Advanced Operating System and Virtualization

Information on the stack

Hiroaki Fukuda

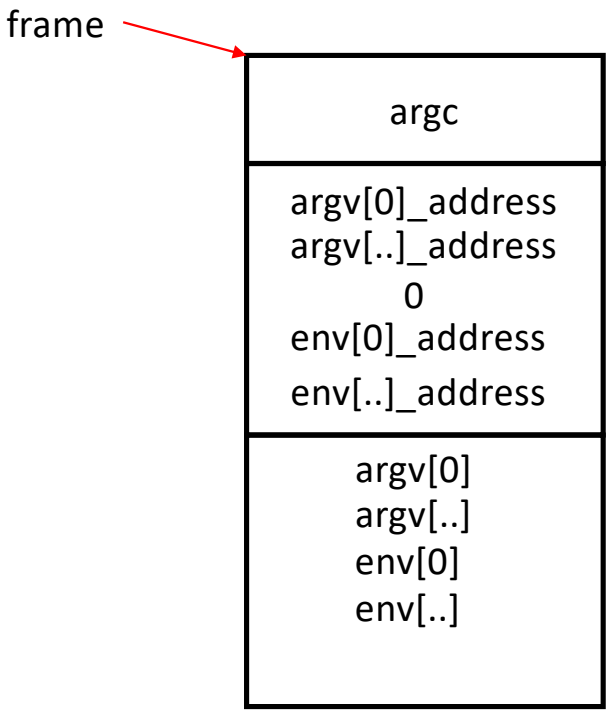
_execve.c

{minix2}/usr/src/lib/posix/_execve.c

```
int execve(const char *path, char * const *argv, char * const *envp)
{
	char * const *ap;
	char * const *ep;
	char *frame;
	char **vp;
	char *sp;
	size_t argc;
	size_t frame_size;
	size_t string_off;
	size_t n;
	int ov;
	message m;

	/* Assumptions: size_t and char *, it's all the same thing. */

	/* Create a stack image that only needs to be patched up slightly
	 * by the kernel to be used for the process to be executed.
	 */

	ov= 0;			/* No overflow yet. */
	frame_size= 0;		/* Size of the new initial stack. */
	string_off= 0;		/* Offset to start of the strings. */
	argc= 0;		/* Argument count. */

	for (ap= argv; *ap != nil; ap++) {
		n = sizeof(*ap) + strlen(*ap) + 1;
		frame_size+= n;
		if (frame_size < n) ov= 1;
		string_off+= sizeof(*ap);
		argc++;
	}

	for (ep= envp; *ep != nil; ep++) {
		n = sizeof(*ep) + strlen(*ep) + 1;
		frame_size+= n;
		if (frame_size < n) ov= 1;
		string_off+= sizeof(*ap);
	}
```

```
	/* Add an argument count and two terminating nulls. */
	frame_size+= sizeof(argc) + sizeof(*ap) + sizeof(*ep);
	string_off+= sizeof(argc) + sizeof(*ap) + sizeof(*ep);

	/* Align. */
	frame_size= (frame_size + sizeof(char *) - 1) & ~(sizeof(char *) - 1);

	/* The party is off if there is an overflow. */
	if (ov || frame_size < 3 * sizeof(char *)) {
		errno= E2BIG;
		return -1;
	}

	/* Allocate space for the stack frame. */
	if ((frame = (char *) sbrk(frame_size)) == (char *) -1) {
		errno = E2BIG;
		return -1;
	}

	/* Set arg count, init pointers to vector and string tables. */
	* (size_t *) frame = argc;
	vp = (char **) (frame + sizeof(argc));
	sp = frame + string_off;

	/* Load the argument vector and strings. */
	for (ap= argv; *ap != nil; ap++) {
		*vp++= (char *) (sp - frame);
		n= strlen(*ap) + 1;
		memcpy(sp, *ap, n);
		sp+= n;
	}
	*vp++= nil;

	/* Load the environment vector and strings. */
	for (ep= envp; *ep != nil; ep++) {
		*vp++= (char *) (sp - frame);
		n= strlen(*ep) + 1;
		memcpy(sp, *ep, n);
		sp+= n;
	}
	*vp++= nil;

	/* Padding. */
	while (sp < frame + frame_size) *sp++= 0;
```

int main(int argc, char** argv, char** env)



int main(int argc, char** argv, char** env)