Contents

Chapter 1

Basics of ASN.1

This chapter defines some basic ASN.1 concepts and describes several most widely used types. It is by no means an authoritative or complete reference. For more information, the

CHAPTER 1. BASICS

```
FruitId ::= ENUMERATED { apple(1), orange(2) }
-- The numbers in braces are optional,
-- the enumeration may be performed
-- automatically by the compiler
ComputerOSType ::= ENUMERATED {
    FreeBSD,          -- will be 0
    Windows,          -- will be 1
    Solaris(5),       -- will remain 5
    Linux,            -- will be 6
    MacOS             -- will be 7
}
```

1.1.4 The OCTET STRING type

This type models the sequence

1.3.2 The SET type

This is a collection of other simple or constructed types. Ordering is not important. The data may arrive in the order which is di erent from the order of speci cation. Data is encoded in the order not necessarily corresponding to the order of speci cation.

1.3.3 The CHOICE type

This type is just a choice between the subtypes speci ed in it. The CHOICE type contains at most one of the subtypes speci ed, and it is always implicitly known which choice is being decoded or encoded. This one resembles the C "union" statement.

1.343 The

Chapter 2

ASN.1 Compiler Usage

The purpose of the ASN.1 compiler, of which this documen

CHAPTER 2.

CHAPTER 2. ASN.1

2.2.2.2 Encoding DER

The Distinguished Encoding Rules is the variant of BER encoding rules which is oriented on representing the structures with length known beforehand. This is probably exactly how you want to encode: either after a BER decoding or after a manual ll-up, the target structure contains the data which size is implicitly known before encoding. The DER encoding i X.509 certi cates.

As with BER decoder, the DER encoder _DEF_Rectangle) or from the stan-alone type descriptor (asn1 function, which is somewhat simpler:

```
/*
 * This is a custom function which writes the
 * encoded output into some FILE stream.
 */
int _write_stream(void *buffer, size_t size, void *app_key) {
    FILE *ostream = app_key;
    size_t wrote;

    wrote = fwrite(buffer, 1, size, ostream);

    return (wrote == size) ? 0 : -1;
}

/*
 * This is the serializer itself,
 * it supplies the DER encoder with the
 * pointer to the custom output function.
 */
ssize_t
simple_serializer(FILE *ostream, Rectangle_t *rect) {
    der_enc_rval_t rval;  /* Return value */

                rval = der_encode(&asn1_DEF_Rect, rect,
                    _write_stream, ostream);
                if(rval.encoded == -1) {
                    /*
                     * Faidure to encode the rectangle data.
                     */
                    fprintf(stderr, "Cannot encode %s: %s\n",
            rval. aided_type->name,
            strerror(errno));
        return -1;
    } else {
        /* Return the number of bytes */
        return rval.encoded;
```

CHAPTER 2. ASN.1

2.2.2.5 Freeing the target structure

Freeing the structure is slightly more complex than it may seem to. When the ASN.1 structure is freed, all the