. In addition to the connections through ts-shell, there is to be a connection using the iucvconn_on_login script.

The setup of the terminal server and target server are assumed to be as described in “Basic scenario” on page 29. The fully qualified host name of the terminal server is assumed to be `termserv1.example.net` and the fully qualified host name of the target system `lxguest1.example.net`.

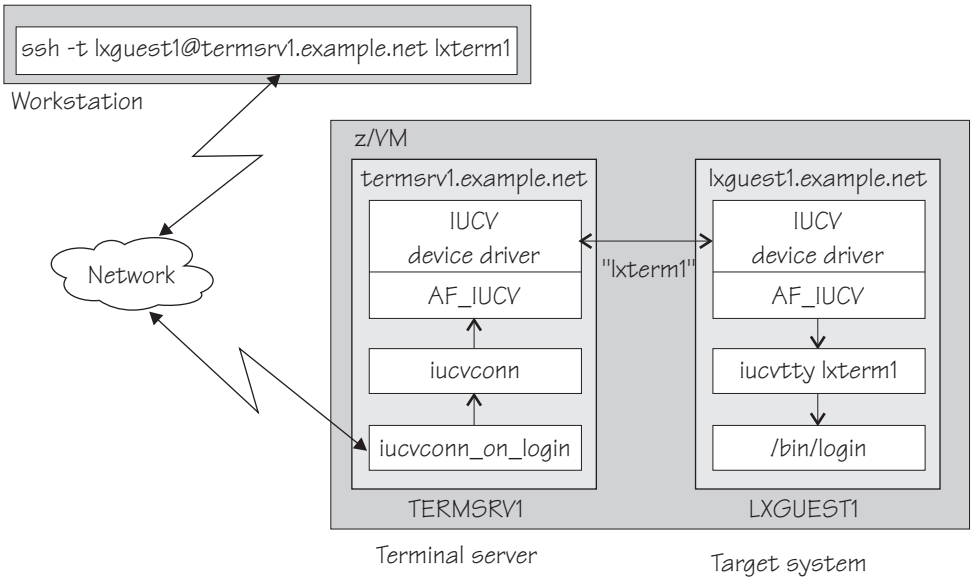

Figure 8. Accessing a terminal device with the iucvconn_on_login script

Extending the terminal server configuration

Perform these steps for the terminal server:

1. As user root, log in to Linux on the terminal server.
2. Copy the iucvconn_on_login script from the s390-tools package documentation to `/usr/bin`. The path depends on your distribution and might or might not include a packages directory or version information for the s390-tools package. For example, enter:

   ```
   [root]# cp /usr/share/doc/packages/s390-tools-1.8.1/ts-shell/iucvconn_on_login /usr/bin
   ```

3. Make the script executable.

   ```
   [root]# chmod +x /usr/bin/iucvconn_on_login
   ```

4. Add the script to `/etc/shells`.

```
[root]# echo "/usr/bin/iucvconn_on_login" >> /etc/shells
```

5. Add `lxguest1` as a new user with iucvconn_on_login as the login shell:

```
[root]# useradd -s /usr/bin/iucvconn_on_login lxguest1
```

6. Set an initial password for the new user and force the new user to change the password at the initial login.

   Example:

```
[root]# passwd lxguest1
...
[root]# chage lxguest1
```

Extending the target system configuration

No changes are required on the target system.

Establishing terminal sessions

Accessing `lxterm1` (default terminal for iucvconn_on_login) on `lxguest1`:

```
$ ssh -t lxguest1@termsrv1.example.net
..
lxguest1@termsrv.example.net's password:
iucvconn_on_login: Connecting to lxguest1 (terminal ID: lxterm1)

login: ...
...
[lxguest1]$ exit
logout
Connection to termsrv1.example.net closed.
$
```

Appendix A. Command reference

© Copyright IBM Corp. 2009

chiucvallow - work with z/VM user ID filters

Runs on target systems to list, verify, and change the z/VM user ID filter of the z/VM IUCV HVC device driver. The filter specifies the z/VM user IDs that are allowed to access HVC terminal devices.

chiucvallow requires root authority.

Format

chiucvallow syntax

```
►►─chiucvallow──┬─ -l ──────────────────┬──────────────────────────────────►◄
                │─ -e ──┬──────────────┤│
                │       └─ <filter> ───┘│
                │─ -V── <filter> ───────┤│
                │─ -s── <filter> ───────┤│
                └─ -c ──────────────────┘│
```

where:

-l or **--list**
displays the z/VM user IDs contained in the current filter.

chiucvallow with the `-l` option is equivalent to **lsiucvallow** (see “lsiucvallow - display the z/VM user ID filter” on page 44).

<filter>
specifies a z/VM user ID filter file.

z/VM user ID filter files list z/VM user IDs to be allowed to access the HVC terminal devices. Each z/VM user ID is specified on a separate line. There can also be comment lines that start with a number sign (#) and blank lines.

-e or **--edit**
edit the current z/VM user ID filter.

If *<filter>* is specified, the z/VM user ID filter in *<filter>* is opened in an editor, otherwise the current z/VM user ID filter is imported into the editor.

When the editor is closed, the edited filter is verified (see “-V or --verify”). If verified successfully, the edited z/VM user ID filter becomes the current filter. If the verification fails, the edited z/VM user ID filter is saved to a backup copy that can then be corrected.

By default, vi is used as the editor. You can specify an alternative editor with the EDITOR environment variable.

-V or **--verify**
verifies that the z/VM user ID filter specified by *<filter>*:

- All listed z/VM user IDs consist of up to eight alphanumeric characters or underscores (_).
- Contains no more than 500 z/VM user IDs.
- Does not exceed 4096 bytes.

-s or **--set**
replaces the current z/VM user ID filter with the filter specified by *<filter>*. The current z/VM user ID filter can be replaced only after *<filter>* has been successfully verified.

-c or **--clear**
clears the current z/VM user ID filter. After the filter has been cleared, any z/VM user ID is allowed to connect to the z/VM IUCV HVC device driver.

-v or **--version**
displays the version of **chiucvallow** and exits.

-h or **--help**
displays out a short help text and exits. For more detail see the chiucvallow man page.

Examples

- A filter file `/etc/ts-filters/filterb` might have the following content:

```
# Primary terminal server
termsrv1
# Backup terminal server
# termsrv2
# Replacement for backup terminal server termsrv2
termsrv3
```

- To make `/etc/ts-filters/filterb` the current filter:

```
[root]# chiucvallow -V /etc/ts-filters/filterb
Verify z/VM user ID: termsrv1 : OK
Verify z/VM user ID: termsrv3 : OK

chiucvallow: Verification summary: verified=2 failed=0 size=18 bytes
[root]# chiucvallow -s /etc/ts-filters/filterb
```

- To list the current filter:

```
[root]# chiucvallow -l
TERMSRV1
TERMSRV3
```

- To clear the filter:

```
[root]# chiucvallow -c
```

iucvconn - start terminal connection

Runs on the terminal server to access a terminal device on a target system. This command is used by ts-shell and by the iucvconn_on_login script.

Format

iucvconn syntax

```
►►─iucvconn─┬─ -e _ ──────────┬──┬──────────────────── <vm_guest>── <terminal_id>──────►◄
            ├─ -e <escape_char>─┤  └─ -s <log_file>─┘
            └─ -e none──────────┘
```

where:

-e or **--escape-char** *<escape_char>*
sets an escape character for the terminal session. You need an escape character to access special **iucvconn** functions. The default escape character is the underscore (_) character. If *<escape_char>* is set to “none”, escaping is not possible. The escape character can be the closing bracket (]), the caret (^), the underscore (_), or any alphabetic character except C, D, Q, S, and Z. The escape character is not case sensitive.

To call a special function press *<escape_char>* while holding down Ctrl, then press the key for the function:

Table 1. Special functions that can be accessed through the escape character

Function character	Function
d	Close the terminal session.
period (.)	Close the terminal session (same as d).
r	Force resizing of the connected terminal.

-s or **--sessionlog** *<log_file>*
creates a transcript of the terminal session and writes session data to three different files.

<log_file>
contains the raw terminal data stream.

*<log_file>***.timing**
contains timing data that can be used for replaying the raw terminal data stream using realistic output delays.

*<log_file>***.info**
contains additional terminal session information.

If any of these files exist, the iucvconn program exits with an error. To proceed either delete the files or choose another file name for *<log_file>*.

<vm_guest>
specifies the z/VM user ID where the target Linux instance runs.

<terminal_id>
identifies a running iucvtty instance, or an HVC terminal device. The *<terminal_id>* is like a port number in TCP/IP communications. *<terminal_id>* is case sensitive and consists of up to 8 alphanumeric characters.

For HVC terminal devices the terminal IDs are lnxhvc*n*, where *n* is an integer in the range 0 through 7. The terminal ID for an iucvtty instance is set in the start command for the instance.

-v or **--version**
prints the version number of the iucvconn program and exits.

-h or **--help**
prints out a short help text and exits. For more detail see the iucvconn man page.

Examples

- To access the `lxterm1` terminal on the Linux instance in z/VM guest virtual machine LXGUEST1:

```
$ iucvconn lxguest1 lxterm1
```

- To access the `lxterm1` terminal on the Linux instance in z/VM guest virtual machine LXGUEST1 and setting the escape character to `X`:

```
$ iucvconn -e x lxguest1 lxterm1
```

- To access the first z/VM IUCV HVC terminal device on the Linux instance in z/VM guest virtual machine LXGUEST2:

```
$ iucvconn lxguest2 lnxhvc0
```

- To access the first z/VM IUCV HVC terminal device on the Linux instance in z/VM guest virtual machine LINUX99 and create a set of session transcript files `~/transcripts/linux99`, `~/transcripts/linux99.timing`, and `~/transcripts/linux99.info`:

```
$ iucvconn -s ~/transcripts/linux99 linux99 lnxhvc0
```

iucvtty - allow remote logins over z/VM IUCV

Runs on target systems to start iucvtty instances. Typically, the **iucvtty** command is called through inittab entries or Upstart job files.

Format

iucvtty syntax

```
►►─iucvtty──┬──────────────┬──<terminal_id>──┬─ -- /bin/login──────────────────────┬──►◄
            └─ -a <regex>──┘                 └─ -- <login_program>──┬──────────────┬┘
                                                                     └─<login_options>─┘
```

where:

-a or **--allow-from** *<regex>*
: is a regular expression that limits permissions for incoming connections to matching z/VM user IDs. The connection is refused if the z/VM user ID does not match. If this parameter is omitted, connections are permitted from any z/VM user ID.

<terminal_id>
: identifies the z/VM IUCV connection. *<terminal_id>* is case sensitive and consists of up to eight alphanumeric characters. The *<terminal_id>* must be specified as a parameter in access requests against an iucvtty instance. The *<terminal_id>* is like a port number in TCP/IP communications.

<login_program>
: specifies the absolute path to the login program to be started when a connection is established. The default is `/bin/login`.

<login_options>
: specifies additional options that depend on the particular login program used.

-v or **--version**
: displays the version number of iucvtty and exits.

-h or **--help**
: displays a short help text and exits. For more detail see the iucvconn man page.

Examples

- To allow remote logins using terminal ID `lxterm1`:

```
[root]# iucvtty lxterm1
```

- To only allow users from LXGUEST1 to access `lxterm1`:

```
[root]# iucvtty -a lxguest1 lxterm1
```

- To only allow users from LINUX10 through LINUX19 to access `lxterm1`:

```
[root]# iucvtty -a "linux1[0-9]" lxterm1
```

- To use `/sbin/sulogin` instead of `/bin/login` for `suterm`:

```
[root]# iucvtty suterm -- /sbin/sulogin
```

lsiucvallow - display the z/VM user ID filter

Runs on target systems to display the current z/VM user ID filter of the z/VM IUCV HVC device driver. The filter specifies the z/VM user IDs that are allowed to connect to the z/VM IUCV HVC device driver.

lsiucvallow requires root authority.

Format

lsiucvallow syntax

```
►►──lsiucvallow──────────────────────────────────────────────────────►◄
```

Examples

In this example, access from TERMSRV1 and TERMSRV2 is allowed.

```
$ lsiucvallow
TERMSRV1
TERMSRV2
```

ts-shell: connect - establish a terminal session

Runs within ts-shell on the terminal server to connect to a target system and accesses a terminal device on the target system.

Format

connect syntax

```
►►──connect── <vm_guest> ┬──────────────┬──────────────────────────────►◄
                         └─ <terminal_id>─┘
```

where:

<vm_guest>
specifies the target system.

<terminal_id>
specifies the terminal ID of the terminal to be accessed.

For HVC terminal devices the terminal IDs are lnxhvc*n*, where *n* is an integer in the range 0 through 7. The terminal ID for an iucvtty instance is set in the start command for the instance (see “iucvtty - allow remote logins over z/VM IUCV” on page 42).

If omitted, a default terminal ID is used. Initially, the default is lnxhvc0. You can change the default for the terminal ID with the ts-shell **terminal** command.

Examples

To connect to an iucvtty terminal with terminal ID `lxterm1` on LXGUEST1:

```
alice@ts-shell> connect lxguest1 lxterm1
```

ts-shell: list - list authorized target systems

Runs within ts-shell on the terminal server to list all target systems for which a ts-shell user is authorized. Lists are displayed with a pager. Close the pager to return to ts-shell.

The default pager used is **less** in secure mode (using the LESSSECURE environment variable). You can use the PAGER environment variable to specify the full path to an alternative pager.

Format

list syntax

```
►►──list──────────────────────────────────────────────────────────►◄
```

Examples

- Listing authorizations that are defined in list format:

```
alice@ts-shell> list
LXGUEST1
LXGUEST3
LXGUEST5
LXGUEST7
LXGUEST9
```

- Listing authorizations that are defined as regular expressions:

```
bob@ts-shell> list
Regular expressions for your authorization:
(?i-xsm:lxguest[02468])
(?i-xsm:^linux[0-9]{2}$)
```

- Listing authorizations that are defined as regular expressions if additional restrictions exist for ts-shell. Those IDs in `/etc/iucvterm/ts-systems.conf` that match one of the regular expressions is appended to the user authorizations.

  If `/etc/iucvterm/ts-systems.conf` reads:

```
LXGUEST1
LXGUEST2
LXGUEST3
LXGUEST5
LINUX07
LINUX11
LINUX13
```

  the previous example becomes:

```
bob@ts-shell> list
Regular expressions for your authorization:
(?i-xsm:lxguest[02468])
(?i-xsm:^linux[0-9]{2}$)
You are authorized to connect to these z/VM guest virtual machines:
LXGUEST2
LINUX07
LINUX11
LINUX13
```

ts-shell: terminal - display and set the default terminal ID

Runs within ts-shell on the terminal server to display and set the default terminal ID used for the connect command.

Format

connect syntax

```
►►──terminal──┬──────────────┬──────────────────────────────────────►◄
              └─<terminal_id>─┘
```

where:

<terminal_id>
is the default terminal ID to be set. If omitted, the current default terminal ID is displayed.

For HVC terminal devices the terminal IDs are lnxhvc*n*, where *n* is an integer in the range 0 through 7. The terminal ID for an iucvtty instance is set in the start command for the instance (see “iucvtty - allow remote logins over z/VM IUCV” on page 42).

Examples

- To display the current terminal ID:

```
alice@ts-shell> terminal
lnxhvc0
```

- To set `lxterm1` as the default terminal ID:

```
alice@ts-shell> terminal lxterm1
```

ts-shell: version, help, exit, quit

In addition to **connect**, **list**, and **terminal**, ts-shell provides the following commands:

version
displays the version of ts-shell.

help
displays a summary of the available ts-shell commands.

exit
closes the terminal server shell session.

quit
closes the terminal server shell session.

Appendix B. ts-shell user authorization file syntax

Authorizations for ts-shell users to connect to target systems are assigned in a user authorization file. This file can include:

- Authorization statements
- Comment lines that start with a number sign (#)
- Blank lines

An authorization statement has the general form:

```
<users> = <list_type>:<targets>
```

where:

<users>
: specifies who is authorized to establish connections. *<users>* can be an individual Linux user ID or a Linux user group. To distinguish users from groups, groups are prefixed with an at sign (@).

<list_type>:<targets>
: specifies the target systems to which connections are authorized. Target systems can be specified as a comma-separated list, in a list file, or as a regular expression.

list:
: is followed by a comma-separated list of individual z/VM user IDs. Consider this method for specifying a small number of target systems.

file:
: is followed by a file path to a configuration file that contains a list of z/VM user IDs, each on a separate line. Consider this method to specify numerous target systems.

Tip: Lists of z/VM user IDs can be extensive. If you have access to the z/VM user directory, see Appendix C, “Creating files with lists of z/VM user IDs,” on page 51 for a convenient method of obtaining a list.

regex:
: is followed by a regular expression that matches z/VM user IDs. Consider this method to specify target systems that follow a naming convention.

Examples:

- The following authorization statement permits user `alice` to connect to target systems LXGUEST1, LXGUEST3, LXGUEST5, LXGUEST7, and LXGUEST9.

  ```
  alice = list:lxguest1,lxguest3,lxguest5,lxguest7,lxguest9
  ```

- The following authorization statement permits all users in group `testgrp` to connect to the target systems listed in a file `/etc/iucvterm/auth/test-systems.list`.

  ```
  @testgrp = file:/etc/iucvterm/auth/test-systems.list
  ```

- The following authorization statement permits user `bob` to connect to the target systems: LXGUEST0, LXGUEST2, LXGUEST4, LXGUEST6, and LXGUEST8.

  ```
  bob = regex:lxguest[02468]
  ```

You can have multiple authorizations for the same user, either directly through multiple authorization statements for the same user or indirectly through authorization statements for groups that the user is a member of.

© Copyright IBM Corp. 2009

For a particular user, you can mix explicit authorizations of types list or file but you cannot mix either of these explicit authorizations with regular expressions. The first type of authorization that is found for a user, explicit or regular expression, sets the authorization type for this user. Further authorizations of the same type are accumulated. Authorizations of the other type are ignored.

Example: The following example assumes that both user `alice` and user `bob` are members of group `users`.

```
@users = list:lxguest0,lxguest1,lxguest2
alice = list:lxguest1,lxguest3,lxguest5,lxguest7,lxguest9
bob = regex:lxguest[02468]
```

For user `alice` the group and individual authorizations accumulate to LXGUEST0, LXGUEST1, LXGUEST2, LXGUEST3, LXGUEST5, LXGUEST7, and LXGUEST9.

For user `bob` the regular expression is ignored and the authorizations are for LXGUEST0, LXGUEST1, and LXGUEST2 as defined for the group.

Appendix C. Creating files with lists of z/VM user IDs

You might need to create lists of z/VM user IDs to specify:

- Target systems that ts-shell can connect to
- Target systems a particular user can connect to
- Target systems for which session logs are to be created
- A z/VM filter file

Such lists can be extensive and writing them manually is both tedious and error prone. If you have access to the z/VM user directory, and your z/VM user IDs follow a naming convention, you can use the **vmur**, **grep**, and **cut** commands to create a list from the z/VM user directory.

The **grep** and **cut** commands are core Linux commands. The **vmur** command is included in the s390-tools package.

Example: The following example assumes that the z/VM user directory has been sent to the reader of your z/VM guest virtual machine and you want to list all z/VM user IDs that begin with LINUX and end with one or more numerals.

```
[root]# vmur receive -H -t 1234 -O |grep -E "^USER LINUX[0-9]+" |cut -d" " -f2 > userlist
```

In the command, **vmur** reads out the file with spool ID 1234 from the reader, **grep** extracts all lines that specify z/VM user IDs according to the pattern, **cut** reduces the line to just the z/VM user ID and the greater than symbol (>) directs the output to a file, `userlist`.

You can find out the spool IDs of the files in your z/VM reader with the command:

```
[root]# vmur list -q rdr
```

Tip: Another convenient way to create lists of IDs that follow a pattern is bash brace expansion. For example to create a list of IDs including `lnxa34` through `lnxa46`, `lnxb34` through `lnxb46`, and `lnxc34` through `lnxc46` enter:

```
$ echo lnx{a..c}{34..46} | tr ' ' '\n'
```

© Copyright IBM Corp. 2009

Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:
INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® or ™), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law

© Copyright IBM Corp. 2009

trademarks in other countries. A current list of IBM trademarks is available on the Web at ″Copyright and trademark information″ at
www.ibm.com/legal/copytrade.shtml

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Other company, product, and service names may be trademarks or service marks of others.

Index

Special characters

A

C

D

E

H

I

K

L

M

P

© Copyright IBM Corp. 2009

Readers' Comments — We'd Like to Hear from You

Linux on System z
How to Set up a Terminal Server Environment on z/VM
June 2009
Linux Kernel 2.6 – Development stream

Publication No. SC34-2596-00

We appreciate your comments about this publication. Please comment on specific errors or omissions, accuracy, organization, subject matter, or completeness of this book. The comments you send should pertain to only the information in this manual or product and the way in which the information is presented.

For technical questions and information about products and prices, please contact your IBM branch office, your IBM business partner, or your authorized remarketer.

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you. IBM or any other organizations will only use the personal information that you supply to contact you about the issues that you state on this form.

Comments:

Thank you for your support.

Submit your comments using one of these channels:
- Send your comments to the address on the reverse side of this form.
- Send your comments via e-mail to: eservdoc@de.ibm.com

If you would like a response from IBM, please fill in the following information:

Name	Address
Company or Organization	
Phone No.	E-mail address

Readers' Comments — We'd Like to Hear from You
SC34-2596-00

Cut or Fold
Along Line

Fold and Tape **Please do not staple** Fold and Tape

PLACE
POSTAGE
STAMP
HERE

IBM Deutschland Research & Development GmbH
Information Development
Department 3248
Schoenaicher Strasse 220
71032 Boeblingen
Germany

Fold and Tape **Please do not staple** Fold and Tape

SC34-2596-00

Cut or Fold
Along Line

SC34-2596-00