UNIX News

Number 10, October 1976

Security Patch

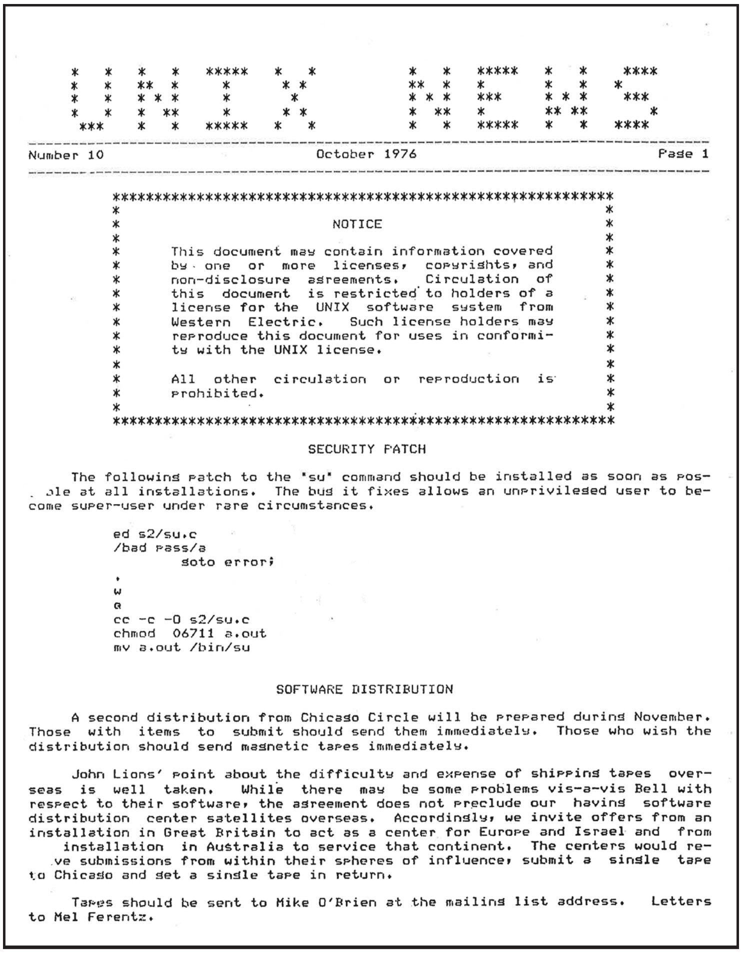
The following patch to the "su" command should be installed as soon as possible at all installations. The bug it fixes allows an unprivileged user to become super-user under rare circumstances.

```
ed s2/su.c
/bad pass/a
   goto error;
.
w
q?
cc -c -O s2/su.c
chmod 06711 a.out
mv a.out /bin/su
```

Software Distribution

A second distribution from Chicago Circle will be prepared during November. Those with items to submit should send them immediately. Those who wish the distribution should send magnetic tapes immediately.

John Lions' point about the difficulty and expense of shipping tapes overseas is well taken. While there may be some problems vis-a-vis Bell with respect to their software, the agreement does not preclude our having software distribution center satellites overseas. Accordingly, we invite offers from an installation in Great Britain to act as a center for Europe and Israel and from an installation in Australia to service that continent. The centers would receive submissions from within their spheres of influence, submit a single tape to Chicago and get a single tape in return.



```
 *   *  *   *  *****  *   *        *   *  *****  *   *   ****
 *   *  **  *    *     * *         **  *  *      *   *  *
 *   *  * * *    *      *          * * *  ***    * * *   ***
 *   *  *  **    *     * *         *  **  *      ** **      *
  ***   *   *  *****  *   *        *   *  *****  *   *  ****
```

Number 10 October 1976 Page 1

```
*****************************************************************
*                                                               *
*                           NOTICE                              *
*                                                               *
*       This document may contain information covered           *
*       by one or more licenses, copyrights, and                *
*       non-disclosure agreements.  Circulation of              *
*       this document is restricted to holders of a             *
*       license for the UNIX software system from               *
*       Western Electric.  Such license holders may             *
*       reproduce this document for uses in conformi-           *
*       ty with the UNIX license.                               *
*                                                               *
*       All other circulation or reproduction is                *
*       prohibited.                                             *
*                                                               *
*****************************************************************
```

SECURITY PATCH

The following patch to the 'su' command should be installed as soon as possible at all installations. The bug it fixes allows an unprivileged user to become super-user under rare circumstances.

```
ed s2/su.c
/bad pass/a
        goto error;
.
w
q
cc -c -O s2/su.c
chmod  06711 a.out
mv a.out /bin/su
```

SOFTWARE DISTRIBUTION

A second distribution from Chicago Circle will be prepared during November. Those with items to submit should send them immediately. Those who wish the distribution should send magnetic tapes immediately.

John Lions' point about the difficulty and expense of shipping tapes overseas is well taken. While there may be some problems vis-a-vis Bell with respect to their software, the agreement does not preclude our having software distribution center satellites overseas. Accordingly, we invite offers from an installation in Great Britain to act as a center for Europe and Israel and from installation in Australia to service that continent. The centers would receive submissions from within their spheres of influence, submit a single tape to Chicago and get a single tape in return.

Tapes should be sent to Mike O'Brien at the mailing list address. Letters to Mel Ferentz.

UNIX News, Number 10, was published in October 1976 by Professor Melvin Ferentz of Brooklyn College of CUNY. We have included excerpts from that issue and have reproduced the text as it appeared in the original, including any typographic errors. Note: We have not included the mailing list and other addresses and telephone numbers that appeared in the original issue.

University of New South Wales

From John Lions

On August 27th a group of more than 30 persons gathered at the University of New South Wales for our first local Users meeting.

David Morrison reported on the initial experience of the University of Newcastle with UNIX. They are currently heavily committed to using Basic Under RSTS on a PDP 11/45, and it was the quality of UNIX Basic which principally colored their reaction. They will undoubtedly be happier after trying the Harvard Software which was described to the meeting by Peter Ivanov.

Ian Johnstone spent some time discussing the security of UNIX. At the School of Electrical Engineering at the University of New South Wales the PDP 11 is run as an open shop staffed by casual, volunteer student operators. It is almost impossible to set up file access permissions in such a way that routine operations can be carried out safely (e.g. killing recalcitrant programs before shut-down) without leaving a loop-hole for the self-aggrandisement of users to super-users. A number of other modifications have been found necessary; groups have been disabled and "cron", for example, as a willing accomplice in crime, as been banished. However as long as the system console is accessible the most determined users cannot be prevented from patching the "suser" route directly. Setting the code for this routine into ROM would be a step in the right direction.

A UNSW implementation of Pascal "S" by John O'Neill, a final year undergraduate, was discussed and the meeting diverted on for a short while onto the subject of "Pascal" in general.

Most participants felt that the meeting was a success and another meeting has been planned for February 18th, 1977. It was agreed that there is a real need for cooperation between UNIX users in view of the unconventional nature of UNIX support.

Particular concern was expressed regarding the co-operative acquisition of software from overseas. Because of the distances involved this presents some difficulties and expense and it would certainly be more convenient for us if one local UNIX licensee, having acquired some item of software could distribute it to other local licensees (subject of course to completion of any required non-disclosure agreements production of DEC licenses, etc.). We have already attempted to raise this matter with Western Electric but so far have received no response.

Beware of icheck –s (or Change It)

From George Rolf, Katholieke Universiteit

I have been vaguely wondering for a while why everything I wrote seemed so much slower than the commands that came with the Unix system (version 6). Now I know why. Icheck –s will rearrange the freelist of a file system in the order of ascending block numbers, where mkfs initializes the freelist with consecutive entries 3 blocks apart on an RK disk, or 4 blocks on an RP. After I dumped the system and restored it onto a fresh file system I felt much happier.

I have also replaced the routine makefree( ) in icheck.c with the code reproduced below, which I borrowed from mkfs.c. Note that the change described in Unix Newsletter number 8 (August 1976) has been taken into account. Also note that this icheck –s produces an optimized lay-out for an RF disk, which the original mkfs does not. Our mkfs does of course.

I stumbled upon this discrepancy between mkfs and icheck while doing some measurements to find out what an optimal layout of the disk might be. I found myself reinventing the wheel. The measurements were the following. I made an executable file of 24 blocks (and one indirect block), and put it in various ways on one cylinder of an RK disk, with the indirect block in an adjacent cylinder. Exactly the same lay-outs were tried out on the RF disk (with 24 block "cylinders" instead of 8 blocks). I then timed read commands of the whole file at once, as well as exec-s on the file. For both devices the optimum is at a distance of 2 between consecutive file blocks. With both tests running at the same time, a distance of 3 blocks on both devices gave the best results, so those were the numbers I took.

I don't know why the Unix system as it is distributed doesn't have a special lay-out for the RF disk. At our installation, we have put the /tmp files on the RF disk, which appears to be a good idea. We have to keep the file system on the second RK drive interchangeable, and our RF disk has only one platter, which makes it a bit inconvenient to put the root directory there.

The only relevant measurements for this sort of questions are of course those obtained from heavy standard loads, or bench marks simulating such a load. We don't have either. Furthermore, the situation might be altogether different with different or more controllers, or for example with a 60 cycle RF disk, which runs 20% faster than ours. If anyone has any further ideas or other experimental results, I will be very anxious to learn of them.

```
in routine check():
change makefree(); to makefree(file);

freebl(i)
int i:
{
        if ((baab[i>>4)&07777] & (1<<(i&017))) == 0)
                free(i);
}

makefree(file)
char *file;
{
        register char *i, *j;
        char *n, *m;
        char *high, *low;
        static char adr[100], flag[100];

        for(j = file; j[0]; j++)
                if(j[0] == 'r')
                        switch (j[1]){
                        case 'k':
                                n = 24;
                                m = 3;
                                break;
                        case 'p':
                                n = 10;
                                m = 4;
                                break;
                        case 'f':
                                n = 8;
                                m = 3;
                                break;
                        default: ;
        }
```

```
    if (n > 100) n = 100;
    for(i = 0; i < n; i++)
        flag[i] = 0;
    j = 0;
    for(i = 0; i < n; i++) {
        while (flag[j])
                j = (j + 1) % n;
        adr[i] = j;
        flag[j]++;
        j = (j + n) % n;
    }
    sblock.s_nfree = 0;
    sblock.s_ninode = 0;
    sblock.s_flock = 0;
    sblock.s_ilock = 0;
    sblock.s_faod = 0;

    high = sblock.s_fsize - 1;
    low = sblock.s_isize + 2;
    free(0);
    for(i = high; lrem(0, i+1, n); i--) {
        if (i < low)
            break;
        freebl(i);
    }
    for(; i >= low + n-1; i =- n)
        for (j = 0; j < n; j++)
            freebl(i-adr[j]);
    for(; i >= low; i--)
        freebl(i);
    bwrite(1, &sblock);
    close(fi);
    sync();
    return;
}
```

Southern Illinois University at Carbondale

From Ray Kohring

Our department has been receiving the UNIX News since this Spring (issue #5 was the first one we received). What we have found most useful are the patches to the software which have been printed. In this light we would like to know if it would be possible to get any back issues that we missed. Any of them would be appreciated.

Our department owns a CAL DATA 135 which is emulating a PDP 11/40 on which we are running UNIX. In General, UNIX has ran well on our setup (exluding finding a missing wire on the MMU), but there are a couple of things which I felt were worth mentioning.

The first has to do with what happens when the user's stack-pointer is odd (that is not even, as opposed to unusual). What happens is the CPU goes through the stack error routine (specifically, red-stack limit) upon a buss-error, which clears the kernel stack-pointer (even though it was a user-mode error). This locks UNIX into a very tight loop (about 8 instructions long) which is retrapping on every attempt to stack something. I cured this by adding the code on the next page to m40.s. I haven't been able to determine if this happens on DEC CPUs also, but an easy check would be to run

```
        dec         sp
        mov         $1,-(sp)
```

and see if it loops.

The second problem is unique to CAL DATA systems with the micro-programming option. Accidently executing op-codes 7-17 (octal) causes all sorts of wonderous things to happen, since these are the spare op-codes (including EFM). The easy (?) cure is to load the appropriate ACM locations with a branch to the illegal instruction trap routine and enable it to replace the second page of control memory. A second alternative is to load routines to do common tasks, such as csav and cret, and modify the c-compiler to use those op-codes. One of our people (Carl Ebeling) has been working on this idea so if anybody wants to try it we could send you what he has done so far.

Note: This patch tests the stack pointer (kernel) to see if it is zero. If it is, it resets it to the top of the user block (where it probably should be) and copies the ps-pc from 0 to the correct stack locations. If it really is a kernel stack error, there will still be a panic.

```
ed m40.s
/trap:/
+
a
        tst     sp          /is the stack pointer zero?
        bne      lf         /no, we're still safe
        clr     177774          /stack limit register, the ps
                                /was put here by accident
        mov      $142000,sp     /restore the sp
        mov      2,-(sp)        /restack ps
        mov      0,-(sp)        /restack pc
        clz|clc                 /reset cc's to show buss-error
        mov      ps,-4(sp)      /redo properly
1:
.
```

The Pennsylvania State University

From Edward C. Horvath

I was directed to you by the UNIX documentation as a contact point for the UNIX user's group. If that is no longer appropriate, please forward this letter to whomever now fills that role.

The Computer Science Department here at Penn State recently acquired a PDP-11/34 and the UNIX system, and we are interested in hearing of and/or participating in the activities of the UNIX user's group.

Our system consists of an 11/34 (which includes memory management but no stack limit option), 96Kb core, a dual-drive RK11, RX11 floppy disk, and an 8-line DZ11 mux. This is a one-cabinet configuration which prices out (after haggling) at around $36K (circa June 1976). We are currently running only two typewriters (console and one DL11) and are in the process of constructing drivers for the RX and DZ. We soon expect to be running 6-8 users, and to expand core to 128K. We also have a 120 1/m Potter printer which we hope to interface to the DZ.

I should mention that UNIX (specifically rk unix) will not boot directly on the 11/34; there are minor programming differences between the 11/40 and the 11/34, none of which seem to surface when the system runs. However, the 11/34 comes standard with a blank front panel—an on/off switch, but no switch register. This drives the system into an infinite bus timeout trap loop when it tries to print the 'mem=' message. We were able to overcome this by laboriously hand-patching the system, a process which I will be happy to coach any new user on; I have attached a copy of the procedure to this letter for your files. We have not, to date, had any other problems with incompatibilities, but I will so inform you if they arise.

First, you can register us in the UNIX user's group.

Second, you can put us in contact with any other users who have constructed/are constructing drivers for the RX or DZ. We would be happy to share ideas and/or software; if we are the first and only developers for either device we will be happy to contribute any software we develop when it becomes available. Please inform me of any format restrictions or distribution clearing houses.

For your information, I have already informed Ken Thompson at BTL of the switch register problem; I'm not sure what steps he will take.

Thank you for your assistance; I look forward to your correspondence.

Bringing up UNIX (specifically rkunix) on the PDP-11/34

This document is for users who wish to run UNIX (6th Ed.) on the standard 11/34—i.e., with the standard front panel. If you have a switch register, the procedure described in 'setting up UNIX' should work just fine. In any case, this document is a supplement to 'setting up UNIX'.

First, generate the binary code RK05 pack. We cannot vouch for the procedures in 'setting up UNIX' for doing this from magtape, as we received the system already on RK05's.

Next, you have to locate the first block of 'rkunix' on the pack. 'rkunix' is a son of 'root', which is the root of the directory tree. (See File System (V) in the UNIX Programmer's Manual). 'rkunix' is described by i node 193 (base 10), which is the 6th i node of the 13th block of the i node list, which starts at byte 240 (base 8) of logical block 16 (base 8) of the RK05 pack (magic number 21). Note that 'rkunix' is a large file, so addr 0 points not at the first block of 'rkunix', but rather at the block of block pointers for 'rkunix'. On our distribution pack, addr 0 is 2723 (base 8). This converts to a 'magic number' for the RK11, namely 3703 (base 8), which may be deposited in the RKDA register to read the block of pointers. Again, on our distribution pack, the first pointer has value 2675 (base 8), which has magic number 3645 (base 8). If your pack disagrees in any way, calculate your own magic numbers! (Use the RK11 description of RKDA in the peripherals manual).

By the way, for magic number xxxx, the following console emulator sequence reads the desired block into core locations 0:777.

```
L    177406
D    177400
D    0
D    xxxx
L    177404
D    5
```

Once you have the first block of rkunix loaded in this way, perform the following sequence:

```
L    346
D    0
L    277406
D    177400
D    0
D    xxxx (magic number for first block)
L    177404
D    3
```

The above places a halt instruction in the trap sequence, and writes the block back out.

Steps thus far need only be done once; what follows is the new boot sequence:

1. Type OK, advance the paper, and hit return. The system should respond with 0.

2. Type 'rkunix' and hit return. The system will flutter a bit, then halt.

3. Hit the boot switch to bring in the emulator, and enter the following sequence:

```
L    176
D    100000
L    326
D    5767
L    12340
D    176
L    41236
D    176
L    0
S
```

The above sequence modifies the system to look at location 176 (base 8) for the contents of the switch register, loads 176 with 100000 (for a single user system, L 176 should be followed by D 173030), repairs the damage we did to the version on the pack, and finally restarts. The sequence described in 'setting up UNIX' now applies.

All of the above nonsense can, of course, be obviated if you can beg or etc. a couple of hours on a 40 or 45, or even get a 'loaner' front panel from your friendly DEC repairman, or, best of all, already have a running UNIX system. In any case, to avoid further heartache, you'll want to recompile the system to boot clean. In addition to the steps indicated in /usr/sys/run (watch out for ar!), you should:

Edit /usr/sys/param.h to change the value of SW to 0176

Make sure /usr/sys/ken/prf.c and user/sys/ken/sys4.c get recompiled and replaced in /usr/sys/libl.

The new system should come up clean (ours did!).