mudar para português

```
XINETD.CONF(5)                                XINETD.CONF(5)

NAME
       xinetd.conf - Extended Internet Services Daemon configuration file

DESCRIPTION
       xinetd.conf  is  the  configuration  file that determines the services
       provided by xinetd.  Any line whose first non-white-space character is
       a '#' is considered a comment line. Empty lines are ignored.

       The file contains entries of the form:

          service <service_name>
          {
             <attribute> <assign_op> <value> <value> ...
             ...
          }

       The  assignment  operator,  assign_op,  can be one of '=', '+=', '-='.
       The majority of attributes support only the simple  assignment  opera-
       tor,  '='.   Attributes    whose  value  is  a set of values support all
       assignment operators.  For such attributes, '+=' means adding a    value
       to  the  set  and '-=' means removing a value from the set.  A list of
       these attributes will be given after all the attributes are described.

       Each entry defines a service identified by the service_name.  The fol-
       lowing is a list of available attributes:

       id       This attribute is used to uniquely  identify  a     ser-
            vice.    This  is  useful because there exist services
            that can use  different     protocols  and     need  to  be
            described with different entries in the configuration
            file.  By default, the service id is the same as  the
            service name.

       type     Any combination of the following values may be used:

            RPC        if this is an RPC service

            INTERNAL    if    this is a service provided by xinetd.

            TCPMUX/TCPMUXPLUS
                    if this is a service that will be started
                    according to the RFC 1078 protocol on the
                    TCPMUX well-known port. See     the  section
                    describing TCPMUX services below.

            UNLISTED    if    this  is  a  service  not listed in a
                    standard system file (like    /etc/rpc  for
                    RPC     services,  or /etc/services for non-
                    RPC services).

       flags        Any combination of the following flags may be used:
```

```
INTERCEPT   Intercept packets or accepted connections
        in    order  to verify that they are coming
        from acceptable  locations    (internal  or
        multi-threaded  services cannot be inter-
        cepted).

NORETRY       Avoid retry     attempts  in  case  of     fork
        failure.

IDONLY       Accept  connections     only when the remote
        end identifies the remote user (i.e.  the
        remote  host  must    run an identification
        server).  This flag applies only to     con-
        nection-based  services.   This  flag  is
        ineffective if the USERID log  option  is
        not used.

NAMEINARGS  This  will    cause  the  first argument in
        "server_args" to be argv[0] when  execut-
        ing the server, as specified in "server".
        This allows you to use  tcpd  by  putting
        tcpd  in  "server"    and  the  name of the
        server in "server_args"  like  in  normal
        inetd.

NODELAY       If    the  service is a tcp service and the
        NODELAY flag is set, then the TCP_NODELAY
        flag  will    be set on the socket.  If the
        service is not a tcp service, this option
        has no effect.

KEEPALIVE   If    the  service is a tcp service and the
        KEEPALIVE    flag   is   set,   then      the
        SO_KEEPALIVE  socket  flag will be set on
        the socket.     If the service is not a  tcp
        service, this option has no effect.

NOLIBWRAP   This  disables  internal  calling  of the
        tcpwrap library to    determine  access  to
        the service.  This may be needed in order
        to use libwrap functionality  not  avail-
        able  to  long-running  processes such as
        xinetd; in this case,  the    tcpd  program
        can     be  called  explicitly (see also the
        NAMEINARGS flag).  For RPC services using
        TCP transport, this flag is automatically
        turned  on,     because  xinetd  cannot  get
        remote  host  address information for the
        rpc port.

SENSOR       This replaces the service with  a  sensor
        that  detects  accesses  to the specified
        port. NOTE: It will     NOT  detect  stealth
        scans.  This  flag should be used only on
        services that you know  you     don't    need.
        When  an access is made to this service's
```

```
                port, the IP Address is added to a global
                no_access  list.  This  causes all subse-
                quent accesses from     the  originating  IP
                address  to     be  denied  access until the
                deny_time setting expires. The amount  of
                time  spent     on this list is configurable
                as the deny_time  attribute.  The  SENSOR
                flag  will    also cause xinetd to consider
                the server attribute to  be     INTERNAL  no
                matter  what  is  typed on the same line.
                Another important thing  to     remember  is
                that if the socket_type is set to stream,
                then the wait attribute should be set  to
                no.

          IPv4        Sets  the  service    to be an IPv4 service
                (AF_INET).

          IPv6        Sets the service to be  an    IPv6  service
                (AF_INET6),     if  IPv6 is available on the
                system.

          REUSE        The REUSE flag is deprecated.   All     ser-
                vices  now implicitly use the REUSE flag.

     disable      This is boolean "yes" or "no".    This will  result  in
          the service being disabled and not starting.  See the
          DISABLE flag description.

     socket_type  Possible values for this attribute include:

          stream        stream-based service

          dgram        datagram-based service

          raw        service that requires direct access to IP

          seqpacket   service that requires reliable sequential
                datagram transmission

     protocol     determines the protocol that is employed by the     ser-
          vice.  The protocol must exist in /etc/protocols.  If
          this attribute is not defined, the  default  protocol
          employed by the service will be used.

     wait     This  attribute     determines if the service is single-
          threaded or multi-threaded and whether or not  xinetd
          accepts     the connection or the server program accepts
          the connection. If its value is yes, the  service  is
          single-threaded;  this    means  that xinetd will start
          the server and then it will  stop  handling  requests
          for  the  service  until the server dies and that the
          server software will accept the     connection.  If  the
          attribute  value is no, the service is multi-threaded
          and xinetd will keep handling  new  service  requests
          and  xinetd  will accept the connection. It should be
```

```
          noted that udp/dgram  services    normally  expect  the
          value to be yes since udp is not connection oriented,
          while tcp/stream servers normally expect the value to
          be no.

     user     determines  the     uid for the server process. The user
          attribute can either be numeric or a name. If a     name
          is  given (recommended),  the user name must exist in
          /etc/passwd.  This attribute is     ineffective  if  the
          effective user ID of xinetd is not super-user.

     group        determines  the gid for the server process. The group
          attribute can either be numeric or a name. If a     name
          is  given (recommended), the group name must exist in
          /etc/group.  If a group is not specified,  the    group
          of  user  will    be  used  (from     /etc/passwd).     This
          attribute is ineffective if the effective user ID  of
          xinetd is not super-user.

     instances    determines the number of servers that can be simulta-
          neously active for  a  service    (the  default  is  no
          limit).     The  value of this attribute can be either a
          number or UNLIMITED which  means  that    there  is  no
          limit.

     nice     determines  the server priority. Its value is a (pos-
          sibly negative) number; check nice(3) for more infor-
          mation.

     server       determines the program to execute for this service.

     server_args  determines  the     arguments  passed  to the server. In
          contrast to inetd, the    server    name  should  not  be
          included in server_args.

     libwrap      overrides  the    service name passed to libwrap (which
          defaults to the server name,  the  first  server_args
          component  with     NAMEINARGS, the id for internal ser-
          vices and the service name for redirected  services).
          This  attribute is only valid if xinetd has been con-
          figured with the libwrap option.

     only_from    determines the remote hosts to which  the  particular
          service     is  available.      Its  value  is a list of IP
          addresses which can be specified in  any  combination
          of the following ways:

          a)   a numeric address in the form of %d.%d.%d.%d. If
               the rightmost components are 0, they are treated
               as     wildcards (for example, 128.138.12.0 matches
               all hosts on the  128.138.12  subnet).   0.0.0.0
               matches  all Internet addresses.  IPv6 hosts may
               be        specified      in     the     form       of
               abcd:ef01::2345:6789.   The  rightmost  rule for
               IPv4 addresses does not apply to IPv6 addresses.
```

```
          b)   a      factorized   address     in   the   form   of
               %d.%d.%d.{%d,%d,...}.  There is no need for  all
               4    components  (i.e. %d.%d.{%d,%d,...%d} is also
               ok).  However, the factorized part     must  be  at
               the end of the address.  This form does not work
               for IPv6 hosts.

          c)   a network name (from /etc/networks).  This     form
               does not work for IPv6 hosts.

          d)   a    host  name.   When  a  connection  is made to
               xinetd, a reverse lookup is performed,  and  the
               canonical name returned is compared to the spec-
               ified host name.  You may also use domain    names
               in     the  form  of    .domain.com.   If the reverse
               lookup of the client's IP is within .domain.com,
               a match occurs.

          e)   an     ip  address/netmask  range  in     the  form of
               1.2.3.4/32.  IPv6 address/netmask ranges in  the
               form of 1234::/46 are also valid.

          Specifying  this  attribute without a value makes the
          service available to nobody.

     no_access    determines the remote hosts to which  the  particular
          service is unavailable. Its value can be specified in
          the same way as the value of the only_from attribute.
          These  two  attributes    determine the location access
          control enforced by xinetd. If none  of     the  two  is
          specified  for a service, the service is available to
          anyone. If both are specified for a service, the  one
          that  is  the  better  match  for  the address of the
          remote host determines if the service is available to
          that  host  (for  example, if the only_from list con-
          tains 128.138.209.0 and the no_access  list  contains
          128.138.209.10     then    the  host  with     the  address
          128.138.209.10 can not access the service).

     access_times determines the time intervals  when  the  service  is
          available. An interval has the form hour:min-hour:min
          (connections will be accepted at  the  bounds  of  an
          interval).  Hours  can range from 0 to 23 and minutes
          from 0 to 59.

     log_type     determines where the  service  log  output  is    sent.
          There are two formats:

          SYSLOG    syslog_facility [syslog_level]
                 The log output is sent to syslog at the speci-
                 fied   facility.      Possible   facility    names
                 include:     daemon,  auth, authpriv, user, mail,
                 lpr, news, uucp, ftp local0-7.  Possible level
                 names  include: emerg, alert, crit, err, warn-
                 ing, notice, info, debug.  If a level  is  not
                 present,     the messages will be recorded at the
```

```
                info level.

          FILE  file [soft_limit [hard_limit]]
                The log output is appended to file which     will
                be created if it does not exist. Two limits on
                the size of the log  file  can  be  optionally
                specified.   The     first    limit  is a soft one;
                xinetd will log a message the first time     this
                limit  is  exceeded (if xinetd logs to syslog,
                the message will be sent at the alert priority
                level).     The  second  limit  is a hard limit;
                xinetd will stop logging for the affected ser-
                vice  (if  the  log file is a common log file,
                then more than one service  may    be  affected)
                and  will  log a message about this (if xinetd
                logs to syslog, the message will     be  sent  at
                the alert priority level).  If a hard limit is
                not specified, it defaults to the  soft    limit
                increased  by  1%  but  the extra size must be
                within  the   parameters      LOG_EXTRA_MIN      and
                LOG_EXTRA_MAX  which  default  to  5K  and 20K
                respectively (these constants are  defined  in
                xconfig.h).

     log_on_success   determines  what  information is logged when a server
          is started and when that server exits (the service id
          is  always  included in the log entry).     Any combina-
          tion of the following values may be specified:

          PID        logs the server process id (if  the     ser-
                  vice  is  implemented  by  xinetd without
                  forking another process the     logged     pro-
                  cess id will be 0)

          HOST        logs the remote host address

          USERID        logs the user id of the remote user using
                  the     RFC  1413  identification  protocol.
                  This  option is available only for multi-
                  threaded stream services.

          EXIT        logs the fact that a server exited    along
                  with  the  exit status or the termination
                  signal (the process id is also logged  if
                  the PID option is used)

          DURATION    logs the duration of a service session

          TRAFFIC        logs  the  total  bytes  in and out for a
                  redirected service.

     log_on_failure   determines what information is logged when  a  server
          cannot    be  started  (either  because  of  a  lack of
          resources or because of access control restrictions).
          The  service  id  is always included in the log entry
          along with the reason for failure.   Any  combination
```

```
     of the following values may be specified:

     HOST        logs the remote host address.

     USERID        logs the user id of the remote user using
             the     RFC  1413  identification  protocol.
             This  option is available only for multi-
             threaded stream services.

     ATTEMPT        logs the fact that a failed     attempt  was
             made  (this option is implied by all oth-
             ers).

rpc_version  determines the RPC version for    a  RPC    service.  The
     version can be a single number or a range in the form
     number-number.

rpc_number   determines the number for  an  UNLISTED     RPC  service
     (this  attribute  is  ignored  if  the service is not
     unlisted).

env      The value of this attribute is a list of  strings  of
     the  form  'name=value'.  These strings will be added
     to the environment before starting a  server  (there-
     fore  the  server's environment will include xinetd's
     environment plus the specified strings).

passenv      The value of this attribute is a list of  environment
     variables  from     xinetd's  environment    that  will be
     passed to the server.  An empty list implies  passing
     no  variables  to the server except for those explic-
     itly defined using the env attribute.    (notice     that
     you  can  use  this attribute in conjunction with the
     env attribute to  specify  exactly  what  environment
     will be passed to the server).

port     determines  the     service  port.     If this attribute is
     specified for a service listed in  /etc/services,  it
     must be equal to the port number listed in that file.

redirect     Allows a tcp service  to  be  redirected  to  another
     host.    When xinetd receives a tcp connection on this
     port it spawns a process that establishes  a  connec-
     tion  to the host and port number specified, and for-
     wards all data between the two hosts.  This option is
     useful when your internal machines are not visible to
     the  outside  world.   Syntax  is:  redirect  =      (ip
     address) (port).  You can also use a hostname instead
     of the IP address in this field.  The hostname lookup
     is  performed  only once, when xinetd is started, and
     the first IP address returned is the one that is used
     until xinetd is restarted.  The "server" attribute is
     not required when this option is specified.   If  the
     "server" attribute is specified, this attribute takes
     priority.
```

```
       bind     Allows a service to be bound to a specific  interface
            on  the     machine.   This  means you can have a telnet
            server listening on a local, secured  interface,  and
            not  on     the  external interface.  Or one port on one
            interface can do something, while the same port on  a
            different  interface can do something completely dif-
            ferent.     Syntax: bind = (ip address of interface).

       interface    Synonym for bind.

       banner       Takes the name of a file to be splatted at the remote
            host  when  a  connection  to  that service is estab-
            lished.     This banner is printed regardless of  access
            control.   It  should *always* be printed when a con-
            nection has been made.    xinetd outputs the  file  as-
            is,  so you must ensure the file is correctly format-
            ted for the service's protocol.     In paticular, if the
            protocol  requires  CR-LF pairs for line termination,
            you must supply them.

       banner_success   Takes the name of a file to be splatted at the remote
            host  when  a  connection to that service is granted.
            This banner is printed as soon as access  is  granted
            for  the  service.  xinetd outputs the file as-is, so
            you must ensure the file is correctly  formatted  for
            the  service's protocol.  In paticular, if the proto-
            col requires CR-LF pairs for  line  termination,  you
            must supply them.

       banner_fail  Takes the name of a file to be splatted at the remote
            host when a connection to  that     service  is  denied.
            This  banner  is  printed  immediately upon denial of
            access.     This is useful for informing your users that
            they  are  doing  something bad and they shouldn't be
            doing it anymore.  xinetd outputs the file as-is,  so
            you  must  ensure the file is correctly formatted for
            the service's protocol.     In paticular, if the  proto-
            col  requires  CR-LF  pairs for line termination, you
            must supply them.

       per_source   Takes an integer or "UNLIMITED" as an argument.     This
            specifies  the    maximum instances of this service per
            source IP address.  This can also be specified in the
            defaults section.

       cps      Limits    the  rate of incoming connections.  Takes two
            arguments.  The first argument is the number of     con-
            nections per second to handle.    If the rate of incom-
            ing connections is higher than this, the service will
            be  temporarily disabled.  The second argument is the
            number of seconds to wait before re-enabling the ser-
            vice  after  it     has  been disabled.  The default for
            this setting  is  50  incoming    connections  and  the
            interval is 10 seconds.

       max_load     Takes a floating point value as the load at which the
```

```
          service will stop accepting connections.   For    exam-
          ple:  2 or 2.5.     The service will stop accepting con-
          nections at this load.    This is the one     minute     load
          average.   This     is an OS dependent feature, and cur-
          rently only Linux, Solaris, and FreeBSD are supported
          for  this.   This feature is only avaliable if xinetd
          was configured with the -with-loadavg option.

     groups       Takes either "yes" or "no".  If the groups  attribute
          is  set     to  "yes",  then the server is executed with
          access to the groups that the server's effective  UID
          has  access  to.   If  the groups attribute is set to
          "no", then the    server    runs  with  no    supplementary
          groups.     This attribute must be set to "yes" for many
          BSD systems.   This  attribute    can  be     set  in  the
          defaults section as well.

     mdns     Takes  either "yes" or "no".  On systems that support
          mdns registration of services (currently only Mac  OS
          X),  this  will enable or disable registration of the
          service.  This defaults to "yes".

     umask        Sets the inherited umask for the service.  Expects an
          octal    value.      This    option    may  be     set  in  the
          "defaults" section to set a umask for  all  services.
          xinetd    sets its own umask to the previous umask OR'd
          with 022.  This is the umask that will    be  inherited
          by  all     child    processes  if the umask option is not
          used.

     enabled      Takes a list of service ID's to     enable.   This     will
          enable    only the services listed as arguments to this
          attribute; the rest will be disabled.  If you have  2
          ftp  services,    you  will  need to list both of their
          ID's and not just ftp. (ftp is the service name,  not
          the ID. It might accidentally be the ID, but you bet-
          ter check.) Note that the service "disable" attribute
          and  "DISABLE"    flag can prevent a service from being
          enabled despite being listed in this attribute.

     include      Takes  a   filename   in   the     form    of   "include
          /etc/xinetd/service".    The  file is then parsed as a
          new configuration file.     It is not the same thing  as
          pasting     the  file into xinetd.conf where the include
          directive is given.  The included file must be in the
          same  form as xinetd.conf.  This may not be specified
          from within a service.    It must be specified  outside
          a service declaration.

     includedir   Takes  a  directory  name  in the form of "includedir
          /etc/xinetd.d".     Every file  inside  that  directory,
          excluding  files with names containing a dot ('.') or
          ending with a tilde ('~'), will be parsed  as  xinetd
          configuration  files.    The  files  will be parsed in
          alphabetical order according to the  C    locale.     This
          allows    you to specify services one per file within a
```

```
     directory.  The includedir directive may not be spec-
     ified from within a service declaration.

rlimit_as    Sets  the  Address  Space resource limit for the ser-
     vice. One parameter is required, which    is  either  a
     positive  integer representing the number of bytes to
     set the limit to (K or M may be used to specify kilo-
     bytes/megabytes)  or  "UNLIMITED".   Due  to  the way
     Linux's libc malloc is implemented, it is more useful
     to  set     this  limit than rlimit_data, rlimit_rss and
     rlimit_stack. This resource limit is only implemented
     on Linux systems.

rlimit_cpu   Sets  the maximum number of CPU seconds that the ser-
     vice may use.  One parameter is     required,  which  is
     either    a positive integer representing the number of
     CPU seconds limit to, or "UNLIMITED".

rlimit_data  Sets the maximum data size  resource  limit  for  the
     service.   One parameter is required, which is either
     a positive integer representing the number  of    bytes
     or "UNLIMITED".

rlimit_rss   Sets the maximum resident set size limit for the ser-
     vice.  Setting this value low will make the process a
     likely candidate for swapping out to disk when memory
     is low.     One parameter is required, which is either a
     positive  integer representing the number of bytes or
     "UNLIMITED".

rlimit_stack Set the maximum stack size  limit  for    the  service.
     One parameter is required, which is either a positive
     integer representing the number of bytes  or  "UNLIM-
     ITED".

deny_time    Sets the time span that access to all services on all
     IP addresses are denied to someone that sets off  the
     SENSOR.     The  unit  of    time  is  in  minutes.    Valid
     options are: FOREVER, NEVER,  and  a  numeric  value.
     FOREVER     causes the IP address not to be purged until
     xinetd is restarted. NEVER has    the  effect  of     just
     logging     the  offending     IP  address.  A typical time
     value would be 60 minutes. This should stop most  DOS
     attacks     while allowing IP addresses that come from a
     pool to be recycled  for  legitimate  purposes.     This
     option    must  be  used in conjunction with the SENSOR
     flag.

You don't need to specify all of the above attributes  for  each     ser-
vice.  The necessary attributes for a service are:

   socket_type
   user        (non-internal services only)
   server        (non-internal services only)
   wait
   protocol        (RPC and unlisted services only)
```

```
   rpc_version    (RPC services only)
   rpc_number    (unlisted RPC services only)
   port        (unlisted non-RPC services only)

The following attributes support all assignment operators:

   only_from
   no_access
   log_on_success
   log_on_failure
   passenv
   env        (does not support the '-=' operator)

These  attributes  can  also appear more than once in a service entry.
The remaining attributes support only the '=' operator and can  appear
at most once in a service entry.

The  configuration  file may also contain a single defaults entry that
has the form

   defaults
   {
      <attribute> = <value> <value> ...
      ...
   }

This entry provides default attribute values for service entries  that
don't specify those attributes. Possible default attributes:

   log_type        (cumulative effect)
   bind
   per_source
   umask
   log_on_success    (cumulative effect)
   log_on_failure    (cumulative effect)
   only_from        (cumulative effect)
   no_access        (cumulative effect)
   passenv        (cumulative effect)
   instances
   disabled        (cumulative effect)
   enabled        (cumulative effect)
   banner
   banner_success
   banner_fail
   per_source
   groups
   cps
   max_load

   Attributes  with    a  cumulative  effect can be specified
   multiple times
   with the values specified each time accumulating (i.e. '=' does
   the same thing as '+=').    With the exception of  disabled     they
   all  have     the same meaning as if they were specified in a ser-
   vice entry.  disabled determines    services  that    are  disabled
   even  if    they  have  entries  in     the configuration file. This
```

```
          allows for quick reconfiguration by specifying disabled ser-
          vices with the disabled attribute instead of commenting them
          out. The value of this attribute is a list of space separated
          service ids. enabled has the same properties as disabled. The
          difference being that enabled is a list of which services are
          to be enabled. If enabled is specified, only the services
          specified are available. If enabled is not specified, all ser-
          vices are assumed to be enabled, except those listed in dis-
          abled.


INTERNAL SERVICES
       xinetd provides the following services internally (both stream and
       datagram based): echo, time, daytime, chargen, and discard.
These
       services are under the same access restrictions as all other services
       except for the ones that don't require xinetd to fork another process
       for them. Those ones (time, daytime, and the datagram-based echo,
       chargen, and discard) have no limitation in the number of instances.


TCPMUX Services
       xinetd supports TCPMUX services that conform to RFC 1078. These ser-
       vices may not have a well-known port associated with them, and can be
       accessed via the TCPMUX well-known port.

       For each service that is to be accessed via TCPMUX, a service entry in
       /etc/xinetd.conf or in a configuration file in an includedir directory
       must exist.

       The service_name field (as defined above for each service in any
       xinetd configuration file) must be identical to the string that is
       passed (according to RFC 1078 protocol) to xinetd when the remote ser-
       vice requestor first makes the connection on the TCPMUX well-known
       port. Private protocols should use a service name that has a high
       probability of being unique. One way is to prepend the service name
       with some form of organization ID.

       The type field can be either TCPMUX or TCPMUXPLUS. If the type is TCP-
       MUXPLUS, xinetd will handle the initial protocol handshake (as defined
       in RFC 1078) with the calling process before initiating the service.
       If the type is TCPMUX, the server that is started is responsible for
       performing the handshake.

       The type field should also include UNLISTED if the service is not
       listed in a standard system file (like /etc/rpc for RPC services, or
       /etc/services for non-RPC services).

       The socket_type for these services must be stream, and the protocol
       must be tcp.

       Following is a sample TCPMUX service configuration:

          service myorg_server
          {
             disable         = no
```

```
            type         = TCPMUX
            socket_type    = stream
            protocol        = tcp
            wait         = no
            user         = root
            server         = /usr/etc/my_server_exec
         }
```

Besides a service entry for each service that can be accessed via the TCPMUX well-known port, a service entry for TCPMUX itself must also be included in the xinetd configuration. Consider the following sample:

```
         service tcpmux
         {
            type         = INTERNAL
            id             = tcpmux
            socket_type    = stream
            protocol        = tcp
            user         = root
            wait         = no
         }
```

NOTES

1. The following service attributes cannot be changed on reconfiguration: socket_type, wait, protocol, type.

2. When the attributes only_from and no_access are not specified for a service (either directly or via defaults) the address check is considered successful (i.e. access will not be denied).

3. The address check is based on the IP address of the remote host and not on its domain address. We do this so that we can avoid remote name lookups which may take a long time (since xinetd is single-threaded, a name lookup will prevent the daemon from accepting any other requests until the lookup is resolved). The down side of this scheme is that if the IP address of a remote host changes, then access to that host may be denied until xinetd is reconfigured. Whether access is actually denied or not will depend on whether the new host IP address is among those allowed access. For example, if the IP address of a host changes from 1.2.3.4 to 1.2.3.5 and only_from is specified as 1.2.3.0 then access will not be denied.

4. If the USERID log option is specified and the remote host either does not run an identification server or the server sends back a bad reply, access will not be denied unless the IDONLY service flag is used.

5. Interception works by forking a process which acts as a filter between the remote host(s) and the local server. This obviously has a performance impact so it is up to you to make the compromise between security and performance for each service. The following tables show the overhead of interception. The first table shows

```
       the time overhead-per-datagram for a UDP-based service using vari-
       ous    datagram sizes.     For TCP-based services we measured the band-
       width reduction because of interception while  sending  a  certain
       amount of data from client to server (the time overhead should the
       same as for UDP-based services but it is "paid" only by the    first
       packet  of a continuous data transmission).    The amount of data is
       given in the table as system_callsxdata_sent_per_call, i.e.     each
       send(2)  system call transferred so many bytes of data.  The band-
       width reduction is given in terms of bytes per  second  and    as  a
       percentage  of  the    bandwidth when interception is not performed.
       All measurements were done on a  SparcStation  IPC  running    SunOS
       4.1.

          Datagram size (bytes)       Latency (msec)
          ---------------------       --------------
          64                1.19
          256                1.51
          1024                1.51
          4096                3.58


          Bytes sent           Bandwidth reduction
          ----------           -------------------
          10000x64           941 (1.2%)
          10000x256           4,231 (1.8%)
          10000x1024           319,300 (39.5%)
          10000x4096           824,461 (62.1%)

EXAMPLE
          #
          # Sample configuration file for xinetd
          #

          defaults
          {
             log_type         = FILE /var/log/servicelog
             log_on_success     = PID
             log_on_failure     = HOST
             only_from         = 128.138.193.0 128.138.204.0
             only_from         = 128.138.252.1
             instances         = 10
             disabled         = rstatd
          }

          #
          # Note 1: the protocol attribute is not required
          # Note 2: the instances attribute overrides the default
          #
          service login
          {
             socket_type     = stream
             protocol         = tcp
             wait         = no
             user         = root
             server         = /usr/etc/in.rlogind
             instances         = UNLIMITED
```

```
}

#
# Note 1: the instances attribute overrides the default
# Note 2: the log_on_success flags are augmented
#
service shell
{
   socket_type     = stream
   wait         = no
   user         = root
   instances         = UNLIMITED
   server         = /usr/etc/in.rshd
   log_on_success     += HOST
}

service ftp
{
   socket_type     = stream
   wait         = no
   nice         = 10
   user         = root
   server         = /usr/etc/in.ftpd
   server_args     = -l
   instances         = 4
   log_on_success     += DURATION HOST USERID
   access_times     = 2:00-9:00 12:00-24:00
}

# Limit telnet sessions to 8 Mbytes of memory and a total
# 20 CPU seconds for child processes.
service telnet
{
   socket_type     = stream
   wait         = no
   nice         = 10
   user         = root
   server         = /usr/etc/in.telnetd
   rlimit_as         = 8M
   rlimit_cpu         = 20
}

#
# This entry and the next one specify internal services. Since
# this is the same service using a different socket type, the
# id attribute is used to uniquely identify each entry
#
service echo
{
   id             = echo-stream
   type         = INTERNAL
   socket_type     = stream
   user         = root
   wait         = no
}
```

```
          service echo
          {
             id             = echo-dgram
             type         = INTERNAL
             socket_type     = dgram
             user         = root
             wait         = no
          }

          #
          # Sample RPC service
          #
          service rstatd
          {
             type         = RPC
             socket_type     = dgram
             protocol         = udp
             server         = /usr/etc/rpc.rstatd
             wait         = yes
             user         = root
             rpc_version     = 2-4
             env         = LD_LIBRARY_PATH=/etc/securelib
          }

          #
          # Sample unlisted service
          #
          service unlisted
          {
             type         = UNLISTED
             socket_type     = stream
             protocol         = tcp
             wait         = no
             server         = /home/user/some_server
             port         = 20020
          }

SEE ALSO
       xinetd(1L),
       xinetd.log(5)
       Postel J., Echo Protocol, RFC 862, May 1983
       Postel J., Discard Protocol, RFC 863, May 1983
       Postel J., Character Generator Protocol, RFC 864, May 1983
       Postel J., Daytime Protocol, RFC 867, May 1983
       Postel J., Harrenstien K., Time Protocol, RFC 868, May 1983
       M. Lottor, TCP Port Service Multiplexer (TCPMUX), RFC 1078 Nov 1988
       StJohns M.,  Identification Protocol, RFC 1413, February 1993

BUGS
       If  the  INTERCEPT  flag is not used, access control on the address of
       the remote host is not performed when wait is yes and  socket_type  is
       stream.

       The  NOLIBWRAP  flag is automatically turned on for RPC services whose
       socket_type is stream because xinetd cannot determine the  address  of
       the remote host.
```

```
       If  the  INTERCEPT  flag is not used, access control on the address of
       the remote host for services where wait  is  yes     and  socket_type  is
       dgram  is  performed  only  on  the  first packet. The server may then
       accept packets from hosts not in the access  control  list.  This  can
       happen with RPC services.

       There is no way to put a SPACE in an environment variable.

       When  wait  is yes and socket_type is stream, the socket passed to the
       server can only accept connections.

       The INTERCEPT flag is not supported for internal  services  or  multi-
       threaded services.

                 14 June 2001                XINETD.CONF(5)
```