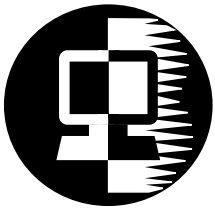
http://www.bintec.de



RELEASE NOTE
BinGO!

December 19, 1997

New System Software:
Release 4.7 Revision 1

This document describes the new features, enhancements, bugfixes, and changes to the BinGO! System Software since Release 4.6 Revision 4.

What's new in Release 4.7.1

Upgrading System Software

1. Retrieve the current system software image from BinTec's HTTP server at http://www.bintec.de.
2. With this image you can upgrade the BinGO! with the **update** command from the SNMP shell via a remote host (i.e. using telnet, minipad, or isdnlogin) or by using the **BOOTmonitor** if you are logged in directly on the console. Information on using the BOOTmonitor can be found in the *BinGO! User's Guide* under *Firmware Upgrades*.
3. Once you've installed Release 4.7 Revision 1 you may want to retrieve the latest documentation (in Adobe's PDF format) which is also available from BinTec's FTP server noted above.

   Note: When upgrading system software, it is also recommended that you use the most current versions of *BRICKware for Windows* and *UNIXTools*. Both can be retrieved from BinTec's FTP server.

What's New in Release 4.7

Release 4.7 Revision 1: Released: 19.12.97

Features

The trace Command

With Release 4.7 Rev. 1 the **trace** command is now officially available on your BinGO!.

For WAN interfaces:

trace [**-h23aFAtpiNxX**] [**-T** ‹*tei*›] [**-c** ‹*cref*›] ‹*channel*› ‹*unit*› ‹*slot*›

For LAN interfaces:

trace [**-h23iNxXl**] [**-d** ‹*MAC filter*›] [**-o**] [**-s** ‹*MAC filter*›] **0 0** ‹*slot*›

The options have the following meaning:

-h hexadecimal output
-2 layer 2 output
-3 layer 3 output
-a asynchronous HDLC (B-Channel only)
-F FAX (B-Channel only)
-A FAX + AT Commands (B-Channel only)
-p PPP (B-Channel only)
-i IP output (B-Channel only)
-N Novell IPX output (B-Channel only)
-t ASCII text output (B-Channel only)
-X asynchronous PPP over X.75 (B-Channel only)
-x raw dump mode
-l set LLC2 filter (LAN only)
-T ‹*tei*›
set tei filter (D-Channel only)
-c ‹*cref*›
set callref filter (D-Channel only)

-d ‹*MAC filter*›
: set destination MAC address filter (LAN only)

-s ‹*MAC filter*›
: set source MAC address filter (LAN only)

-o
: combine two or more -s or -d filters with a logical OR operation

‹*MAC filter*›		**me** = BinGO!'s MAC address **bc** = broadcast packets ‹*MAC address*› (xx:xx:xx:xx:xx:xx)
‹*channel*›		0 = D-Channel or X.21 Interface 1..31 = Bx-Channel
‹*unit*›	0..1	for BinGO! always use unit 0
‹*slot*›	1..7	the slot the module is installed in

The ‹MAC filters› deserve some further explanation. You can combine an -s and a -d filter with a logical AND operation by simply specifying them both (see example *LAN AND filter* below). Now only packets with matching source AND destination address are displayed.

To combine two or more -s or -d filters with a logical OR operation, you specify the first filter, followed by **-o**, then specify the next filter, and so on (see example *LAN OR filter* below).

Examples

ISDN B-Channel

```
trace -h23i 1 0 2
```

PPP Interface

trace -ip ‹*ifcname*›

next used B-Channel

```
trace -ip next
```

LAN AND filter (packets from my BinGO! to the specified MAC address)

```
trace -2iN -s me -d 0:a0:f9:d:5:a 0 0 1
```

LAN OR filter (broadcast packets OR packets from my BinGO!)

```
trace -d bc -o -d me 0 0 1
```

New rtlookup Command

The *rtlookup* (route lookup) command will output the destination interface an IP packet would be routed to.

You can input the destination IP address and several parameters:

rtlookup [**-isuvotp**] ‹*destination IP address*›

-i ‹*source ifindex*›
-s ‹*source IP address*›
-u ‹*source port*›
-v ‹*destination port*›
-o ‹*tos / type of service*›
-t ‹*ttl / time to live*›
-p ‹*protocol*›

Make sure to specify a *source ifindex* if you are testing security features, because otherwise the »packet« will be treated as if it was generated locally on the BinGO!, thus nullifying the effect of most security features, e.g. access lists.

Please note, that the current operating status of the interfaces specified in the rtlookup command will not be affected, e.g. if you issue a rtlookup for a dormant ISDN interface it will correctly be reported to be »not available«.

Examples

```
brick:> rtlookup 123.45.35.34
Matches ipRouteTable, inx = 0
Using ifindex 1000 nexthop 123.45.35.34

brick:> rtlookup -i 1000 1.2.3.4
Denied
```

```
brick:> rtlookup 123.45.35.61
Local destination
```

New system Option for the debug Command

The debug command available from the SNMP shell of your BinGO! can now take the new **system** option.

debug system displays all system debugging information *except* for accounting information.

debug [**show**] | [[**-t**] **all** | **acct**
| **system** | ‹*subs*› [‹*subs*›]]]

Please refer to chapter 7 »Command Reference« of your User's Guide for an explanation of the other options.

Security Feature: ipExtIfBackRtVerify

With the new *ipExtIfBackRtVerify* variable you can filter out all packets which would not be routed back over the same interface they were received on if their source address was used as a destination.

The purpose of this filter is to discard packets with a potentially fake source address.

A syslog message is generated for discarded packets.

INFO/INET: backward route verify failed from if ‹*ifindex*› prot ‹*prot*› ‹*source IP address*› -> ‹*dest. IP address*›

Please note that in cases where packets should take an asymmetric path—i.e. be received via one interface, but transmitted via a different interface—you have to switch *ipExtIfBackRtVerify* **off**, otherwise these packets are also discarded.

This filter can be separately enabled (**on**) for each interface entry in the ***ipExtIfTable***. By default it is switched **off**.

From the Setup Tool you can enable the *Back Route Verify* in the [*WAN Partners*][*EDIT*][*Advanced Settings*] menu.

The IP back route verification can also be used to protect the BinGO! from many »Denial-of-service«-type attacks (see also Bugfixes on page 10).

New ipNatOutTable

This new table can be used to configure Network Address Translation for *outgoing* connections, in effect hiding the internal network addresses from the outside world.

To enable *outgoing* NAT for an interface you have to set the corresponding *NatOutXlat* variable to **on** in the ***ipExtIfTable***.

For example imagine a case where a company used the »free« 10.x.y.z IP addresses for their internal network, and now they want to open the network to the internet. They obviously cannot use the 10.x.y.z addresses, so to avoid having to reconfigure their entire network structure, they use a BinGO! as a gateway to the internet and on the BinGO! configure the ***ipNatOutTable*** to translate each 10.x.y.z address used inside the network to a specific valid IP address for outgoing connections.

If no ***ipNatOutTable*** entry matches the source IP address of an outgoing packet, the IP address of the NAT interface is used as the new source IP address.

Support for new leased line type

With Rel. 4.7 Rev. 1 you can now have leased line bundles where each B channel is connected to a different partner.

Basic Rate Interfaces

In Setup Tool choose the Basic Rate Interface (BRI) you wish to use, and select »*leased line B1+B2 different endpoints*«[1] as the ISDN Switch Type.

1. This type of leased line is called »Digital 64S mit Doppelanschaltung« in Germany.

Or—if you configure your BinGO! directly from the SNMP shell—in the ***isdnChTable*** set the *Type* entries of the BRI you wish to use to **loopback** for the D channel and to **leased_dte** for the B channels. Set *Bundle* to **0** for the B channels. Then in the ***isdnStkTable*** set the *ProtocolProfile* to **not_used** and in the ***isdnIfTable*** set *Autoconfig* to **off**.

This will create two new WAN Partner entries, named **bri**‹*slot*›**-**‹*unit*›**-**‹*B channel*›. You then have to configure these new WAN Partners.

MIB Changes

New ipExtIfBackRtVerify variable

ipExtIfBackRtVerify

Possible values: **off** (1), **on** (2)

This variable activates a check for incoming packets. If set to **on**, incoming packets are only accepted if return packets sent back to their source IP address would be sent over the same interface. This prevents packets being passed from untrusted interfaces to this interface.

Default value: **off**

New ipNatOutTable

This table specifies the IP address translation for outgoing sessions. If no matching entry is found the IP address is set to the IP address defined on the interface configured for NAT. If a matching entry is found, the source IP address of outgoing IP packets is set to the value of *ipNatOutExtAddr*. This table is only used if the outgoing address translation is activated (*ipExtIfNatOutXlat* **on**).

Entries in the table are created and removed manually by network management.

The ***ipNatOutTable*** has consists of the following variables:

ipNatOutIfIndex

This variable specifies the interface index, for which

the table entry shall be valid. If set to 0, the entry will be valid for all interfaces configured to use NAT.

ipNatOutProtocol

Possible values: **icmp** (1), **tcp** (6), **udp** (17), **any** (255), **delete** (256)

This variable specifies the protocol, for which the table entry shall be valid.

Default value: **any**

ipNatOutRemoteAddr

Together with *ipNatOutRemoteMask* this variable specifies the set of target IP addresses for which the table entry is valid. If both variables are set to **0.0.0.0**, the table entry will be valid for any target IP address.

ipNatOutRemoteMask

Together with *ipNatOutRemoteAddr* this variable specifies the set of target IP addresses for which the table entry is valid. If both variables are set to **0.0.0.0**, the table entry will be valid for any target IP address.

ipNatOutExtAddr

This variable specifies the external IP address to which the internal IP address is mapped.

ipNatOutRemotePort

Possible values: **–1**..**65535**

Together with *ipNatOutRemotePortRange* this variable specifies the range of portnumbers for outgoing calls, for which the table entry shall be valid. If both variables are set to **-1**, the entry is valid for all portnumbers. If *ipNatOutPortRange* is set to **-1**, the entry is only valid, when the portnumber of an outgoing call is equal to *ipNatOutRemotePort*. Otherwise, the entry is valid, if the called portnumber is in the range *RemotePort* .. *RemotePortRange*.

Default value: **-1**

ipNatOutRemotePortRange

Possible values: **–1**..**65535**)

Together with *ipNatOutRemotePort* this variable specifies the range of portnumbers for outgoing calls, for which the table entry shall be valid. If both variables are set to **–1**, the entry is valid for all portnumbers. If *ipNatOutPortRange* is set to **-1**, the entry is only valid, when the portnumber of an outgoing call is equal to *ipNatOutRemotePort*. Otherwise, the entry is valid, if the called portnumber is in the range *RemotePort* .. *RemotePortRange*.
Default value: **-1**

ipNatOutIntAddr

Together with *ipNatOutIntMask* this variable specifies the internal host's IP address for outgoing calls matching the table entry. If both variables are set to **0.0.0.0**, the table entry will be valid for any source IP address.

ipNatOutIntMask

Together with *ipNatOutIntAddr* this variable specifies the internal host's IP address for outgoing calls matching the table entry. If both variables are set to **0.0.0.0**, the table entry will be valid for any source IP address.

Bugfixes

CAPI

- The CAPI2_INFO_IND messages for channel identification now contain the correct setting of the Info Number field (0x18).
- When an ISDN trace was mistakenly started on the CAPI TCP port, this led to a system boot.
  This bug has been fixed.

Setup Tool

- When setting a NAT port ([*IP*][*Network Address Translation*][*Config*][*EDIT*]) to **-1**, this value was mistakenly

changed to 65535.
This bug has been fixed.

- When trying to delete an entry from a list in Setup Tool in rare cases a wrong entry was deleted.
  This bug has also been fixed.

Security

- The BinGO! is no longer vulnerable to the »LAND« type »Denial-of-service« attacks via TCP. This type of attack involved sending a TCP packet with identical source and destination IP addresses and a set SYN flag to the BinGO!.