= Plan 9 from Bell Labs =

Plan 9 from Bell Labs is a distributed operating system , originally developed by the Computing Sciences Research Center at Bell Labs between the mid @-@ 1980s and 2002 . It takes some of the principles of Unix , developed in the same research group , but extends these to a networked environment with graphics terminals .

In Plan 9 , virtually all computing resources , including files , network connections , and peripheral devices , are represented through the file system rather than specialized interfaces . A unified network protocol called 9P ties a network of computers running Plan 9 together , allowing them to share all resources so represented .

The name Plan 9 from Bell Labs is a reference to the Ed Wood 1959 cult science fiction Z @-@ movie Plan 9 from Outer Space . Also , Glenda , the Plan 9 Bunny , is presumably a reference to Wood 's film Glen or Glenda . The system continues to be used and developed by operating system researchers and hobbyists .

= = History = =

Plan 9 from Bell Labs was originally developed , starting mid @-@ 1980s , by members of the Computing Science Research Center at Bell Labs , the same group that originally developed Unix and C. The Plan 9 team was initially led by Rob Pike , Ken Thompson , Dave Presotto and Phil Winterbottom , with support from Dennis Ritchie as head of the Computing Techniques Research Department . Over the years , many notable developers have contributed to the project including Brian Kernighan , Tom Duff , Doug McIlroy , Bjarne Stroustrup and Bruce Ellis .

Plan 9 replaced Unix as Bell Labs 's primary platform for operating systems research . It explored several changes to the original Unix model that facilitate the use and programming of the system , notably in distributed multi @-@ user environments . After several years of development and internal use , Bell Labs shipped the operating system to universities in 1992 . Three years later , in 1995 , Plan 9 was made available for commercial parties by AT & T via the book publisher Harcourt Brace . With source licenses costing $ 350 , AT & T targeted the embedded systems market rather than the computer market at large ; Ritchie commented that the developers did not expect to do " much displacement " given how established other operating systems had become .

By early 1996 , the Plan 9 project had been " put on the back burner " by AT & T in favor of Inferno , intended to be a rival to Sun Microsystems ' Java platform . In the late 1990s , Bell Labs ' new owner Lucent Technologies dropped commercial support for the project and in 2000 , a third release was distributed under an open source license . A fourth release under a new free software license occurred in 2002 .

A user and development community , including current and former Bell Labs personnel , produced minor daily releases in form of ISO images . Bell Labs hosted the development . The development source tree is accessible over the 9P and HTTP protocols and is used to update existing installations . In addition to the official components of the OS included in the ISOs , Bell Labs also hosts a repository of externally developed applications and tools .

= = Design concepts = =

Plan 9 is a distributed operating system , designed to make a network of heterogeneous and geographically separated computers function as a single system . In a typical Plan 9 installation , users work at terminals running the window system rio , and they access CPU servers which handle computation @-@ intensive processes . Permanent data storage is provided by additional network hosts acting as file servers and archival storage .

Its designers state that :

[t] he foundations of the system are built on two ideas : a per @-@ process name space and a simple message @-@ oriented file system protocol .

The first idea means that , unlike on most operating systems , processes (running programs) each

have their own view of the namespace , corresponding to what other operating systems call the file system ; a single path name may refer to different resources for different processes . The potential complexity of this setup is controlled by a set of conventional locations for common resources .

The second idea means that processes can offer their services to other processes by providing virtual files that appear in the other processes ' namespace . The client process 's input / output on such a file becomes inter @-@ process communication between the two processes . This way , Plan 9 generalizes the Unix notion of the filesystem as the central point of access to computing resources . It carries over Unix 's idea of device files to provide access to peripheral devices (mice , removable media , etc .) and the possibility to mount filesystems residing on physically distinct filesystems into a hierarchical namespace , but adds the possibility to mount a connection to a server program that speaks a standardized protocol and treat its services as part of the namespace .

For example , the original window system , called 8 ½ , exploited these possibilities as follows . Plan 9 represents the user interface on a terminal by means of three pseudo @-@ files : mouse , which can be read by a program to get notification of mouse movements and button clicks , cons , which can be used to perform textual input / output , and bitblt , writing to which enacts graphics operations (see bit blit) . The window system multiplexes these devices : when creating a new window to run some program in , it first sets up a new namespace in which mouse , cons and bitblt are connected to itself , hiding the actual device files to which it itself has access . The window system thus receives all input and output commands from the program and handles these appropriately , by sending output to the actual screen device and giving the currently focused program the keyboard and mouse input . The program does not need to know if it is communicating directly with the operating system 's device drivers , or with the window system ; it only has to assume that its namespace is set up so that these special files provide the kind of input and accept the kind of messages that it expects .

Plan 9 's distributed operation relies on the per @-@ process namespaces as well , allowing client and server processes to communicate across machines in the way just outlined . For example , the cpu command starts a remote session on a compute server . The command exports part of its local namespace , including the user 's terminal 's devices (mouse , cons , bitblt) , to the server , so that remote programs can perform input / output using the terminal 's mouse , keyboard and display , combining the effects of remote login and a shared network filesystem .

= = = 9P protocol = = =

All programs that wish to provide services @-@ as @-@ files to other programs speak a unified protocol , called 9P . Compared to other systems , this reduces the number of custom programming interfaces . 9P is a generic , medium @-@ agnostic , byte @-@ oriented protocol that provides for messages delivered between a server and a client . The protocol is used to refer to and communicate with processes , programs , and data , including both the user interface and the network . With the release of the 4th edition , it was modified and renamed 9P2000 .

Unlike most other operating systems , Plan 9 does not provide special application programming interfaces (such as Berkeley sockets , X resources or ioctl system calls) to access devices . Instead , Plan 9 device drivers implement their control interface as a file system , so that the hardware can be accessed by the ordinary file input / output operations read and write . Consequently , sharing the device across the network can be accomplished by mounting the corresponding directory tree to the target machine .

= = = Union directories and namespaces = = =

Plan 9 allows the user to collect the files (called names) from different directory trees in a single location . The resulting union directory behaves as the concatenation of the underlying directories (the order of concatenation can be controlled) ; if the constituent directories contain files having the same name , a listing of the union directory (ls or lc) will simply report duplicate names . Resolution

of a single path name is performed top @-@ down : if the directories top and bottom are unioned into u with top first , then u / name denotes top / name if it exists , bottom / name only if it exists and top / name does not exist , and no file if neither exists . No recursive unioning of subdirectories is performed , so if top / subdir exists , the files in bottom / subdir are not accessible through the union .

A union directory can be created by using the bind command :

In the example above , / arm / bin is mounted at / bin , the contents of / arm / bin replacing the previous contents of / bin . Inferno 's bin directory is then union mounted after / bin , and Alice 's personal bin directory is union mounted before . When a file is requested from / bin , it is first looked for in / usr / alice / bin , then in / arm / bin , and then finally in / usr / inferno / Plan9 / arm / bin .

The separate @-@ namespaces @-@ for @-@ separate @-@ processes thus replaces the notion of a search path in the shell . Where Unix shells have a list of directories to search for programs when given a command , the Plan 9 shell only looks in the directory / bin ; adding commands is done by binding several directories together to appear as a single / bin .

Furthermore , the kernel can keep separate mount tables for each process , and can thus provide each process with its own file system namespace . Processes ' namespaces can be constructed independently , and the user may work simultaneously with programs that have heterogeneous namespaces . Namespaces may be used to create an isolated environment similar to chroot , but in a more secure way .

Plan 9 's union directory architecture inspired 4.4BSD and Linux union file system implementations , although the developers of the BSD union mounting facility found the non @-@ recursive merging of directories in Plan 9 " too restrictive for general purpose use " .

= = = Special virtual filesystems = = =

= = = = / proc = = = =

Instead of having system calls specifically for process management , Plan 9 provides the / proc file system . Each process appears as a directory containing information and control files which can be manipulated by the ordinary file IO system calls .

The file system approach allows Plan 9 processes to be managed with simple file management tools such as ls and cat ; however , the processes cannot be copied and moved as files .

= = = = / net = = = =

Plan 9 does not have specialised system calls or ioctls for accessing the networking stack or networking hardware . Instead , the / net file system is used . Network connections are controlled by reading and writing control messages to control files . Sub @-@ directories such as / net / tcp and / net / udp are used as an interface to their respective protocols .

= = = Unicode = = =

To reduce the complexity of managing character encodings , Plan 9 uses Unicode throughout the system . The initial Unicode implementation was ISO 10646 . Ken Thompson invented UTF @-@ 8 , which became the native encoding in Plan 9 . The entire system was converted to general use in 1992 . UTF @-@ 8 preserves backwards compatibility with traditional null terminated strings , enabling more reliable information processing and the chaining of multilingual string data with Unix pipes between multiple processes . Using a single UTF @-@ 8 encoding with characters for all cultures and regions eliminates the need for switching between code sets .

= = = Combining the design concepts = = =

Though interesting on their own , the design concepts of Plan 9 were supposed to be most useful when combined together . For example , to implement a network address translation (NAT) server , a union directory can be created , overlaying the router 's / net directory tree with its own / net . Similarly , a virtual private network (VPN) can be implemented by overlaying in a union directory a / net hierarchy from a remote gateway , using secured 9P over the public Internet . A union directory with the / net hierarchy and filters can be used to sandbox an untrusted application or to implement a firewall . In the same manner , a distributed computing network can be composed with a union directory of / proc hierarchies from remote hosts , which allows interacting with them as if they are local .

When used together , these features allow for assembling a complex distributed computing environment by reusing the existing hierarchical name system .

= = Software for Plan 9 = =

As a benefit from the system 's design , most tasks in Plan 9 can be accomplished by using ls , cat , grep , cp and rm utilities in combination with the rc shell (the default Plan 9 shell) .

Factotum is an authentication and key management server for Plan 9 . It handles authentication on behalf of other programs such that both secret keys and implementation details need only be known to Factotum .

= = = Graphical programs = = =

Unlike Unix , Plan 9 was designed with graphics in mind . After booting , a Plan 9 terminal will run the rio windowing system , in which the user can create new windows displaying rc . Graphical programs invoked from this shell replace it in its window .

The plumber provides an inter @-@ process communication mechanism which allows system @-@ wide hyperlinking .

Sam and acme are Plan 9 's text editors .

= = = Storage system = = =

Plan 9 supports the Kfs , Paq , Cwfs , FAT , and Fossil file systems . The last was designed at Bell Labs specifically for Plan 9 and provides snapshot storage capability . It can be used directly with a hard drive or backed with Venti , an archival file system and permanent data storage system .

= = = Software development = = =

The distribution package for Plan 9 includes special compiler variants and programming languages , and provides a tailored set of libraries along with a windowing user interface system specific to Plan 9 . The bulk of the system is written in a dialect of C (ANSI C with some extensions and some other features left out) . The compilers for this language were custom built with portability in mind ; according to their author , they " compile quickly , load slowly , and produce medium quality object code " .

A concurrent programming language called Alef was available in the first two editions , but was then dropped for maintenance reasons and replaced by a threading library for C.

= = = Unix compatibility = = =

Though Plan 9 was supposed to be a further development of Unix concepts , compatibility with preexisting Unix software was never the goal for the project . Many command line utilities of Plan 9 share the names of Unix counterparts , but work differently .

Plan 9 can support POSIX applications and can emulate the Berkeley socket interface through the ANSI / POSIX Environment (APE) that implements an interface close to ANSI C and POSIX , with

some common extensions (the native Plan 9 C interfaces conform to neither standard) . It also includes a POSIX @-@ compatible shell . APE 's authors claim to have used it to port the X Window System (X11) to Plan 9 , although they do not ship X11 " because supporting it properly is too big a job " . Some Linux binaries can be used with the help of a " linuxemu " (Linux emulator) application ; however , it is still a work in progress . Vice versa , the vx32 virtual machine allows a slightly modified Plan 9 kernel to run as a user process in Linux , supporting unmodified Plan 9 programs .

= = Reception = =

= = = Comparison to contemporary operating systems = = =

In 1991 , Plan 9 's designers compared their system to other early nineties operating systems in terms of size , showing that the source code for a minimal (" working , albeit not very useful ") version was less than one @-@ fifth the size of a Mach microkernel without any device drivers (5899 or 4622 lines of code for Plan 9 , depending on metric , vs. 25530 lines) . The complete kernel comprised 18000 lines of code . (According to a 2006 count , the kernel was then some 150 @,@ 000 lines , but this was compared against more than 4 @.@ 8 million in Linux .)

Within the operating systems research community , as well as the commercial Unix world , other attempts at achieving distributed computing and remote filesystem access were made concurrently with the Plan 9 design effort . These included the Network File System and the associated vnode architecture developed at Sun Microsystems , and more radical departures from the Unix model such as the Sprite OS from UC Berkeley . Sprite developer Welch points out that the SunOS vnode architecture is limited compared to Plan 9 's capabilities in that it does not support remote device access and remote inter @-@ process communication cleanly , even though it could have , had the preexisting UNIX domain sockets (which " can essentially be used to name user @-@ level servers ") been integrated with the vnode architecture .

One critique of the " everything is a file " , communication @-@ by @-@ textual @-@ message design of Plan 9 pointed out limitations of this paradigm compared to the typed interfaces of Sun 's object @-@ oriented operating system , Spring :

Plan 9 constrains everything to look like a file . In most cases the real interface type comprises the protocol of messages that must be written to , and read from , a file descriptor . This is difficult to specify and document , and prohibits any automatic type checking at all , except for file errors at run time . (...) [A] path name relative to a process ' implicit root context is the only way to name a service . Binding a name to an object can only be done by giving an existing name for the object , in the same context as the new name . As such , interface references simply cannot be passed between processes , much less across networks . Instead , communication has to rely on conventions , which are prone to error and do not scale .

A later retrospective comparison of Plan 9 , Sprite and a third contemporary distributed research operating system , Amoeba , found that

the environments they [Amoeba and Sprite] build are tightly coupled within the OS , making communication with external services difficult . Such systems suffer from the radical departure from the UNIX model , which also discourages portability of already existing software to the platform (...) . The lack of developers , the very small range of supported hardware and the small , even compared to Plan 9 , user base have also significantly slowed the adoption of those systems (...) . In retrospect , Plan 9 was the only research distributed OS from that time which managed to attract developers and be used in commercial projects long enough to warrant its survival to this day .

= = = Impact = = =

Plan 9 demonstrated that an integral concept of Unix ? that every system interface could be represented as a set of files ? could be successfully implemented in a modern distributed system . Some features from Plan 9 , like the UTF @-@ 8 character encoding of Unicode , have been

implemented in other operating systems . Unix @-@ like operating systems such as Linux have implemented 9P , Plan 9 's file system , and have adopted features of rfork , Plan 9 's process creation mechanism . Additionally , in Plan 9 from User Space , several of Plan 9 's applications and tools , including the sam and acme editors , have been ported to Unix and Linux systems and have achieved some level of popularity . Several projects seek to replace the GNU operating system programs surrounding the Linux kernel with the Plan 9 operating system programs . The 9wm window manager was inspired by 8 ½ , the older windowing system of Plan 9 ; wmii is also heavily influenced by Plan 9 . In computer science research , Plan 9 has been used as a grid computing platform and as a vehicle for research into ubiquitous computing without middleware .

However , Plan 9 has never approached Unix in popularity , and has been primarily a research tool :

[I] t looks like Plan 9 failed simply because it fell short of being a compelling enough improvement on Unix to displace its ancestor . Compared to Plan 9 , Unix creaks and clanks and has obvious rust spots , but it gets the job done well enough to hold its position . There is a lesson here for ambitious system architects : the most dangerous enemy of a better solution is an existing codebase that is just good enough .

Other factors that contributed to low adoption of Plan 9 include the lack of commercial backup , the low number of end @-@ user applications , and the lack of device drivers .

Plan 9 proponents and developers claim that the problems hindering its adoption have been solved , that its original goals as a distributed system , development environment , and research platform have been met , and that it enjoys moderate but growing popularity . Inferno , through its hosted capabilities , has been a vehicle for bringing Plan 9 technologies to other systems as a hosted part of heterogeneous computing grids .

Several projects work to extend Plan 9 , including 9atom and 9front . These forks augment Plan 9 with additional hardware drivers and software , including an improved version of the Upas e @-@ mail system , the go compiler , Mercurial version control system support , and other programs . Plan 9 was ported to the Raspberry Pi single @-@ board computer . The Harvey project attempts to replace the custom Plan 9 C compiler with GCC , to leverage modern development tools such as GitHub and Coverity and speed up development .

= = License = =

Starting with the release of Fourth edition on April 2002 , the full source code of Plan 9 from Bell Labs was freely available under Lucent Public License 1 @.@ 02 , which is considered to be open source license by the Open Source Initiative (OSI) , free software license by the Free Software Foundation , and it passes the Debian Free Software Guidelines .

In February 2014 , the University of California , Berkeley , has been authorized by the current Plan 9 copyright holder ? Alcatel @-@ Lucent ? to release all Plan 9 software previously governed by the Lucent Public License , Version 1 @.@ 02 under the GNU General Public License , Version 2 .

= = Derivatives and forks = =

Inferno is a descendant of Plan 9 , and shares many design concepts and even source code in the kernel , particularly around devices and the Styx / 9P2000 protocol . Inferno shares with Plan 9 the Unix heritage from Bell Labs and the Unix philosophy . Many of the command line tools in Inferno were Plan 9 tools that were translated to Limbo .

9atom augments the Plan 9 distribution with the addition of a 386 PAE kernel , an amd64 cpu and terminal kernel , nupas , extra pc hardware support , IL and Ken 's fs .

9front is a fork of Plan 9 . It was started to remedy a perceived lack of devoted development resources inside Bell Labs , and has accumulated various fixes and improvements .

9legacy is an alternative distribution . It includes a set of patches based on the current Plan 9 distribution .

Akaros is designed for many @-@ core architectures and large @-@ scale SMP systems .

Harvey OS is an effort to get the Plan 9 code working with gcc and clang .