```
Commands
        pipeline [!word...] [;]                 basic
        expr pipeline [!word...] [;]            set dot, run once
        expr, expr pipeline [!word...] [;]      set dot, repeat
        ,expr pipeline [!word...] [;]           repeat
        expr [!word...] [;]                     set dot, last pipeline, run once
        ,expr [!word...] [;]                    last pipeline, repeat
        expr, expr [!word...] [;]               set dot, last pipeline, repeat
        !word... [;]                            shell escape

Comments
        //                              Comment to end of line

Expressions
        Arithmetic
                integer                 0i binary, 0o octal, 0t decimal, 0x hex
                0t[0-9]+\.[0-9]+        IEEE floating point
                ’cccccccc’              Little-endian character const
                <identifier             variable lookup
                identifier              symbol lookup
                (expr)                  the value of expr
                .                       the value of dot
                &                       last dot used by dcmd
                +                       dot+increment
                ^                       dot-increment

                increment is effected by the last formatting dcmd.

        Unary Ops
                #expr                   logical NOT
                ˜expr                   bitwise NOT
                -expr                   integer negation
                %expr                   object file pointer dereference
                %/[csil]/expr           object file typed dereference
                %/[1248]/expr           object file sized dereference
                *expr                   virtual address pointer dereference
                */[csil]/expr           virtual address typed dereference
                */[1248]/expr           virtual address sized dereference

                [csil] is char-, short-, int-, or long-sized

        Binary Ops
                expr *  expr            integer multiplication
                expr %  expr            integer division
                left #  right           left rounded up to next right multiple
                expr +  expr            integer addition
                expr -  expr            integer subtraction
                expr << expr            bitwise left shift
                expr >> expr            bitwise right shift (logical)
                expr == expr            logical equality
                expr != expr            logical inequality
                expr &  expr            bitwise AND
                expr ^  expr            bitwise XOR
                expr |  expr            bitwise OR

Symbols
        kernel          {module‘}{file‘}symbol
        proc            {LM[0-9]+‘}{library‘}{file‘}symbol

DCMDs
        ::{module‘}d
        expr>var        write the value of expr into var
```

```
Variables
        0               Most recent value [/\?=]ed.
        9               Most recent count for $< dcmd
        b               base VA of the data section
        d               size of the data
        e               VA of entry point
        hits            Event callback match count
        m               magic number of primary object file, or zero
        t               size of text section
        thread          TID of current representative thread.

        registers are exported as variables (g0, g1, ...)

Read formats
        /               format VA from .
        \               format PA from .
        ?               format primary object file, using VA from .
        =               format value of .

        B (1)   hex                     +       dot += increment
        C (1)   char (C-encoded)        -       dot -= increment
        V (1)   unsigned                ^ (var) dot -= incr*count
        b (1)   octal                   N       newline
        c (1)   char (raw)              n       newline
        d (2)   signed                  T       tab
        h (2)   hex, swap endianness    r       whitespace
        o (2)   octal                   t       tab
        q (2)   signed octal            a       dot as symbol+offset
        u (2)   decimal                 I (var) address and instruction
        D (4)   signed                  i (var) instruction
        H (4)   hex, swap endianness    S (var) string (C-encoded)
        O (4)   octal                   s (var) string (raw)
        Q (4)   signed octal            E (8)   unsigned
        U (4)   unsigned                F (8)   double
        X (4)   hex                     G (8)   octal
        Y (4)   decoded time32_t        J (8)   hex
        f (4)   float                   R (8)   binary
        K (4|8) hex uintptr_t           e (8)   signed
        P (4|8) symbol                  g (8)   signed octal
        p (4|8) symbol                  y (8)   decoded time64_t

Write formats
        [/\?][vwWZ] value...            value is immediate or $[expr]

        /       write virtual addresses
        \       write physical addresses
        ?       write object file

        v (1)   write low byte of each value, starting at dot
        w (2)   write low 2 bytes of each value, starting at dot
        W (4)   write low 4 bytes of each value, starting at dot
        Z (8)   write all 8 bytes of each value, starting at dot

Search formats
        [/\?][lLM] value [mask]         value and mask are immediate or $[expr]

        /       search virtual addresses
        \       search physical addresses
        ?       search object file

        l (2)   search for 2-byte value, optionally masked
        L (4)   search for 4-byte value, optionally masked
        M (8)   search for 8-byte value, optionally masked
```

```
General dcmds
        ::help dcmd
                gives help text for 'dcmd'
        ::dmods -l [module...]
                Lists dcmds and walkers grouped by the dmod which provides them
        ::log -e file
                log session to file
        ::quit / $q
                quit

Target-related dcmds
        ::status
                print summary of current target
        $r / ::regs
                display current register values for target
        $c / ::stack / $C
                print current stack trace ($C: with frame pointers)
        addr[,b]::dump [-g sz] [-e]
                Dump at least b bytes starting at address addr.  -g sets
                the group size -- for 64-bit debugging, '-g 8' is useful.
        addr::dis
                dissasemble text, starting around addr.

CTF-related
        addr::print [type] [field...]
                Uses CTF info to print out a full structure, or
                particular fields thereof
        ::sizeof type / ::offsetof type field / ::enum enumname
                Get information about a type
        addr::array [type count] [var]
                Walks the count elements of an array of type 'type'
                starting at address.
        addr::list type field [var]
                Walk a circular or NULL-terminated list of type 'type',
                which starts at addr and uses 'field' as its linkage.
        ::typegraph / addr::whattype / addr::istype type / addr::notype
                bmc's type inference engine -- works on non-debug

Kernel: proc-related
        0tpid::pid2proc
                convert the process ID 'pid' (in decimal) into a proc_t ptr
        as::as2proc
                convert a 'struct as' pointer to its associated proc_t ptr
        vn::whereopen
                finds all processes with a particular vnode open
        ::pgrep pattern
                prints out proc_t ptrs which match pattern
        [procp]::ps
                process table, or (with procp) the line for particular proc_t
        ::ptree
                prints out a ptree(1)-like indented process tree
        procp::pfiles
                prints out information on a process' file descriptors

        [procp]::walk proc
                walks all processes, or the tree rooted at procp

Kernel: thread-related
        threadp::findstack
                print out a stack trace (with frame pointers) for threadp
        [threadp]::thread
                summary information about all threads or a particular thread

        [procp]::walk thread
                walk all threads, or all threads in a process (with procp)

Kernel: synchronization-related
        [sobj]::wchaninfo [-v]
                information on blocked-on condition variables.  With
                sobj, info about that wchan.  With -v, lists all threads
                blocked on the wchan.
        sobj::rwlock
                dumps out a rwlock, including detailed blocking information

        sobj::walk blocked
                walk all threads blocked on sobj, a synchronization object

Kernel: CPU-related
        ::cpuinfo [-v]
                gives information about CPUs on the system and what they
                are doing.  With '-v', shows threads on the runqueues.
        ::cpupart
                gives information about CPU partitions (psrset(1m)s)
        addr::cpuset
                prints out a cpuset as a list of included CPUs.
        [cpuid]::ttrace
                dump out traptrace records, which are generated in DEBUG
                kernels.  These include all traps and various other events of
                interest.

        ::walk cpu
                walk all cpu_ts on the system

Kernel: memory-related
        ::memstat
                Display memory usage summary
        pattern::kgrep [-d dist|-m mask|-M invmask]
                Searches the kernel heap for pointers equal to pattern
        addr::whatis [-b]
                tries to identify what a given kernel address is.  With
                '-b', gives bufctl address for the buffer (see
                $<bufctl_audit, below)

Kernel: kmem-related
        ::kmastat
                Give statistics on the kmem caches and vmem arenas in the system
        ::kmem_cache
                Information about the kmem caches on the system
        [cachep]::kmem_verify
                Validates all buffers in the system, checking for corruption.
                With cachep, shows the details of a particular cache.
        threadp::allocdby / threadp::freedby
                Shows buffers that were last allocated/freed by a particular
                thread, and are still in that state.
        ::kmalog [fail | slab]
                Dumps out the transaction log, showing recent kmem activity.
                With fail/slab, outputs records of allocation failures and
                slab creations (which are always enabled)
        ::findleaks [-dvf]
                Find memory leaks, coalesced by stack trace.
        ::bufctl [-v]
                print out a summary line for a bufctl -- can also filter them
                -v dumps out a kmem_bufctl_audit_t.

        ::walk cachename
                prints out all allocated buffers in the cache named cachename.

        [cp]::walk kmem/[cp]::walk freemem/[cp]::walk bufctl/[cp]::walk freectl
                Walks {allocated,freed}{buffers,bufctls} for all caches,
                or the particular kmem_cache_t cp.
```