

Predicate Analysis with BLAST 2.7.3

 Pavel Shved, Mikhail Mandrykin,
and Vadim Mutilin

ISPRAS

Institute for System Programming of the Russian Academy of Sciences

BLAST 2.5



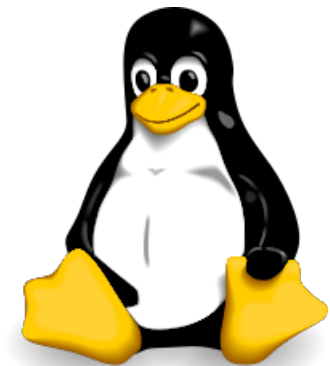
Berkeley
Lazy
Abstraction
Software
Verification
Tool

BLAST 2.7.3



Linux Driver Verification Project Goals

- Provide infrastructure for application of verification tools to the Linux device drivers
- Research new verification approaches in the industrial settings
- Improve quality of the Linux device drivers



Witness Visualization

- Recommendations for using existing witness format for the ease of visualization
- Extensions of the format

Error trace	.../lkbc/drivers/usb/serial/kobil_sct.c
<pre> 942 Global variable declarations 125 ldv_main 5 217 ldv_insmod 2 Initialize the module after insmod with 'usb_serial_module_init' 134 ldv_usb_serial_scenario_1 392 Device has been inserted in the system. Invoke callback probe 407 Do any local initialization of the device and private memory st 581 Line discipline is attached to the terminal. Invoke callback op 171 kobil_open 180 ldv_kzalloc 28 Check that correct flag was used in context of interr 647 Write a block of characters to the tty device in atomic context 647 Switch to interrupt context 296 kobil_write 338 GFP_ATOMIC flag should be used in context of interrupt </pre>	<pre> 326 327 328 329 todo = priv->filled - priv->cur_pos; 330 331 while (todo > 0) { 332 /* max 8 byte in one urb (endpoint size) */ 333 length = min(todo, port->interrupt_out_size); 334 /* copy data to transfer buffer */ 335 memcpy(port->interrupt_out_buffer, 336 priv->buf + priv->cur_pos, length); 337 port->interrupt_out_urb->transfer_buffer_length = length; 338 339 priv->cur_pos = priv->cur_pos + length; 340 result = usb_submit_urb(port->interrupt_out_urb, GFP_NOIO); 341 dev_dbg(&port->dev, "%s - Send write URB returns: %i\n", __func__, result); 342 todo = priv->filled - priv->cur_pos; 343 344 if (todo > 0) 345 msleep(24); 346 } </pre>

On-Demand Memory

- Goal: providing a specification for external functions
 - Pointers returned by external function can be safely dereferenced
 - We don't lose the paths assuming that a pointer is always not null

Thank you!



Vadim Mutilin

mutilin@ispras.ru

<http://linuxtesting.org/project/ldv>

