



Manual do usuário

**Protocolo CSTA para a linha
de centrais UnniTI**



Protocolo CSTA para a linha de centrais UnniTI

Parabéns, você acaba de adquirir um produto com a qualidade e segurança Intelbras.

Este documento visa mostrar as particularidades da implementação do protocolo CSTA fase pela INTELBRAS nas centrais UnniTI, comandos e eventos implementados, comandos e eventos específicos implementados e programações necessárias.

Índice

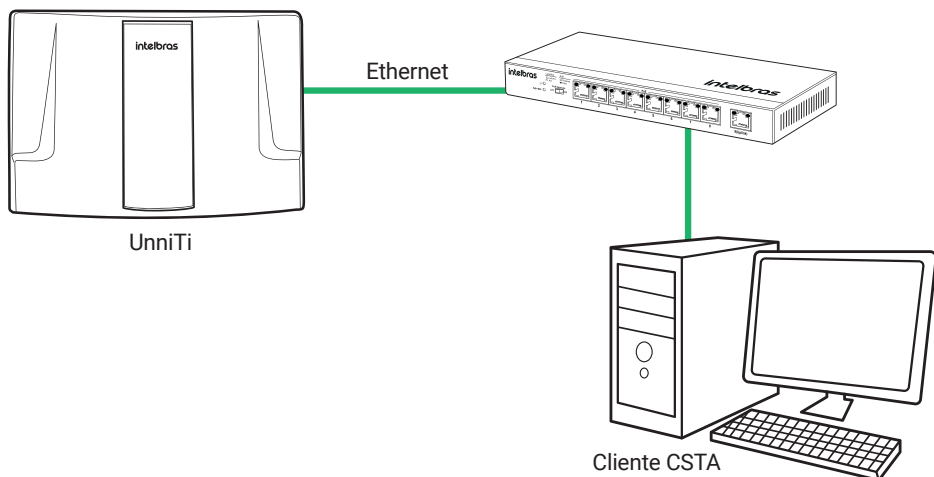
1. Informações	5
1.1. ICTI	5
1.2. Chave de registro	5
1.3. Programação do protocolo CSTA	5
1.4. Compatibilidade	5
1.4.1. Pabx	5
1.4.2. Versões	5
2. Comunicação com o ICTI	6
2.1. Porta	6
2.2. Formato do pacote de dados TCP	6
2.2.1. Formato	6
2.2.2. Exemplo	6
2.3. Autenticação	6
2.3.1. Formato ACSE	6
2.3.1.1. Tipos de pacotes ACSE	6
2.3.1.2. Norma ACSE- 1	7
2.3.2. Campos Utilizados	14
2.3.2.1. Pacote AARE	16
2.3.2.2. Pacote RLRQ	17
2.3.2.3. Pacote RLRE	17
2.3.2.4. Pacote ABRT	17
2.3.3. Funcionamento	17
3. Comandos e eventos CSTA implementados	17
3.1. Switching function services	17
3.1.1. Implementados parcialmente	18
3.2. Switching function events	18
3.2.1. Call events	18
3.2.2. Agent events	18
3.2.3. Other feature events	18
3.2.4. Maintenance events	18
3.2.5. Private events	18
3.2.6. Voice unit events	18
3.3. Computing function services	19
3.4. Bidirectional services	19
3.5. Status reporting services	19
3.6. Input/output services	19
3.7. Voice unit services	19

4. Comandos e Eventos Privados para UnniTI	20
4.1. Comandos Escape	20
4.2. Formato padrão CSTA	20
4.2.1. Comando escape	20
4.2.2. CommonArguments para o Result	20
4.2.3. Definição do private data	21
4.3. Solicitação de dados dos dispositivos da central.	22
4.3.1. Solicitação	22
4.3.2. Resposta	23
4.4. Solicitação ou definição do turno	25
4.4.1. Solicitação/Definição	25
4.4.2. Resposta	27
5. Protocolo CSTA	28
5.1. Normas	28
5.2. Nome de dispositivos.	28
5.2.1. Descrição dos dispositivos utilizados pelo CSTA da linha UnniTI	28
5.3. Restrições	29
5.3.1. Comandos	29
5.3.2. Eventos	29
5.3.3. Operação do ramal analógico	29
5.4. Características de Comandos CSTA da UnniTI	29
5.4.1. Makecall	29
5.4.1.1. Possibilidades	29
5.4.1.2. Exemplos de comandos makecall.	30
5.4.2. AnswerCall	32
5.4.2.1. Possibilidades	32
5.4.2.2. Exemplos do comando AnswerCall.	33
5.4.3. ClearConnection	34
5.4.3.1. Possibilidades	34
5.4.4. SendDtmf	34
5.4.4.1. Possibilidades	34
5.4.4.2. Exemplo de comando SendDtmf	35
5.5. Exemplos de cenários	35

1. Informações

1.1. ICTI

O ICTI (Interface Computador Telefone Intelbras) já está disponível na central UnniTI, não necessitando de nenhuma instalação adicional.



1.2. Chave de registro

É necessária a existência de uma licença para que seja possível utilizar o CSTA da UnniTI. A chave de hardware que acompanha todas as versões da central UnniTI, já possui a licenças CSTA habilitada.

1.3. Programação do protocolo CSTA

A central UnniTI já está com o CSTA habilitado por padrão, não existindo nenhuma configuração adicional a ser feita.

1.4. Compatibilidade

1.4.1. Pabx

O serviço de CSTA está disponível para os seguintes modelos da linha UnniTI:

- » UnniTI 1000
- » UnniTI 2000
- » UnniTI 3000

1.4.2. Versões

Todas as versões das centrais UnniTI são compatíveis com o protocolo CSTA.

2. Comunicação com o ICTI

2.1. Porta

O cliente CSTA deve-se conectar à porta 7001 da central UnniTI.

Esta conexão deve ser do tipo TCP/IP.

2.2. Formato do pacote de dados TCP

Após o pacote de autenticação ou CSTA ter sido codificado, é necessário incluir um cabeçalho a este pacote.

2.2.1. Formato

Cada pacote é enviado ou recebido no seguinte formato:

SS ss DD DD DD D..D DD DD

- » **SS ss** : tamanho dos dados enviados ou recebidos, sendo SS o byte mais significativo e ss o byte menos significativo
- » **DD..** : dados CSTA ou ACSE.

2.2.2. Exemplo:

00 2C A1 2A 02 01 02 02 01 47 30 22 80 03 2A 31 33 30 18 80 03 02 00 01 81 02 00 00 82 02 02 01 83 02 06 0D 85 02 01 01 84 01 FF 0A 01 01

- » **Tamanho:** 0x002c = 44 bytes
- » **Dados:** A1 2A 02 01 02 02 01 47 30 22 80 03 2A 31 33 30 18 80 03 02 00 01 81 02 00 00 82 02 02 01 83 02 06 0D 85 02 01 01 84 01 FF 0A 01 01

2.3. Autenticação

Para que o cliente possa receber e enviar dados CSTA, é necessário o envio de uma mensagem de autenticação. A autenticação é feita utilizando o protocolo ACSE, onde são passados o id da aplicação, o usuário e a senha.

2.3.1. Formato ACSE

Utiliza-se o padrão ACSE – 1 definido na norma X.227:04/1995.

2.3.1.1. Tipos de pacotes ACSE

- » **AARQ:** solicitação de conexão. Enviado pelo cliente.
- » **AARE:** resposta do AARQ. Enviado pelo servidor
- » **RLRQ:** solicitação de desconexão. Enviado pelo cliente
- » **RLRE:** resposta de RLRQ. Enviado pelo servidor
- » **ABRT:** finalização forçada da conexão. Enviado pelo servidor ou pelo cliente. Não necessita de resposta.

2.3.1.2. Norma ACSE-1

Transcrição direta da norma.

– Module ACSE-1 (X.227:04/1995)

– See also the index of all ASN.1 assignments needed in this document

ACSE-1 {joint-iso-itu-t association-control(2) modules(0) apdus(0) version1(1)}

– ACSE-1 refers to ACSE version 1

DEFINITIONS ::=

BEGIN

EXPORTS

acse-as-id, ACSE-apdu, aCSE-id, Application-context-name, AP-title,
AE-qualifier, AE-title, AP-invocation-identifier, AE-invocation-identifier,
Mechanism-name, Authentication-value, ACSE-requirements, ObjectSet;

IMPORTS

Name, RelativeDistinguishedName

FROM InformationFramework {joint-iso-itu-t ds(5) module(1)

informationFramework(1) 3};

– The data types Name and RelativeDistinguishedName are imported from ISO/IEC 9594-2.

– object identifier assignments

acse-as-id OBJECT IDENTIFIER ::=

{joint-iso-itu-t association-control(2) abstract-syntax(1) apdus(0)
version1(1)}

– may be used to reference the abstract syntax of the ACSE APDUs

aCSE-id OBJECT IDENTIFIER ::=

{joint-iso-itu-t association-control(2) ase-id(3) acse-ase(1) version(1)}

– may be used to identify the Association Control ASE.

– top level CHOICE

ACSE-apdu ::= CHOICE {

aarq AARQ-apdu,

aare AARE-apdu,

rlrq RLRQ-apdu,

rlre RLRE-apdu,

abrt ABRT-apdu,

...

}

```

AARQ-apdu ::= [APPLICATION 0] IMPLICIT SEQUENCE {
    protocol-version
        [0] IMPLICIT BIT STRING {version1(0)} DEFAULT {version1},
    application-context-name      [1] Application-context-name,
    called-AP-title               [2] AP-title OPTIONAL,
    called-AE-qualifier           [3] AE-qualifier OPTIONAL,
    called-AP-invocation-identifier [4] AP-invocation-identifier OPTIONAL,
    called-AE-invocation-identifier [5] AE-invocation-identifier OPTIONAL,
    calling-AP-title              [6] AP-title OPTIONAL,
    calling-AE-qualifier           [7] AE-qualifier OPTIONAL,
    calling-AP-invocation-identifier [8] AP-invocation-identifier OPTIONAL,
    calling-AE-invocation-identifier [9] AE-invocation-identifier OPTIONAL,
    -- The following field shall not be present if only the Kernel is used.
    sender-acse-requirements      [10] IMPLICIT ACSE-requirements OPTIONAL,
    -- The following field shall only be present if the Authentication functional unit is selected.
    mechanism-name                [11] IMPLICIT Mechanism-name OPTIONAL,
    -- The following field shall only be present if the Authentication functional unit is selected.
    calling-authentication-value   [12] EXPLICIT Authentication-value OPTIONAL,
    application-context-name-list
        [13] IMPLICIT Application-context-name-list OPTIONAL,
    -- The above field shall only be present if the Application Context Negotiation functional unit
    is selected
    implementation-information     [29] IMPLICIT Implementation-data OPTIONAL,
    ...,
    ...,
    user-information
        [30] IMPLICIT Association-information OPTIONAL
}

```

```

AARE-apdu ::= [APPLICATION 1] IMPLICIT SEQUENCE {
    protocol-version
        [0] IMPLICIT BIT STRING {version1(0)} DEFAULT {version1},
    application-context-name      [1] Application-context-name,
    result                        [2] Associate-result,
    result-source-diagnostic      [3] Associate-source-diagnostic,
    responding-AP-title           [4] AP-title OPTIONAL,
    responding-AE-qualifier       [5] AE-qualifier OPTIONAL,
    responding-AP-invocation-identifier [6] AP-invocation-identifier OPTIONAL,
    responding-AE-invocation-identifier [7] AE-invocation-identifier OPTIONAL,
    -- The following field shall not be present if only the Kernel is used.
    responder-acse-requirements   [8] IMPLICIT ACSE-requirements OPTIONAL,
    -- The following field shall only be present if the Authentication functional unit is selected.

```



```

mechanism-name          [9] IMPLICIT Mechanism-name OPTIONAL,
-- This following field shall only be present if the Authentication functional unit is selected.
responding-authentication-value
  [10] EXPLICIT Authentication-value OPTIONAL,
application-context-name-list
  [11] IMPLICIT Application-context-name-list OPTIONAL,
-- The above field shall only be present if the Application Context Negotiation functional unit
is selected
implementation-information
  [29] IMPLICIT Implementation-data OPTIONAL,
...,
...,
user-information
  [30] IMPLICIT Association-information OPTIONAL
}

```

```

RLRQ-apdu ::= [APPLICATION 2] IMPLICIT SEQUENCE {
  reason          [0] IMPLICIT Release-request-reason OPTIONAL,
  ...,
  ...,
  user-information [30] IMPLICIT Association-information OPTIONAL
}

```

```

RLRE-apdu ::= [APPLICATION 3] IMPLICIT SEQUENCE {
  reason          [0] IMPLICIT Release-response-reason OPTIONAL,
  ...,
  ...,
  user-information [30] IMPLICIT Association-information OPTIONAL
}

```

```

ABRT-apdu ::= [APPLICATION 4] IMPLICIT SEQUENCE {
  abort-source     [0] IMPLICIT ABRT-source,
  abort-diagnostic [1] IMPLICIT ABRT-diagnostic OPTIONAL,
-- This field shall not be present if only the Kernel is used.
  ...,
  ...,
  user-information [30] IMPLICIT Association-information OPTIONAL
}

```

ABRT-diagnostic ::= ENUMERATED {
 no-reason-given(1), protocol-error(2),
 authentication-mechanism-name-not-recognized(3),
 authentication-mechanism-name-required(4), authentication-failure(5),
 authentication-required(6), ...
}

ABRT-source ::= INTEGER {acse-service-user(0), acse-service-provider(1)
 }(0..1, ...)

ACSE-requirements ::= BIT STRING {
 authentication(0), application-context-negotiation(1)}

Application-context-name-list ::= SEQUENCE OF Application-context-name

Application-context-name ::= OBJECT IDENTIFIER

-- Application-entity title productions follow (not in alphabetical order)

AP-title ::= CHOICE {
 ap-title-form1 AP-title-form1,
 ap-title-form2 AP-title-form2,
 ...
}

AE-qualifier ::= CHOICE {
 ae-qualifier-form1 AE-qualifier-form1,
 ae-qualifier-form2 AE-qualifier-form2,
 ...
}

-- When both AP-title and AE-qualifier data values are present in an AARQ or AARE APDU, both must

-- have the same form to allow the construction of an AE-title as discussed in CCITT Rec. X.665
 |

-- ISO/IEC 9834-6.

AP-title-form1 ::= Name

-- The value assigned to AP-title-form1 is The Directory Name of an application-process title.

AE-qualifier-form1 ::= RelativeDistinguishedName

- The value assigned to AE-qualifier-form1 is the relative distinguished name of a particular application-entity of the application-process identified by AP-title-form1.

AP-title-form2 ::= OBJECT IDENTIFIER

AE-qualifier-form2 ::= INTEGER

```
AE-title ::= CHOICE {
    ae-title-form1 AE-title-form1,
    ae-title-form2 AE-title-form2,
    ...
}
```

- As defined in CCITT Rec. X.650 | ISO 7498-3, an application-entity title is composed of an application-
 - process title and an application-entity qualifier. The ACSE protocol provides for the transfer of an
 - application-entity title value by the transfer of its component values. However, the following data type
 - is provided for International Standards that reference a single syntactic structure for AE titles.

AE-title-form1 ::=

Name

- For access to The Directory (ITU-T Rec. X.500-Series | ISO/IEC 9594), an AE title has AE-title-form1.
- This value can be constructed from AP-title-form1 and AE-qualifier-form1 values contained in an
- AARQ or AARE APDU. A discussion of forming an AE-title-form1 from AP-title-form1 and AE-qualifier-
- form1 may be found in CCITT Rec. X.665 | ISO/IEC 9834-6.

AE-title-form2 ::= OBJECT IDENTIFIER

- A discussion of forming an AE-title-form2 from AP-title-form2 and AE-qualifier-form2 may be
- found in CCITT Rec. X.665 | ISO/IEC 9834-6.

AE-invocation-identifier ::= INTEGER

AP-invocation-identifier ::= INTEGER

- End of Application-entity title productions

Associate-result ::= INTEGER {

accepted(0), rejected-permanent(1), rejected-transient(2)}(0..2, ...)

```

Associate-source-diagnostic ::= CHOICE {
  acse-service-user
    [1] INTEGER {null(0), no-reason-given(1),
      application-context-name-not-supported(2),
      calling-AP-title-not-recognized(3),
      calling-AP-invocation-identifier-not-recognized(4),
      calling-AE-qualifier-not-recognized(5),
      calling-AE-invocation-identifier-not-recognized(6),
      called-AP-title-not-recognized(7),
      called-AP-invocation-identifier-not-recognized(8),
      called-AE-qualifier-not-recognized(9),
      called-AE-invocation-identifier-not-recognized(10),
      authentication-mechanism-name-not-recognized(11),
      authentication-mechanism-name-required(12),
      authentication-failure(13), authentication-required(14)}
      (0..14, ...),
  acse-service-provider
    [2] INTEGER {null(0), no-reason-given(1), no-common-acse-version(2)}
      (0..2, ...)
}

```

Association-information ::= SEQUENCE SIZE (1, ..., 0 | 2..MAX) OF EXTERNAL

```

Authentication-value ::= CHOICE {
  charstring [0] IMPLICIT GraphicString,
  bitstring [1] IMPLICIT BIT STRING,
  external [2] IMPLICIT EXTERNAL,
  other
    [3] IMPLICIT SEQUENCE {other-mechanism-name
      MECHANISM-NAME.&id({ObjectSet}),
      other-mechanism-value
      MECHANISM-NAME.&Type
      ({ObjectSet}){@.other-mechanism-name}}
}

```

- The abstract syntax of (calling/responding) authentication-value is determined by the authentication
 - mechanism used during association establishment. The authentication mechanism is either explicitly
 - denoted by the &id field (of type OBJECT IDENTIFIER) for a mechanism belonging to the class
 - MECHANISM-NAME, or it is known implicitly by
 - prior agreement between the communicating partners. If the "other" component is chosen, then
 - the "mechanism-name" component must be present in accordance with
 - ITU-T Rec. X.680 | ISO/IEC 8824. If the value "mechanism-name" occurs in the AARQ-apdu or the
 - AARE-apdu, then that value must be the same as the value for "other-mechanism-name"
- Implementation-data ::= GraphicString

Mechanism-name ::= OBJECT IDENTIFIER

MECHANISM-NAME ::= TYPE-IDENTIFIER

ObjectSet MECHANISM-NAME ::=
 {...}

Release-request-reason ::= INTEGER {normal(0), urgent(1), user-defined(30)}
 (0 | 1 | 30, ...)

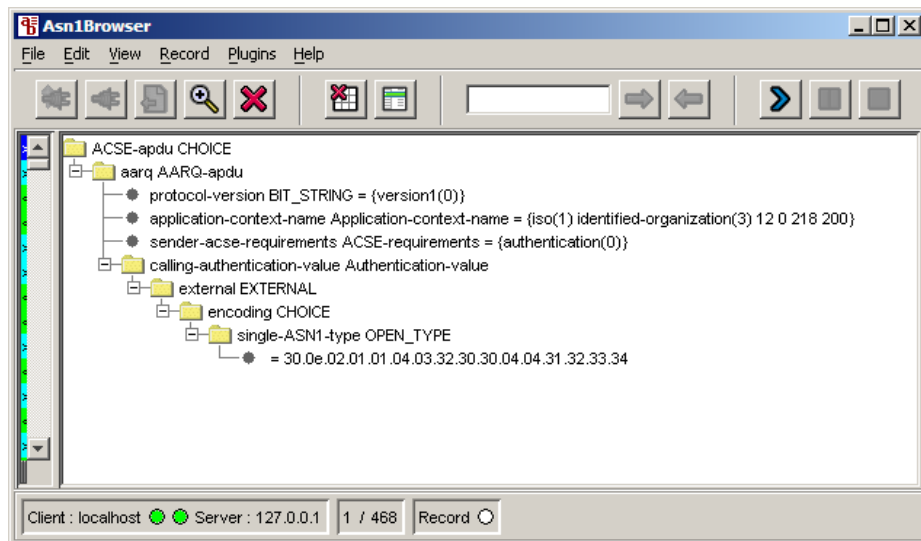
Release-response-reason ::= INTEGER {
 normal(0), not-finished(1), user-defined(30)}(0 | 1 | 30, ...)

END

Pacote AARQ

Este pacote contém as informações de autenticação e deve ser a primeira mensagem enviada pelo cliente ao servidor.

O cliente deverá aguardar a resposta AARE.



Formato do pacote AARQ

2.3.2. Campos Utilizados

» **protocol-version**

Fixo.

Definir o campo "version1" como "true"

» **application-context-name**

Fixo.

Definir como 1.3.12.0.218.200

» **sender-acse-requirements**

Fixo.

Definir os subcampos como:

authentication = true

application-context-negotiation = false

» **calling-authentication-value**

O campo deve ser preenchido com a estrutura Authentication-value.

Authentication-value ::= CHOICE

```
{
  external [2] IMPLICIT PrivateExtData
}
```

PrivateExtData ::= [UNIVERSAL 8] IMPLICIT

```
{
  encoding AsnData
}
```

```
AsnData ::= ANY
{
single-ASN1-type [0] External-authentication
}
```

```
External-authentication ::= SEQUENCE
{
ApplId INTEGER,
UserName OCTECT STRING OPTIONAL
Password OCTECT STRING OPTIONAL
AdditionalData OCTECT STRING OPTIONAL
}
```

» **ApplId**

É o identificador da aplicação e deve possuir o valor 1 (um) sempre.

» **UserName**

Nome do usuário.

É uma string.

Exemplo: 2000 ; pedro

» **Password**

Senha utilizada em conjunto com UserName.

É uma string.

Não está sendo utilizada e deve ter o valor de 1234 obrigatoriamente.

» **AdditionalData**

Não utilizado.

Observação

O usuário e senha não estão sendo validados, mas são utilizados para realizar “logs” da comunicação.

Formato do pacote

```
60 S1 80 02 07 80 a1 09 06 07 2b 0c 00 81 5a 81 48 8a 02 06 80 ac 15 a2 13 a0 11 30 S2 02 01 01 04
S3 UU UU UU UU 04 04 31 32 33 34
```

» UU : Usuário em ASCII

» S3 : Número de bytes do usuário.

» S2 : Número de bytes até o final do pacote = 11+ S3

» S1 : Número de bytes até o final do pacote = 27 +S2

Exemplo de pacote

Obs.: não está incluso o header de tamanho do tcp.

UserName = 200 . Em azul.

Password = 1234 . Em verde.

```
60 29 80 02 07 80 a1 09 06 07 2b 0c 00 81 5a 81 48 8a 02 06 80 ac 14 a2 12 a0 10 30 0e 02
01 01 04 03 32 30 30 04 04 31 32 33 34
```

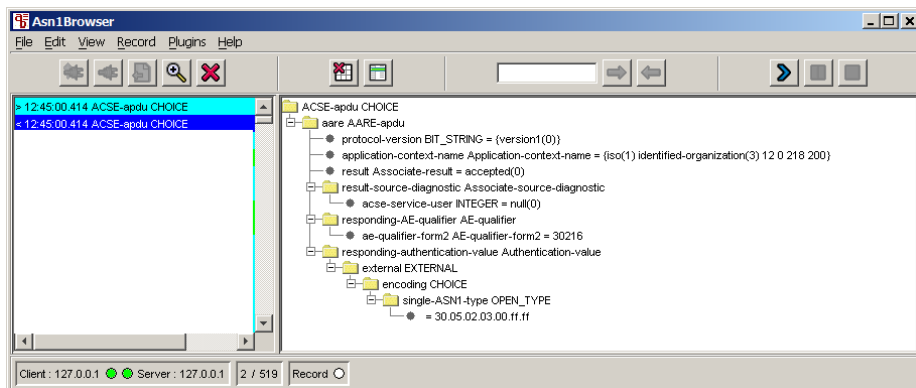
UserName = 2000. Em azul.

Password = 1234 . Em verde.

```
60 2a 80 02 07 80 a1 09 06 07 2b 0c 00 81 5a 81 48 8a 02 06 80 ac 15 a2 13 a0 11 30 0f 02
01 01 04 04 32 30 30 04 04 31 32 33 34
```

2.3.2.1. Pacote AARE

Este pacote contém as informações de resposta ao pacote AARQ.



Pacote AARE

Campos Utilizados

» **protocol-version**

Sempre version1.

» **application-context-name**

Sempre com 1.3.12.0.218.200

» **result e result-source-diagnostic**

Se a conexão foi aceita, result contém accepted(0) e result-source-diagnostic contém null(0).

Se a conexão não foi aceita devido a senha/usuário inválidos ou porque não existem licenças, result contém rejected-permanent(1) ou rejected-transient(2) e result-source-diagnostic contém no-reason-givn(1) ou authentication-failure(13)

» **responding-AE-qualifier**

Utilizado para retornar a versão do ICTI. É um número inteiro que representa a versão. No exemplo a versão é 30216, ou seja, 3.02.16.

Este valor está disponível apenas se o resultado da autenticação for positivo.

Formato do pacote

61 2e 80 02 07 85 a1 09 06 07 2b 0c 00 81 5a 81 48 82 03 02 01 RR 83 05

81 03 02 01 RD 85 04 02 02 VV VV aa 0b a2 09 a0 07 30 05 02 03 00 ff ff

RR : resposta da conexão, pode ser 0, 1 ou 2 conforme descrito em result .

RD: complemento da resposta.

VV VV: versão do ICTI.

O restante do pacote não se altera.

Exemplo de pacote

Obs.: não está incluso o header de tamanho do TCP.

61 2e 80 02 07 85 a1 09 06 07 2b 0c 00 81 5a 81 48 82 03 02 01 00 83 05

81 03 02 01 00 85 04 02 02 76 08 aa 0b a2 09 a0 07 30 05 02 03 00 ff ff

2.3.2.2. Pacote RLRQ

Não utilizado.

2.3.2.3. Pacote RLRE

Não utilizado.

2.3.2.4. Pacote ABRT

Enviado para indicar uma desconexão antes de fechar a conexão TCP.

É opcional para o cliente, não sendo necessário o envio antes da desconexão.

O Icti envia este pacote como resposta ao login quando A central não terminou a inicialização.

Exemplo de pacote

Obs.: não está incluso o header de tamanho do TCP.

64 03 80 01 00

2.3.3. Funcionamento

O cliente CSTA deve enviar um pacote AARQ com o usuário preenchido e deve aguardar o retorno do pacote AARE. Deve-se, então, verificar a resposta da autenticação, caso seja accepted(0), pode-se iniciar o envio dos pacotes CSTA. Caso contrário, a central fechará a conexão TCP.

Caso o pacote AARQ não seja enviado e um pacote CSTA seja enviado, a central não analisa este pacote.

3. Comandos e eventos CSTA implementados

Os itens riscados não estão implementados.

3.1. Switching function services

Alternate call

Answer call

Associate data

~~Call completion~~

~~Clear call~~

Clear connection

Conference call

Consultation call

Divert call

Hold call

Make call

Make predictive call

Park call

Query device

Reconnect call

Retrieve call

Send DTMF tones

~~Single step conference~~

~~Single step transfer~~

Transfer call

3.1.1. Implementados parcialmente

» **Set feature:** implementados os comandos para controlar agentes.

3.2. Switching function events

3.2.1. Call events

~~Call cleared~~
Conferenced
Connection cleared
Delivered
Diverted
Established
Failed
Held
Network reached
Originated
Queued
Retrieved
Service initiated
Transferred

3.2.2. Agent events

Agent logged on
Agent logged off
Agent ready
Agent not ready
Agent working after call
Agent busy

3.2.3. Other feature events

~~Call information~~
~~Do not disturb~~
~~Forwarding~~
~~Message waiting~~
~~Auto answer~~
~~Microphone mute~~
~~Speaker mute~~
~~Speaker volume~~

3.2.4. Maintenance events

Back in service
Out of service

3.2.5. Private events

Private

3.2.6. Voice unit events

~~Play~~
~~Record~~
~~Review~~
~~Stop~~
~~Suspend play~~
~~Suspend record~~
~~Voice attribute change~~

3.3. Computing function services

- Route request
- Reroute request
- Route select request
- Route used request
- Route end request

3.4. Bidirectional services

- Escape
- System status

3.5. Status reporting services

- Change monitor filter
- Monitor start
- Monitor stop
- Snapshot call
- Snapshot device

3.6. Input/output services

- Data path resumed
- Data path suspended
- Fast data
- Resume data path
- Send broadcast data
- Send data
- Send multicast data
- Start data path
- Stop data path
- Suspend data path

3.7. Voice unit services

- Concatenate message
- Delete message
- Play message
- Query voice attribute
- Record message
- Reposition
- Resume
- Review
- Set voice attribute
- Stop
- Suspend
- Synthesize message

4. Comandos e Eventos Privados para UnniTI

4.1. Comandos Escape

Utiliza a construção normal do *CSTA* para comando *ESCAPE*.

Ocorre a definição da propriedade *CSTAPrivateData* que é formado por um identificador único, que deve ser preenchido no envio do comando *escape* e está preenchido no recebimento do *result* do comando *escape*, e um dado específico da Intelbras.

Este dado depende do contexto e é utilizado para a solicitação e na resposta de um comando. Utiliza um padrão de construção com tipos primitivos conforme o protocolo ASN.1.

4.2. Formato padrão CSTA

Notação utilizada conforme normas do CSTA.

4.2.1. Comando escape

escapeService OPERATION

ARGUMENT EscapeServiceArgument

RESULT EscapeServiceResult

ERRORS {universalFailure}

::= 51

EscapeServiceArgument ::=

SEQUENCE

{

security [0] IMPLICIT CSTASecurityData OPTIONAL,

privateData [1] IMPLICIT SEQUENCE OF CSTAPrivateData

}

EscapeServiceResult ::=

CHOICE

{

extensions CSTACommonArguments,

noData NULL

}

4.2.2. CommonArguments para o Result

CSTACommonArguments ::= [APPLICATION 30] IMPLICIT SEQUENCE

{

security [0] IMPLICIT CSTASecurityData OPTIONAL,

privateData [1] IMPLICIT SEQUENCE OF CSTAPrivateData OPTIONAL

}

4.2.3. Definição do private data

```
CSTAPrivateData ::= SEQUENCE
{
  intelbrasId OBJECT IDENTIFIER,
  intelbrasData [8] IntelbrasData
}
```

IntelbrasId := iso.3.6.1.4.1.26138 (1.3.6.1.4.1.26138) – FIXO

```
IntelbrasData ::= CHOICE
{
  setFeature [0] SetIntelbrasFeature ,
  response [1] IntelbrasResponseData
}
```

```
SetIntelbrasFeature ::= CHOICE
{
  ExistingDevices [0] IMPLICIT NULL ,
  turno [1] Implicit INTEGER
  EnvioDeSms[2] Implicit SmsData}
```

```
IntelbrasResponseData ::= CHOICE
{
  dadosDevice [0] IMPLICIT SEQUENCE OF DadosDeviceIntelbras,
  turnoAtual [1] IMPLICIT INTEGER,
  EnvioDeSms[2] IMPLICIT INTEGER
}
```

```
DadosDeviceIntelbras ::= SEQUENCE
{
  deviceType DeviceTypeIntelbras,
  deviceName IA5String,
  active BOOLEAN,
  deviceId Integer
}
```

```
DeviceTypeIntelbras ::= ENUMERATED
{
  Ramal (0),
  TroncoAnalógico (1),
  TroncoDigital (2),
  Porteiro (3),
  Correio (4),
  Estacionamento (5),
  TroncoVoip (6),
  RamalDigital (7),
  Grupo (8),
```

```

Modem (10),
Ramallp (11),
PedidoChamada ( 12),
JuntorGsm (13),
Disa (14),
GrupoCorreio (15),
BuscaPessoa (16),
Emergencia (17),
Generico (254),
}

```

```

SmsData ::= SEQUENCE
{
    DestinationDevice IA5String, //Número de celular que receberá a mensagem.
    SourceDevice IA5String, //Rama que envia a mensagem
    Message IA5String //mensagem com até 160 caracteres
}

```

4.3. Solicitação de dados dos dispositivos da central

Utilizado para saber quais os ramais, troncos e outros dispositivos que estão disponíveis na central para serem utilizados.

4.3.1. Solicitação

The screenshot shows a network protocol analyzer interface. On the left, a list of captured packets is displayed, with the selected packet being an ACSE-APDU CHOICE. The main pane shows the raw packet data in hexadecimal and ASCII, along with a hierarchical tree view of the packet structure. The tree view indicates a ROS CHOICE containing an invoke SEQUENCE, which includes an invokeId, an opcode Code, an argument OPEN_TYPE, an EscapeServiceArgument, and a privateData SEQUENCE_OF. The privateData SEQUENCE_OF contains a CSTA PrivateData structure with a specific value.

Comando para retornar os dispositivos existentes.

Deve ser executado o comando ESCAPE com a opção intelbrasData preenchida com o dado setFeature definida como ExistingDevices.

Exemplo do pacote Csta gerado:

A1 1C 02 01 02 02 01 33 30 14 A1 12 30 10 06 08 2B 06 01 04 01 81 CC 1A A8 04 A0 02 80 00

O pacote parcial A1 1C 02 01 02 02 01 33 30 14 A1 12 é padrão CSTA e o restante são os dados específicos da Intelbras.

30 10

06 08 2B 06 01 04 01 81 CC 1A Object identifier

A8 04 IntelbrasData

A0 02 SetFeatureIntelbras

80 00 ExistingDevices

O dado em verde é o valor para ExistingDevices.

4.3.2. Resposta

13:49:31.806 ACSE-apdu CHOICE

13:49:31.806 ACSE-apdu CHOICE

13:49:32.367 EscapeServiceArgument

13:49:33.132 EscapeServiceResult

000000: a2 82 11 86 02 01 01 30 82 11 7f 02 01 33 7e 82 |.....0....3~.|

000016: 11 78 a1 82 11 74 30 82 11 70 06 08 2b 06 01 04 |..x...t0...p...+...|

000032: 01 81 cc 1a a8 82 11 62 a1 82 11 5e a0 82 11 5a ||.....b...^....Z|

000048: 30 0e 0a 01 0e 16 03 23 36 39 01 01 ff 02 01 0d |0.....#69.....|

000064: 30 0f 0a 01 0f 16 03 23 38 37 01 01 ff 02 02 1f |0.....#87.....|

000080: 6c 30 11 0a 01 10 16 05 2a 31 39 30 32 01 01 ff |10.....*1902...|

000096: 02 02 07 6c 30 11 0a 01 0a 16 05 2a 31 39 30 35 |...m0.....*1905|

ROS CHOICE

returnResult SEQUENCE

invokeId InvokeId

present INTEGER = 1

result SEQUENCE

opcode Code

local INTEGER = 51

result OPEN_TYPE

EscapeServiceResult

extensions CSTACommonArguments

privateData SEQUENCE_OF

CSTAPrivateData

= 30 82 11 70 06 08 2b 06 01 04 01 81 cc 1a a8 82 11 62 a1 82 11 5e a0 82 11 5a 30 0e 0a 01 0e 16 03 2

Resposta com os dispositivos existentes.

Como resposta, é enviado no result do comando Escape, o dado IntelbrasResponseData com o dado dadosDevice. Este dado é composto de um conjunto de dados de cada dispositivo existente, contendo o nome do dispositivo, Ex.: 201, o tipo do dispositivo (DeviceTypeIntelbras), se o dispositivo está ou não ativo e um identificador interno.

Dispositivo ativo significa que o dispositivo está funcionando corretamente.

Dispositivo não ativo significa que o dispositivo foi definido como existente na central, mas não existe hardware para torná-lo funcional ou não está operando, como o caso de um ramal digital que está desconectado.

Exemplo do pacote Csta gerado:

a2 82 04 90 02 01 01 30 82 04 89 02 01 33 7e 82 04 82 a1 82 04 7e 30 82 04 7a 06 08 2b 06 01 04
01 81 cc 1a a8 82 04 6c a1 82 04 68 a0 82 04 64 30 0f 0a 02 00 fe 16 03 23 36 39 01 01 ff 02 01
0d 30 10 0a 02 00 fe 16 03 23 38 37 01 01 ff 02 02 1f 6c 30 12 0a 02 00 fe 16 05 2a 31 39 30 30
01 01 ff 02 02 07 6c 30 12 0a 02 00 fe 16 05 2a 31 39 30 34 01 01 ff 02 02 07 70 30 12 0a 02 00
fe 16 05 2a 31 39 30 35 01 01 ff 02 02 07 71 30 11 0a 01 03 16 05 2a 31 39 30 36 01 01 ff 02 02
07 72 30 11 0a 01 04 16 05 2a 31 39 33 30 01 01 ff 02 02 07 8a 30 11 0a 01 04 16 05 2a 31 39 33
31 01 01 ff 02 02 07 8b 30 11 0a 01 04 16 05 2a 31 39 33 32 01 01 ff 02 02 07 8c 30 11 0a 01 04
16 05 2a 31 39 33 33 01 01 ff 02 02 07 8d 30 11 0a 01 04 16 05 2a 31 39 33 34 01 01 ff 02 02 07
8e 30 11 0a 01 04 16 05 2a 31 39 33 35 01 01 ff 02 02 07 8f 30 11 0a 01 04 16 05 2a 31 39 33 36
01 01 ff 02 02 07 90 30 11 0a 01 04 16 05 2a 31 39 33 37 01 01 ff 02 02 07 91 30 11 0a 01 06 16
05 2a 32 32 35 30 01 01 ff 02 02 08 ca 30 11 0a 01 06 16 05 2a 32 32 35 31 01 01 ff 02 02 08 cb
30 11 0a 01 06 16 05 2a 32 32 35 32 01 01 ff 02 02 08 cc 30 11 0a 01 06 16 05 2a 32 32 35 33 01
01 ff 02 02 08 cd 30 11 0a 01 06 16 05 2a 32 32 35 34 01 01 ff 02 02 08 ce 30 11 0a 01 06 16 05
2a 32 32 35 35 01 01 ff 02 02 08 cf 30 11 0a 01 06 16 05 2a 32 32 35 36 01 01 ff 02 02 08 d0 30
11 0a 01 06 16 05 2a 32 32 35 37 01 01 ff 02 02 08 d1 30 10 0a 01 05 16 05 2a 38 32 31 34 01 01
ff 02 01 0e 30 0f 0a 01 00 16 03 32 30 30 01 01 ff 02 02 03 e8 30 0f 0a 01 00 16 03 32 30 31 01
01 ff 02 02 03 e9 30 0f 0a 01 00 16 03 32 30 32 01 01 ff 02 02 03 ea 30 0f 0a 01 00 16 03 32 30
33 01 01 ff 02 02 03 eb 30 0f 0a 01 00 16 03 32 30 34 01 01 ff 02 02 03 ec 30 0f 0a 01 00 16 03
32 30 35 01 01 ff 02 02 03 ed 30 0f 0a 01 00 16 03 32 30 36 01 01 ff 02 02 03 ee 30 0f 0a 01 00
16 03 32 30 37 01 01 ff 02 02 03 ef 30 0f 0a 01 00 16 03 32 30 38 01 01 ff 02 02 03 f0 30 0f 0a
01 00 16 03 32 30 39 01 01 ff 02 02 03 f1 30 0f 0a 01 00 16 03 32 31 30 01 01 ff 02 02 03 f2 30

0f 0a 01 00 16 03 32 31 31 01 01 ff 02 02 03 f3 30 0f 0a 01 00 16 03 32 31 32 01 01 ff 02 02 03
f4 30 0f 0a 01 00 16 03 32 31 33 01 01 ff 02 02 03 f5 30 0f 0a 01 00 16 03 32 31 34 01 01 ff 02
02 03 f6 30 0f 0a 01 00 16 03 32 31 35 01 01 ff 02 02 03 f7 30 0f 0a 01 00 16 03 32 31 36 01 01
ff 02 02 03 f8 30 0f 0a 01 00 16 03 32 31 37 01 01 ff 02 02 03 f9 30 0f 0a 01 00 16 03 32 31 38
01 01 ff 02 02 03 fa 30 0f 0a 01 00 16 03 32 31 39 01 01 ff 02 02 03 fb 30 0f 0a 01 00 16 03 32
32 30 01 01 ff 02 02 03 fc 30 0f 0a 01 00 16 03 32 32 31 01 01 ff 02 02 03 fd 30 0f 0a 01 00 16
03 32 32 32 01 01 ff 02 02 03 fe 30 0f 0a 01 00 16 03 32 32 33 01 01 ff 02 02 03 ff 30 0f 0a 01
00 16 03 32 32 34 01 01 ff 02 02 04 00 30 0f 0a 01 00 16 03 32 32 35 01 01 ff 02 02 04 01 30 0f
0a 01 00 16 03 32 32 36 01 01 ff 02 02 04 02 30 0f 0a 01 00 16 03 32 32 37 01 01 ff 02 02 04 03
30 0f 0a 01 07 16 03 32 32 38 01 01 ff 02 02 04 04 30 0f 0a 01 07 16 03 32 32 39 01 01 ff 02 02
04 05 30 0f 0a 01 07 16 03 32 33 30 01 01 ff 02 02 04 06 30 0f 0a 01 07 16 03 32 33 31 01 01 ff
02 02 04 07 30 10 0a 01 01 16 04 38 39 30 31 01 01 ff 02 02 08 48 30 10 0a 01 01 16 04 38 39 30
32 01 01 ff 02 02 08 49 30 10 0a 01 01 16 04 38 39 30 33 01 01 ff 02 02 08 4a 30 10 0a 01 01 16
04 38 39 30 34 01 01 ff 02 02 08 4b 30 10 0a 01 01 16 04 38 39 30 35 01 01 ff 02 02 08 4c 30 10
0a 01 01 16 04 38 39 30 36 01 01 ff 02 02 08 4d 30 10 0a 01 01 16 04 38 39 30 37 01 01 ff 02 02
08 4e 30 10 0a 01 01 16 04 38 39 30 38 01 01 ff 02 02 08 4f

O dado em azul é padrão, podendo alterar os campos com tamanho.

O dado a2 82 04 90 02 01 01 30 82 04 89 02 01 33 7e 82 04 82 a1 82 04 7e é padrão CSTA e o restante é específico da Intelbras.

30 82 04 7a indica que é uma sequência de dados

06 08 2b 06 01 04 01 81 cc 1a Object identifier

a8 82 04 6c IntelbrasData

a1 82 04 68 IntelbrasResponseData

a0 82 04 64 ExistingDevices

Os dados que não estão em azul são referentes aos dispositivos existentes

30 0f 0a 02 00 fe Primeiro device

0a 02 00 fe deviceType

16 03 23 36 39 deviceName (Ex.:#69)

01 01 ff active

02 01 0d deviceId

..... O bloco anterior se repete para cada item da coleção de dispositivos

30 0f 0a 02 00 fe 16 03 23 36 39 01 01 ff 02 01 0d

30 10 0a 02 00 fe 16 03 23 38 37 01 01 ff 02 02 1f 6c

30 12 0a 02 00 fe 16 05 2a 31 39 30 30 01 01 ff 02 02 07 6c

.....

.....

30 10 0a 01 01 16 04 38 39 30 37 01 01 ff 02 02 08 4e

30 10 0a 01 01 16 04 38 39 30 38 01 01 ff 02 02 08 4f último device

Obs.: o tamanho de cada conjunto de dados é diferente pois possuem dados com tamanhos diferentes.

4.4. Solicitação ou definição do turno

Utilizado para solicitar o estado do turno e para alterar o turno.

4.4.1. Solicitação/Definição

Enviar o comando Escape com a opção intelbrasData preenchida com os dados setFeature e turno e como parâmetro do turno um número:

- 0- Para apenas solicitar sem alterar
- 1- Para definir o turno como turno 1
- 2- Para definir o turno como turno 2.
- 3- Para definir o turno como turno 3.
- 4- Para definir o turno como turno 4.

Exemplo do pacote Csta gerado:

A1 1D 02 01 07 02 01 33 30 15 A1 13 30 11 06 08 2B 06 01 04 01 81 CC 1A A8 05 A0 03 81 01 XX

Onde XX é o parâmetro do turno (0,1, 2, 3 ou 4)

O pacote parcial A1 1D 02 01 07 02 01 33 30 15 A1 13 é padrão CSTA e o restante são os dados específicos da Intelbras.

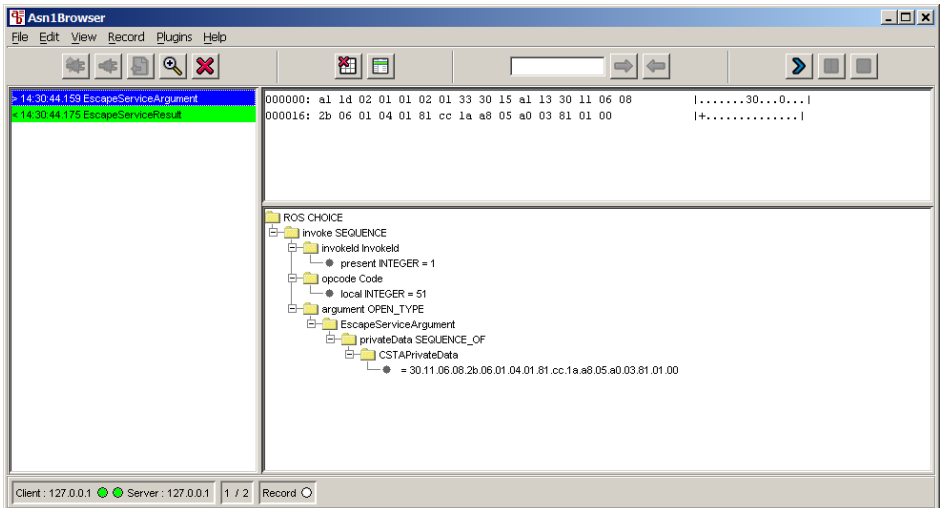
30 11

06 08 2B 06 01 04 01 81 CC 1A Object identifier

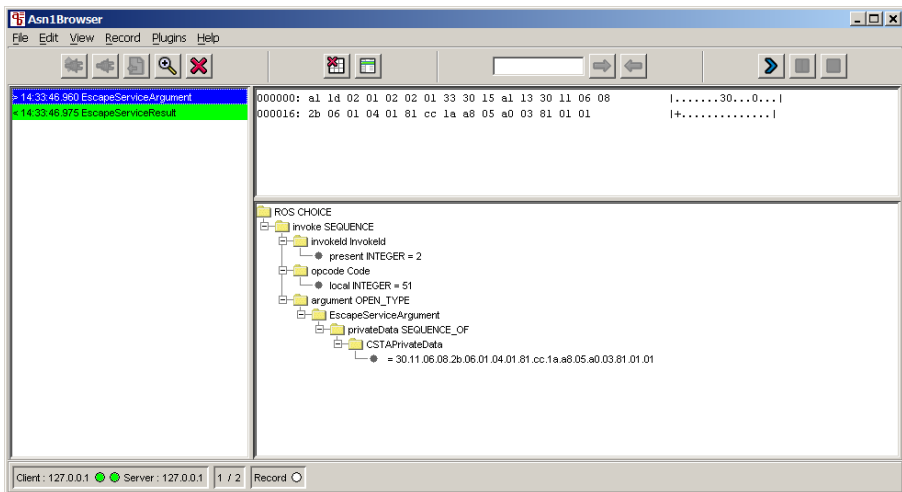
A8 05 IntelbrasData

A0 03 SetFeatureIntelbras

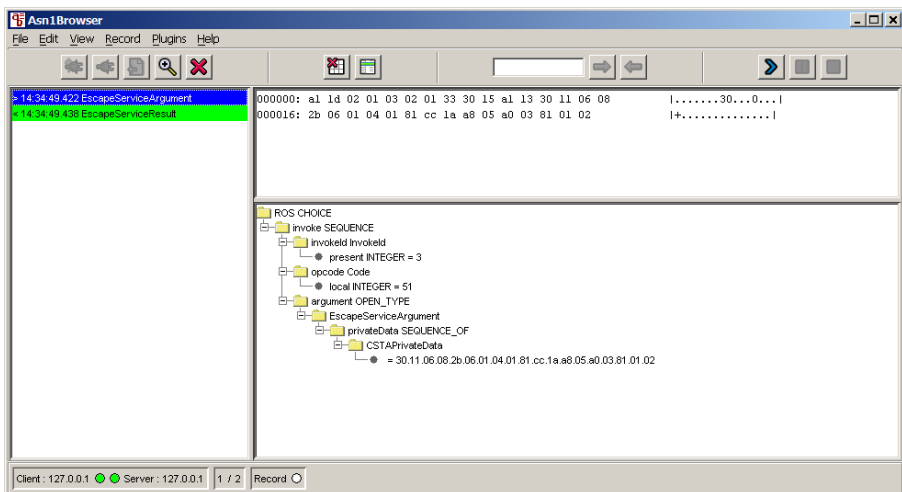
81 01 01 turno (definindo o turno como diurno)



Solicitação do turno



Definição de turno diurno.



Definição de turno noturno

4.4.2. Resposta

Como resposta, é enviado no result do comando Escape, o dado *IntelbrasResponseData*, com o dado turno que foi passado.

Este dado contém:

1. Turno 1,
2. Turno 2,
3. Turno 3,
4. Turno 4.

Exemplo do pacote CSTA gerado

A2 1F 02 01 01 30 1A 02 01 33 7E 15 A1 13 30 11 06 08 2B 06 01 04 01 81 CC 1A A8 05 A1 03 A1 01 01

O pacote parcial A2 1F 02 01 01 30 1A 02 01 33 7E 15 A1 13 é padrão CSTA e o restante são os dados específicos da Intelbras.

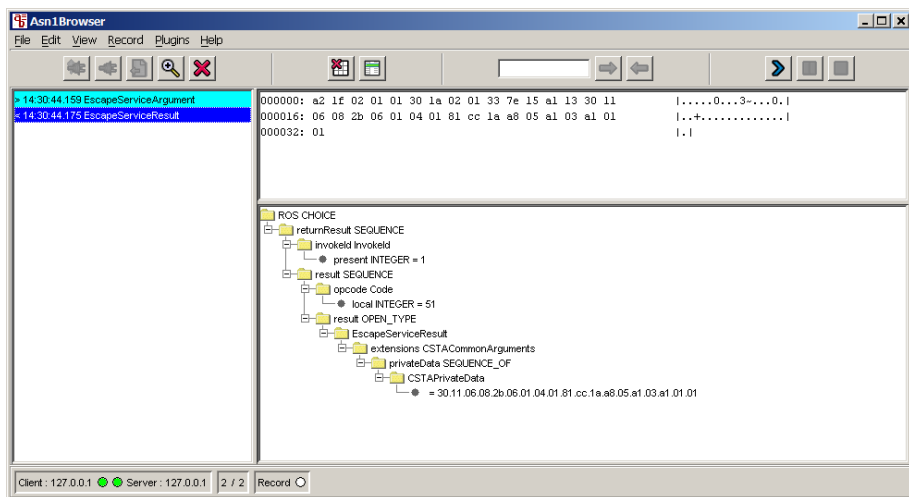
30 11

06 08 2B 06 01 04 01 81 CC 1A Object identifier

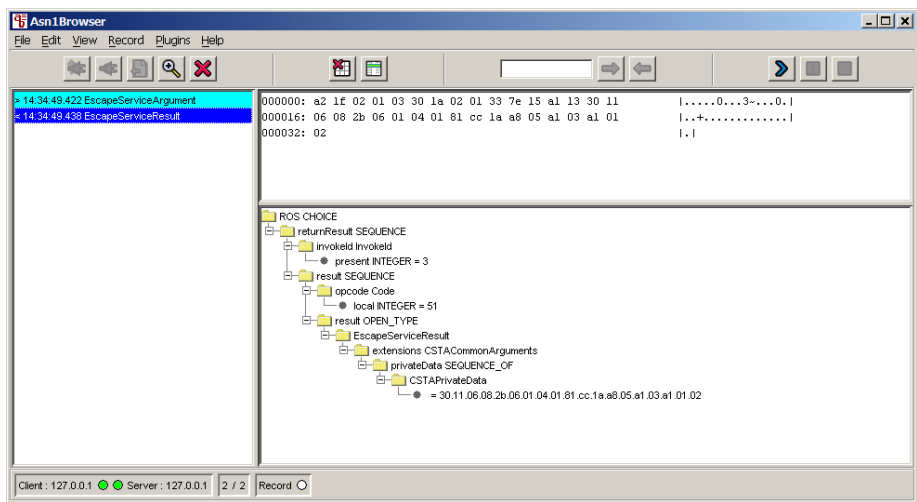
A8 05 IntelbrasData

A1 03 IntelbrasResponseData

81 01 01 turnoAtual com o valor diurno



Resposta para turno diurno



Resposta para turno noturno

5. Protocolo CSTA

5.1. Normas

O serviço de CSTA procura seguir as seguintes normas para CSTA fase 2:

- » Ecma-217
- » Ecma-218
- » Tr-068

Elas podem ser baixas nos sites:

- » <http://www.ecma-international.org/publications/standards/Ecma-217.htm>
- » <http://www.ecma-international.org/publications/standards/Ecma-218.htm>
- » <http://www.ecma-international.org/publications/techreports/E-TR-068.htm>

5.2. Nome de dispositivos

O nome dos dispositivos utilizados pelos comandos e eventos do CSTA é o próprio número do ramal, por exemplo 2000. Estes nomes variam conforme a programação da central.

Alguns nomes podem conter # e * quando estes dispositivos são algum serviço interno da central.

5.2.1. Descrição dos dispositivos utilizados pelo CSTA da linha UnniTI

Identificador	Descrição
#6901 a #6920	Disas
#87	Grupo do correio
*1902	Busca pessoa
*22	Pedido de chamada
*8214	Estacionamento
61	Grupo (é o valor no campo <i>acesso</i> na programação de grupo)
691	Grupo (é o valor no campo <i>acesso</i> na programação de grupo)
8901 a 8924	Juntor Analógico
8908	Juntor Analógico
8931 a 8990	Juntor E1
8991 a 89150	Juntor voip
89300 a 89324	Juntor gsm

Os identificadores dependem da programação da central. Utilize o comando Escape para listar todos os dispositivos existentes.

5.3. Restrições

5.3.1. Comandos

Alguns comandos CSTA, listados anteriormente, somente podem ser utilizados com ramais analógicos e ramais digitais. Os comandos que causam uma ação no ramal, como desligar, atender, transferir, discar somente funcionam e ramais analógicos. Caso o comando seja executado para um dispositivo não permitido, ocorre um erro como resposta ao comando, conforme norma CSTA.

No caso de ramais digitais, é necessário que o headset esteja conectado para que as ações sejam feitas automaticamente. Caso não esteja conectado, a ação terá que ser feita ação manual

Ramais IPs não permitem as ações acima.

O comando *makepredictivecall* funciona para todos os ramais.

O comando *Divert*, com função de desvio, funciona para todos os ramais.

5.3.2. Eventos

Os eventos CSTA estão disponíveis para todos os dispositivos da central.

5.3.3. Operação do ramal analógico

O ramal analógico opera sem a necessidade de se colocar no gancho após o término da ligação. Pode-se tirar o telefone do gancho e mantê-lo assim permanentemente. Para desligar, basta enviar um comando *connectionclear*, para atender deve-se enviar um comando *answer call*, para fazer uma ligação deve-se enviar um comando *makecall*, para discar algum número utilize o comando *sendDtmf*.

Se o telefone estiver fora do gancho, ao receber uma ligação, não ocorre o *ring* do telefone, mas o usuário receberá um sinal sonoro diferenciado indicando que existe uma ligação. Se estiver no gancho, o *ring* será gerado, mas para após o envio do comando *answerCall*.

5.4. Características de Comandos CSTA da UnniTI

5.4.1. Makecall

5.4.1.1. Possibilidades

- » Pode-se utilizar o comando *makecall* para tirar um ramal do gancho, para isto basta definir o *CalledDirectoryNumber* com vazio
- » Pode-se tirar do gancho com o *makecall* e discar via teclado do telefone
- » Pode-se executar um *makecall* com um número parcial e discar o restante pelo teclado.
- » Para discar um tronco é necessário incluir a rota de acesso ao tronco, por exemplo, 030300000 (neste caso 0 é a rota). A central permite várias programações que podem evitar a utilização da rota, e da operadora no caso de ligações DDD.
- » É possível definir um *correlatorData*, um campo texto de até 128 caracteres
- » É possível definir um código de conta (*Account Code*) e a sua senha (*Authentication Code*) através dos parâmetros do comando.

5.4.1.2. Exemplos de comandos makecall

Tirando do gancho

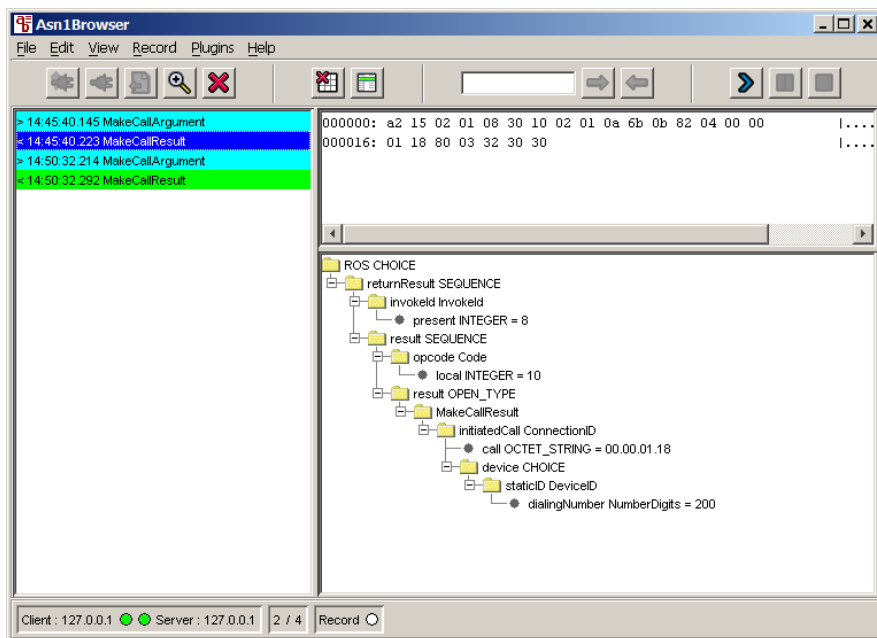
The screenshot shows the Asn1Browser interface. The top menu bar includes File, Edit, View, Record, Plugins, and Help. Below the menu is a toolbar with icons for settings, back, forward, search, and other functions. The main window is divided into two panes. The left pane shows a log of messages: a blue line for '> 14:42:58.030 MakeCallArgument' and a green line for '< 14:42:58.077 MakeCallResult'. The right pane displays a tree structure of the ASN.1 data. The root is 'ROS CHOICE', which contains 'invoke SEQUENCE'. This sequence includes 'invokeld Invokeld' (present INTEGER = 7), 'opcode Code' (local INTEGER = 10), and 'argument OPEN_TYPE'. The 'argument OPEN_TYPE' contains 'MakeCallArgument', which includes 'callingDevice DeviceID' (dialingNumber NumberDigits = 200), 'calledDirectoryNumber CalledDeviceID' (deviceIdentifier ExtendedDeviceID), 'deviceIdentifier DeviceID' (dialingNumber NumberDigits = 200), 'accountCode AccountInfo =', 'authCode AuthCode =', and 'correlatorData CorrelatorData ='. The bottom status bar shows 'Client : 127.0.0.1', 'Server : 127.0.0.1', '1 / 2', and a 'Record' button.

Comando para tirar do gancho

Discagem do ramal 200 para o ramal 201

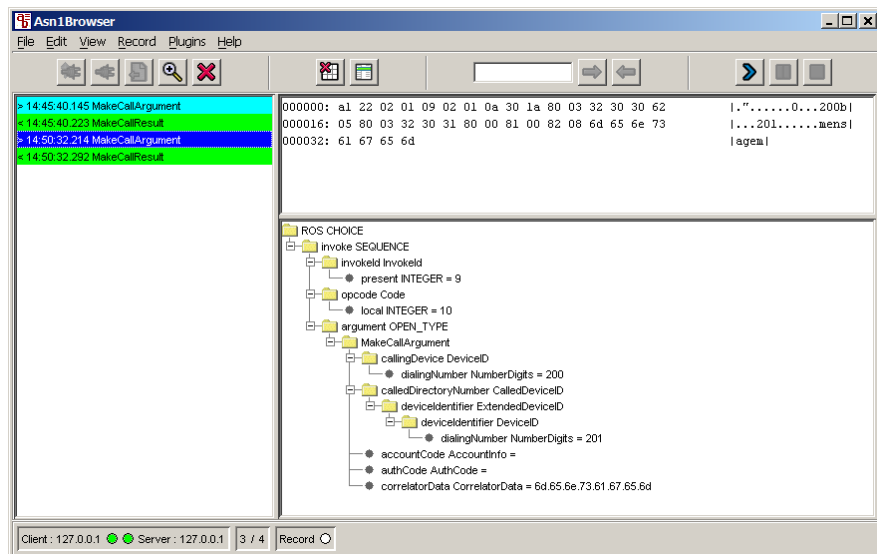
The screenshot shows the Asn1Browser interface. The top menu bar includes File, Edit, View, Record, Plugins, and Help. Below the menu is a toolbar with icons for settings, back, forward, search, and other functions. The main window is divided into two panes. The left pane shows a log of messages: a blue line for '> 14:45:40.145 MakeCallArgument' and a green line for '< 14:45:40.223 MakeCallResult'. The right pane displays a tree structure of the ASN.1 data. The root is 'ROS CHOICE', which contains 'invoke SEQUENCE'. This sequence includes 'invokeld Invokeld' (present INTEGER = 8), 'opcode Code' (local INTEGER = 10), and 'argument OPEN_TYPE'. The 'argument OPEN_TYPE' contains 'MakeCallArgument', which includes 'callingDevice DeviceID' (dialingNumber NumberDigits = 200), 'calledDirectoryNumber CalledDeviceID' (deviceIdentifier ExtendedDeviceID), 'deviceIdentifier DeviceID' (dialingNumber NumberDigits = 201), 'accountCode AccountInfo =', 'authCode AuthCode =', and 'correlatorData CorrelatorData ='. The bottom status bar shows 'Client : 127.0.0.1', 'Server : 127.0.0.1', '1 / 2', and a 'Record' button.

Comando



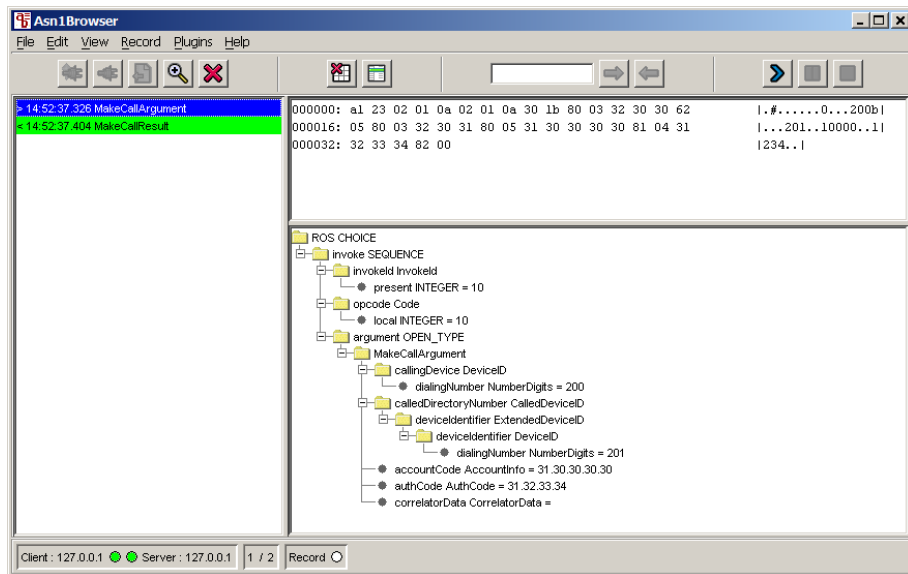
Resposta

Discagem com correlator data



Comando

Código de conta = 10000 e senha = 1234



Comando

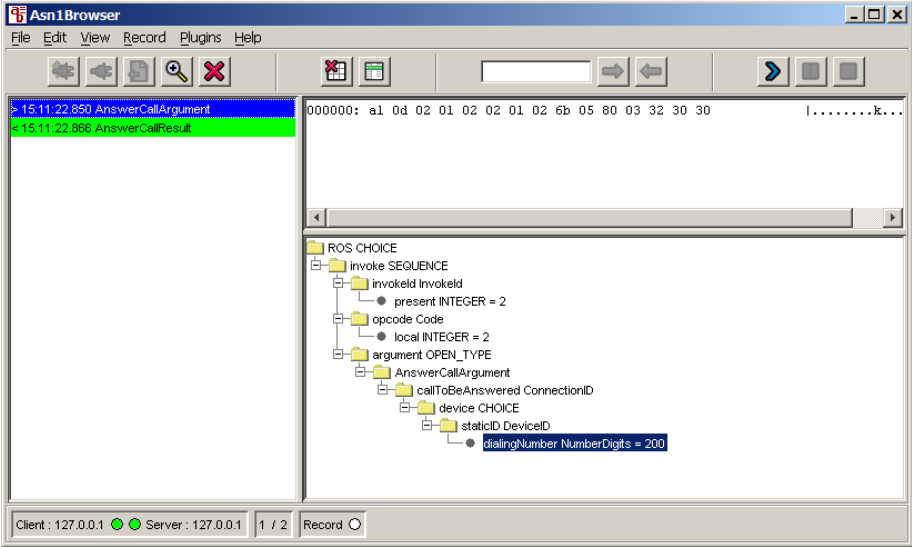
5.4.2. AnswerCall

5.4.2.1. Possibilidades

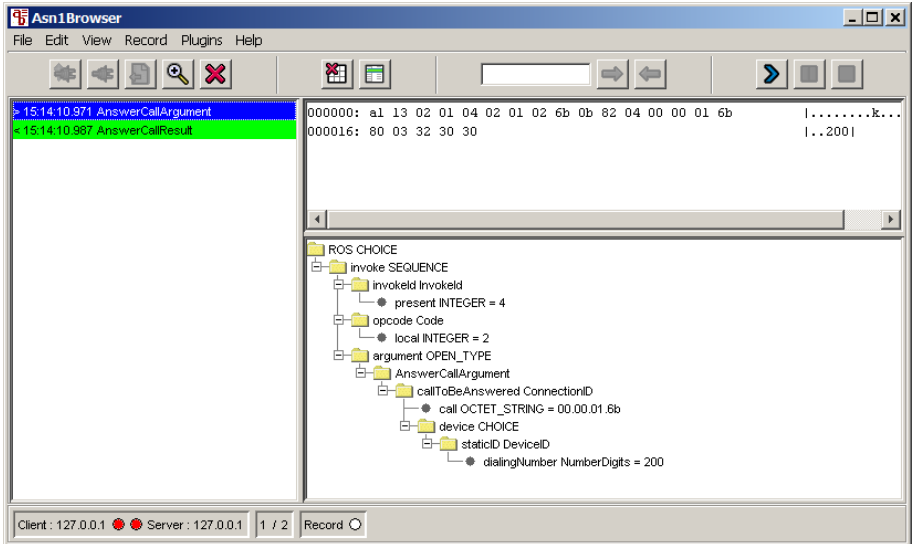
- » Pode-se enviar apenas o ramal caso queira-se atender a ligação que está *ringando*. A *call* é opcional.
- » Pode-se atender uma ligação que está na fila, com isto a ligação que está ringando passa para a fila. Neste caso é necessário definir a *call* da ligação da fila e o ramal que deve atender.

5.4.2.2. Exemplos do comando AnswerCall

Definição apenas do ramal



Definição do ramal e identificador da ligação

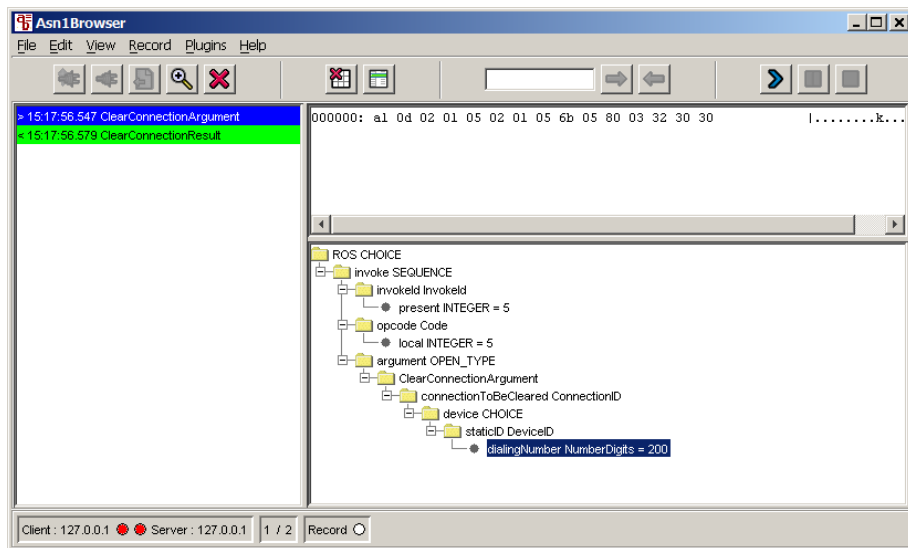


5.4.3. ClearConnection

5.4.3.1. Possibilidades

» Pode-se enviar apenas o ramal. A *call* é opcional.

Exemplo de comando ClearConnection



5.4.4. SendDtmf

5.4.4.1. Possibilidades

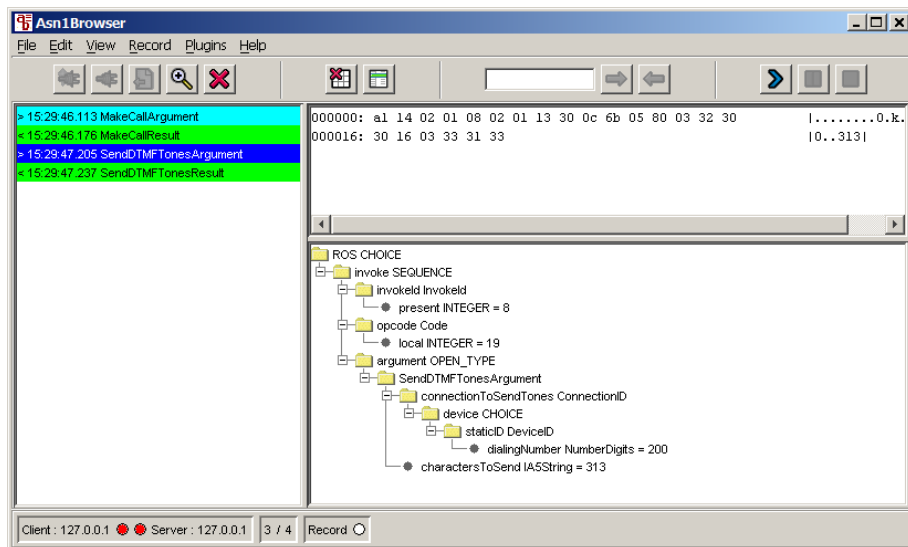
Este comando pode ser utilizado para enviar dígitos dtmf após uma ligação ter sido completada. Isto é útil para se enviar dígitos para URA (unidade remota de atendimento), por exemplo: bancos.

Caso se utilize o comando *makecall* para se tirar do gancho ou manualmente retirar o telefone do gancho, pode-se digitar os números individualmente ou em grupo através do comando *SendDtmf*.

A identificação da ligação é opcional.

5.4.4.2. Exemplo de comando SendDtmf

Discagem após Makecall tirar do gancho



5.5. Exemplos de cenários

O documento *Intelbras_UnniTi_Csta_Cenarios.html* contém diversos cenários de ligações e seus comandos e eventos CSTA.

Está escrito em um formato parecido com a norma TR-068.

Nos cenários, alguns troncos analógicos estão conectados a ramais do própria central e alguns troncos possuem identificação de chamada enquanto outros não tem.

A central possui dois links E1, sendo conectados entre si. Assim uma ligação que sai por um link1 da central pelo outro link na própria central.

A rota para o link 1 do E1 é 83 e do segundo é 84.

intelbras



fale com a gente

Suporte a clientes: ☎ (48) 2106 0006

Fórum: forum.intelbras.com.br

Suporte via chat: chat.apps.intelbras.com.br

Suporte via e-mail: suporte@intelbras.com.br

SAC / Onde comprar? / Quem instala? : 0800 7042767

Produzido por: Intelbras S/A – Indústria de Telecomunicação Eletrônica Brasileira
Rodovia SC 281, km 4,5 – Sertão do Maruim – São José/SC – 88122-001
CNPJ 82.901.000/0014-41 – www.intelbras.com.br

01.23
Indústria brasileira