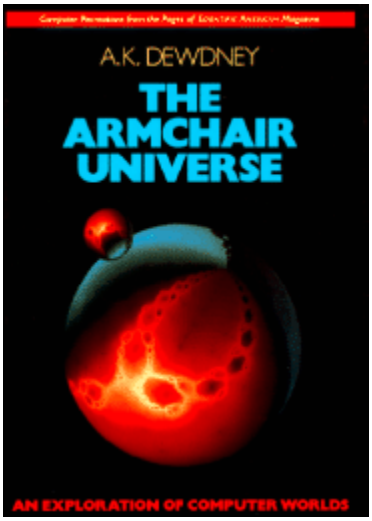
The Animal Episode

by John Walker

The Legend



The *Computer Recreations* column of *Scientific American* for March 1985 related a somewhat garbled version of the ANIMAL saga a little more than ten years after the events upon which it was based. David D. Clark of the Laboratory for Computer Science at the Massachusetts Institute of Technology recounted the story in a letter to columnist A. K. Dewdney. (The original columns are collected in Dewdney's wonderful 1988 book, *The Armchair Universe* [ISBN 0-716-71938-X], which should be on the shelf of anybody interested in computer exploration of the universe of mathematics.) Already, the ANIMAL story had begun to mutate into an "urban legend" of computerdom—moving through space and time, recalled by a "friend of a friend", and accreting additional details with every telling.

> "A similar story concerns a game called ANIMAL in which a program tries to determine what animal a human is thinking of by playing a form of Twenty Questions. …ANIMAL achieved the ability to reproduce itself.
>
> "On a very early computer system, which lacked any shared directory structure but also lacked any protection tools, a programmer invented a novel way of making the game available to several users. A version of the game existed in one user's directory. Whenever he played the game, the program made a copy of itself into another directory. … If that directory had previously no version of ANIMAL, the game had been offered to yet another user."
>
> …ANIMAL was such a popular game that eventually every directory in the company system contained a copy. "Furthermore, as employees of the company were transferred to other divisions…they took ANIMAL as well, and thus it spread from machine to machine within the company." … Only when someone devised a more "virulent" version of the game was the situation brought under control. When the new version of ANIMAL was played, it copied itself into other directories not once but twice. Given enough time, it was thought, this program would eventually overwrite all the old versions of ANIMAL. After a year had passed, a certain date triggered each copy of the

> new ANIMAL program. “Instead of replicating itself twice whenever it was invoked, it now played one final game, wished the user ‘goodbye’ and then deleted itself, And thus ANIMAL was purged from the system.”

What Actually Happened

It's rare enough to hear an urban legend about which you know any of the facts, but it's astonishing to be reading a major magazine and encounter such a story which you not only personally recall but which you're *responsible for*. Right after I read the account of ANIMAL in Dewdney's column, I dashed off the following letter to relate the actual 1975 events on which the ANIMAL legend was based. (I've slightly edited the letter to remove 1985-era references which are now out of date; the substance is unchanged.)

February 21, 1985

A. K. Dewdney
Editor, *Computer Recreations*
Scientific American
415 Madison Avenue
New York, NY 10017

Greetings,

I was amused to see the story of the “Animal” program which migrated from computer to computer in your March column. From time to time I see versions of this apocryphal story. Details vary from version to version. As it happens, I know something about what really occurred, which is actually pretty interesting.

I wrote the program, you see.

In January of 1975 I was working in the data processing department of a large engineering company in the United States. This company had recently installed a UNIVAC® 1100/42 computer system, a very modern and extremely fast mainframe computer which was run under the Univac 1100 series operating system, Exec-8. This Univac operating system was for its time quite advanced and offered significantly greater security than, for example, Unix does today. The system provided shared directories as well as private, protected files.

In April 1974 I had written an “Animal program to end all Animal programs”. Unlike most versions of this game which simply have a continuously expanding database with no correctness tests, mine incorporated self-correction: if a later player contradicted information in the database, the program would establish whether the question at which the decision chain diverged could be answered either way for the animal in question, and if not, removed the previous entry. Thus, erroneous information would be corrected by later players. The program also performed ELIZA-style substitution of nouns for

pronouns and sentence transformations, and nondeterministic replies to seem more natural. This program, which was written entirely in Univac assembly language (yes, we did things like that in those days), was an immediate hit, and many other Univac users asked me to send them copies of the program. This, of course was before the days of worldwide data networks, so this involved writing magnetic tapes and mailing them to each requestor.

What a drag.

So, in January '75 I decided to do something about this. I wrote a general purpose subroutine called PERVADE, which could be called by any program. When called, it created an independent process which, while the host program was going about its business, would examine all the directories accessible to its caller. If a directory did not contain a copy of the program, or contained an older version, PERVADE would copy the version being executed into that directory. PERVADE was very fastidious, and took great care not to destroy, for example, an unrelated user program with the same name.

As a result, after executing a program incorporating PERVADE, all directories which could be accessed by a user would contain up-to-date copies of the program. Some of these directories would inevitably be shared with other users, who, upon running the program, would find copies of it in their directories and so on. Eventually, a privileged user (one who could write the system protected directories, e.g., super-user in Unix) would run the program, and it would copy itself into the system library, thus making it available to all users. From that point it was a matter of days until it was in every directory in the system (and this was a *big* system; it supported hundreds of users and had tens of thousands of files).

This program did not in any way violate or subvert the security of the system, nor did it take advantage of any bug or design flaw in the operating system. It spread into successively more protected directories in what today is called a “classic Trojan Horse attack”. In 1975, when I thought of it, I just called it “a neat idea”.

Since users exchanged tapes with other installations in the company and with other Univac sites regularly, copies of the program would find themselves on these tapes, and the program would then establish itself on other systems. Aside from the fact that I wanted to distribute ANIMAL, it made a perfect host for PERVADE—its dialogue with the user was extended and gave PERVADE plenty of time to do its work unbeknownst to the user.

This all worked quite nicely. About a week after I turned the program loose in San Francisco, it was well established on another of the company's machines located near Washington, D.C. Univac had a very unusual and little known feature in their system: if you just entered a directory name, the system would execute the program most recently

placed in that directory. This obscure feature was used there to run some engineering programs, with the result (ANIMAL having copied itself most recently into most of these directories) that a user expecting to, say, run a thermal design program would often be asked:

```
Think of an animal.
```

Imagine the frivolity this engendered.

Within a month or so, ANIMAL was well established at numerous installations of Univac computers. Since these sites regularly exchanged tapes with Univac's software development centre at Roseville, Minnesota, soon the program was loose there. Several people told me at the time that they received software distribution tapes from Univac which contained ANIMAL. I never confirmed this myself. But I don't doubt it.

ANIMAL would indeed copy itself into every directory on a system, but it was small, and when updating itself overwrote the old copy, so it would not clog the disc with multiple copies. Throughout the entire episode, the ANIMAL phenomenon was viewed with benign amusement by the managers of the systems on which it established itself. Lost on almost all of these people was one of my major points, to make them think of what could have happened if I were not a nice guy and ANIMAL an innocuous program.

The story of the “HUNTER” program is apocryphal. Several of my associates at the time talked about ways to get rid of the plague, and the program described by Clark was proposed as a solution. Writing itself into a directory twice is nonsense, of course, the object is just to replace all copies. The time delay is intended to ensure that all true copies are replaced before D-day. (But what if only one remains, I reminded proponents: a program which reproduces itself without bound is more survival-prone than one which commits suicide, and what are the odds you'd get every one, especially those on archive tapes which are kept for decades? And only one is needed to start the spread again.)

As it turned out, no HUNTER was needed. As I alluded to earlier, ANIMAL was a very good citizen: it took extensive precautions to avoid even the slightest risk of inadvertently destroying user data or programs in the directories into which it copied itself. This involved very extensive system-dependent checks of the status of files accessible by a user. In 1976, Univac released a new operating system in which the format of the system tables examined by PERVADE changed. When PERVADE examined these modified tables, it concluded that no files were eligible to have copies of ANIMAL installed in them, so on any machine converted to this new operating system, ANIMAL became a benign game program.

Univac's operating system change *was not* brought on by ANIMAL, it was, in fact, designed well before I wrote PERVADE and loosed

ANIMAL on the world. The change did not, however, make PERVADE impossible or even more difficult; the program could have been easily modified to continue to spread under the new system, but by that time I was no longer involved with the Univac system, and no other programmer took on the task of modifying PERVADE. Years later, many users looked in their directories, and seeing ANIMAL, executed it and enjoyed the game. PERVADE, unable to understand the new directory structure, simply exited with no action. The user, happily guessing animals, never wondered where the ANIMAL program came from, or what its strange history held.

For your apocrypha file, I am including source listings, in Univac assembly code, of PERVADE. The comments at the beginning tell the story well, even a decade later, if you allow for some Univac-ese ("activity" means "process", "element" means "file", "program file" means "directory", "run" means "job", and "PCT" is the table of files accessible by a user).

I am also enclosing a contemporaneous account of the story. The highlighted information is of interest. I never loosed "PERVADE SQUARED" or "PERVADE FACTORIAL" (which *would* have filled up all the files on a system) on the world, thank goodness.

As I said, all of this was before the era of Unix and worldwide networks. A program which works exactly like ANIMAL with PERVADE is, in fact, much easier to implement under Unix today than on the Univac system in 1975. In fact, the technique will work on any system which allows multiple processes per user and shared directories. And with networks, things can happen much, much faster. The consequences are left as an exercise for the reader….

Unlike many of the apocryphal stories in computerdom, there are dozens of people who remember this episode and can confirm all the details. I enjoyed the ANIMAL affair at the time, and enjoyed reading the account of it in your column. I hope you've enjoyed this account of what really happened.

These days I'm still pervading software, the AutoCAD computer aided design and drafting program, but since it does not distribute itself automatically and I charge money for it, I have to work much harder and don't have as much time for recreations such as ANIMAL. But then I think of Unix and all those humming network lines….

Have fun,

John Walker

Fingered!

A couple of months later, *Scientific American*'s *Computer Recreations* set the record

straight in a piece entitled “Core War Encore”.

> Meanwhile the source of the rumor that sparked the invention of Core War has finally been revealed. In “Core War” I told the story of CREEPER and REAPER. In “Core War Encore” I described ANIMAL, a game program similar to the Core War software that replicated itself in the computers of all players who used the same time-sharing system. The author of this program, John Walker, has stepped forward to claim responsibility for the most successful version. Actually, ANIMAL is only part of the story. Inside the game program was another piece of software called PERVADE that was responsible for reproducing the program. …

Self-Reproducing Legends

Order this book from amazon.com

Even the most robust, quasi-eternal, and omnipresent self-reproducing computer program stands humbled before an urban legend, which reproduces in the fertile medium of the human mind and is transmissible by the most casual anecdote. So it was, five years after the ANIMAL story had first been fully explained in the press, that a further evolved version appeared in the 1990 book *The Devouring Fungus* by Karla Jennings (ISBN 0-393-30732-8). This book (without a single usable citation to any original source) relates the Animal story as follows in its chapter on “Electronic Plagues”.

> … The earliest viruses were harmless tracers secretly encoded into programs who wanted to follow the paths of illegally copied programs. Others were accidents, program design flaws that wreaked havoc on files or made copies of themselves until they choked the computer's memory space dead. Some even started as games, like Animal, which became a pest when it duplicated itself like crazy. Created at Dartmouth College in the early 1970s, *(the hallmark of an evolving urban legend—it moves through time and space so nobody can actually pin down what really happened)*, it asked users to think of an animal, then queried them, `DOES IT HAVE FOUR LEGS? DOES IT LIVE IN THE OCEAN?` *et cetera*.
>
> Eventually the program worked down its knowledge tree until it guessed the animal's identity right and printed `I WIN`! If it was wrong, it printed `TELL ME SOMETHING ABOUT YOUR ANIMAL THAT'S DIFFERENT FROM ALL THE OTHER ANIMALS`, adding new information to its tree and the trees of every other Animal program in the system.
>
> The expanding Animal spread like prairie fire, growing so fast that it choked out the system disk space. The staff destroyed all

> its copies but one, which somebody had saved under a different name, and that copy took off again like a hurricane. *The legend grows with time….*
>
> Then a clever staff member released a ferocious new Animal red in tooth and claw, identical to the original except that when the user signed off, it made two copies of itself, growing much faster through the system than the old program and destroying any of the old programs that it encountered. Soon the old Animal was extinct and the new Animal king of the wiry jungle, but not for long. The new Animal's master programmed it to commit suicide at a certain date, and it disappeared.

Looking Backward

The entire ANIMAL escapade, from release through neutering with the next release of Exec-8, only lasted about a year (although copies of ANIMAL can be found, to this day, in UNIVAC software archives). The legend, however, has far outlasted the reproductive life of its progenitor. Every year or so, in a book, magazine article, USENET newsgroup, or other medium, I come across a reference to ANIMAL, usually an amalgam of other "friend of a friend" recountings.

More than 20 years after ANIMAL was born and briefly thrived, it's hard to imagine that the first computer virus was viewed, in those days, not as a security threat but entirely with amusement. I was, at the time, very conscious of the threat a malicious program might have created and frequently, in discussions with other programmers and site managers, used ANIMAL as an example of an easily-exploited yet little-perceived vulnerability in a sophisticated operating system which its vendor considered reasonably secure (and was, in fact, far more secure than typical present-day personal computer operating systems). I had hoped that ANIMAL would, in some small way, get people thinking about operating systems in which security was based on a formal model, as opposed to the chewing-gum and baling wire *ad hoc*, "find a hole and plug it" approach of the vast majority of systems, then and now.

In 1975, when most computers were isolated islands which communicated with one another primarily by exchanging magnetic tapes or card decks, with modem connections used mostly by remote terminals accessing a central computer, the risk posed by a malicious ANIMAL-like program was small and could have been easily contained. Today, as the Internet is knitting together all the computers of Earth into an intercommunicating and and cooperative tapestry, the lessons of ANIMAL are more significant than ever.

Back to UNIVAC Memories
PERVADE Source Code
ANIMAL Source Code

Compiled by John Walker
August 21, 1996