Vulnerability and Information-Flow Analysis for COTS

S. Jha, B. Miller, T. Reps
University of Wisconsin
CodeSurfer

/* Decompress stdin to stdout. This routine adapts to the codes in the
 * file building the "string" table on-the-fly; requiring no table to
 * be stored in the compressed file. The tables used herein are shared
 * with those of the compress() routine. See the definitions above.
 */

decompress() {
    register char_type *stackp;
    register int finchar;
    register code_int code, oldcode, incode;

    /* As above, initialize the first 256 entries in the table.
    */
    maxcode = MAXCODE(n_bits = INIT_BITS);
    for ( code = 255; code >= 0; code-- ) {

{ 
    mid = (low+high)>>1;
    if (rno < prob_tab[c][mid])
        high = mid;
    else if (rno > prob_tab[c][mid])
        low = mid + 1;
    else /* exact match found - unlikely */
        return ((char)mid);
}

return ((char)low);

double ran2()
{
    seedi=((314157*seedi)+19)&0xffffffff;
    return ( (double) seedi/(double)0xffffffff);
}

double ran()
/* See "Random Number Generators: Good Ones Are Hard To Find", */
/* Park & Miller, CACM 31#10 October 1988 pages 1192-1201. */
/*****************************/
/* THIS IMPLEMENTATION REQUIRES AT LEAST 32 BIT INTEGERS ! */
/*****************************/
#define A_MULTIPLIER  16807L
#define M_MODULUS     2147483647L /* (2**31)-1 */
#define QUOTIENT 127773L /* 2147483647L / 16807L */
#define R_REMAINDER 28365L /* 2147483647L % 16807L */
{
    long lo;
    long hi;
    long test;

    hi = seedi / QUOTIENT;

Decompress stdin to stdout. This routine adapts to the codes in the file building the "string" table on-the-fly, requiring no table to be stored in the compressed file. The tables used herein are shared with those of the compress() routine. See the definitions above.

```c
decompress()
{
    register char_type *stackp;
    register int finchar;
    register code_int code, oldcode, incode;

    /*
    * As above, initialize the first 256 entries in the table.
    */
    maxcode = MAXCODE(n_bits = INITBITS);
    for (code = 255; code >= 0; code-- ) {
        tab_prefixof(code) = 0;
        tab_suffixof(code) = (char_type)code;
    }
    free_ent = ((block_compress) ? FIRST : 256 );
```
InBuff = (unsigned char *)from_buf;
outBuff = (unsigned char *)to_buf;
do_decomp = action;

if (do_decomp == 0) {
    compress();
    #ifdef DEBUG
    if(verbose)    dump_tab();
    #endif /* DEBUG */
    } else {
    /* Check the magic number */
    if (nomagic == 0) {
        if ((getbyte() != (magic_header[0] & 0xFF)) ||
                (getbyte() != (magic_header[1] & 0xFF))) {
            fprintf(stderr, "stdin: not in compressed format\n");
            exit(1);
        }
        maxbits = getbyte(); /* set -b from file */
        block_compress = maxbits & BLOCK_MASK;
        maxbits &= BIT_MASK;
        maxmaxcode = 1 << maxbits;
        fsize = 1000000; /* assume stdin large for USERMEM */
        if(maxbits > BITS) {
            fprintf(stderr,
"stdin: compressed with %d bits, can only handle %d bits maxbits, BITS\n";
            exit(1);
        }
    }
    #ifndef DEBUG
decompress();
    #else

Browsing a Dependence Graph

Pretend this is your favorite browser

What does clicking on a link do?

Or you move to an internal tag

You get a new page
Program point: `if (do_decomp == 0) { compress(); #ifdef ...`
InBuff = (unsigned char *)from_buf;
outBuff = (unsigned char *)to_buf;

if (do_decomp == 0) {
    compress();
    #ifdef DEBUG
    if (verbose) dump_tab();
    #endif /* DEBUG */
} else {
    /* Check the magic number */
    if ((nomagic == 0) {
        if ((getbyte() != (magic_header[0] & 0xFF)) || (getbyte() != (magic_header[1] & 0xFF))) {
            fprintf(stderr, "stdin: not in compressed format\n");
            exit(1);
        }
        maxbits = getbyte(); /* set -b from file */
        block_compress = maxbits & BLOCK_MASK;
        maxbits &= BIT_MASK;
        maxmaxcode = 1 << maxbits;
        fsize = 1000000; /* assume stdin large for USRMEM */
        if (maxbits > BITS) {
            fprintf(stderr, "stdin: compressed with %d bits, can only handle %d bits maxbits, BITS");
            exit(1);
        }
    }
    #ifndef DEBUG
decompress();
    #else
}

int main(int argc, char *argv[])
{
    int count, i, oper;
    int comp_count, new_count;
    char start_char;
    int N;
    char C;

    printf("SPEC 129.compress harness\n");
    scanf("%i %c %i", &count, &start_char, &seedi);
    printf("Initial File Size:%i Start character:%c\n", count, start_char
    fill_text_buffer(count, start_char, orig_text_buffer);
    for (i = 1; i <= 25; i++)
    {
        new_count=add_line(orig_text_buffer, count, i, start_char);
        count=new_count;
        oper=COMPRESS;
        printf("The starting size is: %d\n", count);
        comp_count=spec_select_action(orig_text_buffer, count, oper, comp_text
        printf("The compressed size is: %d\n", comp_count);
        oper=UNCOMPRESS;
        new_count=spec_select_action(comp_text_buffer, comp_count, oper, new_t
        printf("The compressed/uncompressed size is: %d\n", new_count);
        compare_buffer(orig_text_buffer, count, new_text_buffer, new_count);
    }
Front Ends
- EDG
- ANSI C
- C++
- IdaPro

x86 binary code
- Java
- UML (Rose/RT)
- Verilog
- VHDL
- Jovial

Other infrastructure: command-line, preprocessor, include-file instances, library, and loader support