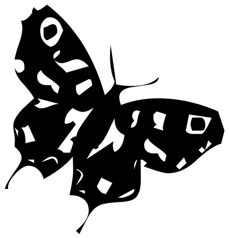
Alexander Gabert <pappy@gentoo.org> http://hardened.gentoo.org

The Hardened Gentoo toolchain

- abstract + introduction
- using the PIE/SSP toolchain
- PIE, MAC/DAC ACL kernel support
- PIE/SSP userland
- Gentoo portage
- conclusion + discussion



The Hardened Gentoo toolchain

- abstract

```
/*

    This presentation serves as an introduction to the hardened toolchain
    used at the Hardened Gentoo project, which, when combined with
    the PaX kernel, strong DAC/MAC control mechanisms and a thorough
    low-entry oriented user documentation provides extensive "full scale"
    protection for a wide range of applications and business portfolios.

*/
```

The Hardened Gentoo toolchain

- introduction
  - **who** am i**?**
    - Alexander Gabert <pappy@gentoo.org>
  - **why** are we doing it**?**
    - runtime protection and TPE (trusted path execution)
  - **what** are we changing and what is a toolchain**?**
  - gcc(cpp,cc1) + binutils(ld,as) <=> kernel + glibc(ld.so)
  - **where** is it used**?**

Adamantix * Gentoo-Hardened * Debian-Hardened

The Hardened Gentoo toolchain

- theory
  - **why do we have to change the toolchain?**
    - what is runtime executable protection?
    - why attack mitigation and trusted computing base?

The Hardened Gentoo toolchain

- reality
  - **“best of breed” technology:**
    - **PIE/SSP** toolchain -> **runtime protection**
      - PIE: Position Independent Executables
      - SSP: Stack Smashing Protection (ProPolice)
    - **PaX** kernel -> **memory protection**
      - PT_PAX support (with soft mode)
    - **MAC/DAC ACL** sub-projects -> **system integrity**
      - MAC: mandatory access control
      - DAC: discretionary access control
      - ACL: Access Control List

talking about the toolchain...

- compiler changes
  - Stack Smashing Protection: -fstack-protector-all
  - Position Independent Code/Executables: -fPIC -fPIE
  - gcc automatic specs file
- binutils and glibc modifications
  - PT_PAX segment support for the ELF header
  - __guard_setup, __guard, __stack_smash_handler in glibc
- kernel patches
  - PaX, grsecurity, LIDS, selinux, RSBAC

compiler modification #1

- SSP compiler enhancement
  - X86, AMD64, SPARC, PPC: hardware independent
  - HPPA: upgrowing stack -> no SSP available

compiler modification #1

```
    pappy@papillon examples $ gcc -S -o /dev/stdout -fstack-protector-all SSP-function.c
    .globl __stack_smash_handler       // external declaration of __stack_smash_handler function

    --->> entering function

         FUNCTION PROLOGUE             // before the function initially starts, we set up the guard

    movl __guard, %eax            // move(long word) the __guard to %eax
    movl %eax, -24(%ebp)               // move the __guard value to the appropriate STACK position

    --->> function executes with normal behaviour

         FUNCTION EPILOGUE             // at the end of the function, the epilogue checks the guard

    cmpl __guard, %edx            // compare the __guard
    je   .L2                      // jump to .L2 if compare shows equal
                                  // __guard and position on stack
    movl -24(%ebp), %eax               // debug information for __stack_smash_handler
    movl %eax, 4(%esp)            // more debug
    movl $.LC0, (%esp)            // yet more debug
    call __stack_smash_handler         // exit the function via __stack_smash_handler()
.L2:                              // continue normal execution
    --->> leaving function
```

compiler modification #2

- PIC/PIE compiler enhancement
  - X86: speed penalty (register shortage, multimedia asm)
  - AMD64: native PIC architecture
  - SPARC, HPPA, PPC: RISC: special PIC/DATA register support
  - fPIC (symbol visibility, addressing in main executable via GOT, PLT)
  - fPIE (symbol addressing in main executable via local relative text segment)

GOT: Global Offset Table (for symbols referring data)
PLT: Process Linkage Table (for symbols referring functions)

compiler modification #2

```
pappy@papillon examples $ gcc -fPIC -S -o /dev/stdout PIC-function.c

--->> entering function

     FUNCTION PROLOGUE             // before the function initially starts, set up PIC register

   call    __i686.get_pc_thunk.bx            // call the PIC setup (get Program Counter)
                                        // the PIC setup function stores EIP in EBX
   addl    $_GLOBAL_OFFSET_TABLE_, %ebx      // calculate distance to GOT using the current EBX

--->> function executes and returns with normal behaviour

/* definition of PIC setup function */
__i686.get_pc_thunk.bx:                      // we have been called, EIP was pushed on the stack
   movl    (%esp), %ebx                      // move the EIP on the stack to %ebx
   ret                                       // go back to the calling function

see for references:
     "The Levine book" Linkers & Loaders (John R. Levine) ISBN 1-55860-496-0 pp. 169
```

compiler modification #3

- gcc specs compiler modification
  - automatic, invisible introduction of PIC/PIE/SSP flags
  - automatic reaction to “suppression” (ASM, GLIBC, GCC)
  - multiple architecture/version support
  - small set of specific changes to internal gcc source code
  - portable, small code base for hardening patch

`See “The Specs Language” in /var/tmp/portage/gcc-3.4.2-r2/work/gcc-3.4.2/gcc/gcc.c`

compiler modification #3

- gcc specs compiler modification

```
*cc1:
%(cc1_cpu) %{profile:-p} %{!m64: %{!msse2:-mno-sse2} }

%{!D__KERNEL__:

%{!static: %{!fno-PIC: %{!fno-pic: %{!shared: %{!nostdlib: %{!nostartfiles:
%{!fno-PIE: %{!fno-pie: %{!nopie: %{!fPIC:%{!fpic:

                                       -fPIE

} } } } }
} } } } } }

%{!nostdlib:
    %{!fno-stack-protector:            -fstack-protector
%{!D_LIBC: %{!D_LIBC_REENTRANT:
    %{!fno-stack-protector-all:        -fstack-protector-all
} } } } }

}
```

See “The Specs Language” in /var/tmp/portage/gcc-3.4.2-r2/work/gcc-3.4.2/gcc/gcc.c

compiler modification #3

- gcc specs compiler modification

```
*endfile:
%{shared:crtendS.o%s;static:crtend.o%s;nopie:crtend.o%s;:crtendS.o%s} crtn.o%s

*startfile:
%{!shared: %{pg|p|profile:gcrt1.o%s;static:crt1.o%s;nopie:crt1.o%s;:Scrt1.o%s}}    crti.o%s
%{shared:crtbeginS.o%s;static:crtbeginT.o%s;nopie:crtbegin.o%s;:crtbeginS.o%s}
```

See “The Specs Language” in /var/tmp/portage/gcc-3.4.2-r2/work/gcc-3.4.2/gcc/gcc.c

compiler modification #3

- gcc specs compiler modification

```
*link_command:
%{!fsyntax-only:%{!c:%{!M:%{!MM:%{!E:%{!S:    %(linker) %l

%{!nopie: %{!static: %{!A: %{!i: %{!r: %{!Bstatic: %{!shared: %{!nostdlib: %{!nostartfiles:
%{!fno-PIE: %{!fno-pie:

                                -pie

} } } } } } } } } } }

%{pie: } %{!norelro:

                                -z relro

}
%{relro: } %{!nonow:

                                -z now

} %{now: }
```

See "The Specs Language" in /var/tmp/portage/gcc-3.4.2-r2/work/gcc-3.4.2/gcc/gcc.c

compiler modification #3

- gcc specs compiler modification
  - precise working set of PIC/PIE and SSP operations
  - transparent build object file reordering and replacement
  - all features and support by Peter S. Mazinger (Hungary)

gcc version 3.4.2 (Gentoo Hardened Linux 3.4.2-r2, ssp-3.4.1-1, pie-8.7.6.5)

binutils/glibc modifications

- binutils and glibc modifications
  - PT_PAX segment support (ELF header)
  - __guard_setup, __guard, __stack_smash_handler in glibc
  - ATTN! race condition: libraries versus main executable
    - prologue:
      copy __guard@@EXECUTABLE to guard stack location
    - epilogue:
      compare guard stack location to __guard@@LIBRARY
  - TODO: improve entropy generation for __guard_setup
  - TODO: separate libssp.so independent of GNU libc

kernel improvements #1

- kernel patches
  - PaX and grsecurity: PIE and MAC-ACL
    - PIE: process segment randomization

```
08048000-0804c000 r-xp 00000000 09:00 15280      /bin/cat
0804c000-0804d000 rw-p 00003000 09:00 15280      /bin/cat
0804d000-0806e000 rwxp 00000000 00:00 0
40000000-40016000 r-xp 00000000 09:00 92276      /lib/ld-2.3.4.so
40016000-40017000 rw-p 00015000 09:00 92276      /lib/ld-2.3.4.so
40023000-40024000 rw-p 00000000 00:00 0
40024000-4012b000 r-xp 00000000 09:00 91807      /lib/libc-2.3.4.so
4012b000-4012e000 rw-p 00106000 09:00 91807      /lib/libc-2.3.4.so
4012e000-40131000 rw-p 00000000 00:00 0
bfffe000-c0000000 rwxp fffff000 00:00 0
```

    - SSP as additional in-depth defense

kernel improvements #2

- additional kernel patches
  - complementary MAC systems:
    - LIDS (Linux Intrusion Detection System)
    - Selinux (Gentoo Hardened)
    - RSBAC (Adamantix)
  - interchangeable with PIE/SSP solution
  - cascading security model

Hardened PIE/SSP userland

- I) default PIE/SSP gcc specs file
  - least intrusive approach
  - no modification of normal packages
  - failing packages add filter-flags logic
- II) PaX kernel and glibc
- III) MAC/DAC ACL coverage

Gentoo Portage

- flag-o-matic.eclass
  - ebuilds: “filter-flags”
  - SSP and PIE filter arguments
  - etc/make.conf:CFLAGS
  - eclass abstraction layer
  - common suppression flag

pitfalls

- multimedia and bootloaders:
  - mplayer, xine, grub, XFree86, Xorg

```
#if defined ( __PIC__ || __pic__ )
    PIC_VERSION
#else
    ASM_VERSION
#endif
```

The Hardened Gentoo toolchain

- conclusion
  - security is a process, not a solution
    - keep up with patches and security fixes
    - stay informed about new vulnerabilities
  - easy adoption of core technology
    - important for progress of the solution
    - secure design by default!
    - drawback: costs = investment + maintenance
  - **Open Source**: developers welcome!

The Hardened Gentoo toolchain

- discussion
  - feel free to ask questions

```
        .file   "hello.c"
.globl __stack_smash_handler
        .section        .rodata
.LC0:
        .string "hello\n"
.LC1:
        .string "main"
        .text
.globl main
        .type   main, @function
main:
        pushl   %ebp
        movl    %esp, %ebp
        pushl   %ebx
        subl    $36, %esp
        call    __i686.get_pc_thunk.bx                          // prologue: PIC register setup (%ebx)
        addl    $_GLOBAL_OFFSET_TABLE_, %ebx
        andl    $-16, %esp
        movl    $0, %eax
        addl    $15, %eax
        addl    $15, %eax
        shrl    $4, %eax
        sall    $4, %eax
        subl    %eax, %esp
        movl    __guard@GOT(%ebx), %eax                     // prologue: store __guard on stack
        movl    (%eax), %eax
        movl    %eax, -24(%ebp)
        leal    .LC0@GOTOFF(%ebx), %eax
        movl    %eax, (%esp)
        call    printf@PLT
        movl    $0, %eax
        movl    __guard@GOT(%ebx), %edx                     // epilogue: compare __guard to stack value
        movl    (%edx), %edx
        cmpl    %edx, -24(%ebp)
        je      .L2
        movl    -24(%ebp), %eax
        movl    %eax, 4(%esp)
        leal    .LC1@GOTOFF(%ebx), %eax
        movl    %eax, (%esp)
        call    __stack_smash_handler@PLT
.L2:
        movl    -4(%ebp), %ebx
        leave
        ret
        .size   main, .-main
        .section        .gnu.linkonce.t.__i686.get_pc_thunk.bx,"ax",@progbits
.globl __i686.get_pc_thunk.bx
        .hidden __i686.get_pc_thunk.bx
        .type   __i686.get_pc_thunk.bx, @function
__i686.get_pc_thunk.bx:
        movl    (%esp), %ebx
        ret
        .section        .note.GNU-stack,"",@progbits
        .ident  "GCC: (GNU) 3.4.2  (Gentoo Hardened Linux 3.4.2-r2, ssp-3.4.1-1, pie-8.7.6.5)"
```

```
 10:24:52 [/space/chroots/chroot002:2229.pts-0.papillon]papillon ~
 # gcc -v
Reading specs from /usr/lib/gcc/i686-pc-linux-gnu/3.4.2/specs
Reading specs from /usr/lib/gcc-lib/i686-pc-linux-gnu/3.4.2/specs
Configured with: /var/tmp/portage/gcc-3.4.2-r2/work/gcc-3.4.2/configure --enable-version-specific-runtime-libs --prefix=/usr --bindir=/usr/i686-pc-linux-gnu/gcc-bin/3.4 --
includedir=/usr/lib/gcc/i686-pc-linux-gnu/3.4.2/include --datadir=/usr/share/gcc-data/i686-pc-linux-gnu/3.4 --mandir=/usr/share/gcc-data/i686-pc-linux-gnu/3.4/man --
infodir=/usr/share/gcc-data/i686-pc-linux-gnu/3.4/info --with-gxx-include-dir=/usr/lib/gcc/i686-pc-linux-gnu/3.4.2/include/g++-v3 --host=i686-pc-linux-gnu --enable-nls --
without-included-gettext --enable-__cxa_atexit --enable-clocale=gnu --enable-shared --with-system-zlib --disable-checking --disable-werror --disable-libunwind-
exceptions --with-gnu-ld --enable-threads=posix --disable-multilib --enable-languages=c,c++,f77,objc,java
Thread model: posix
gcc version 3.4.2  (Gentoo Hardened Linux 3.4.2-r2, ssp-3.4.1-1, pie-8.7.6.5)


 10:24:53 [/space/chroots/chroot002:2229.pts-0.papillon]papillon ~
 # readelf -s hello | egrep "__guard|__guard_setup|__stack_smash_handler"
    19: 00000000     4 OBJECT  GLOBAL DEFAULT  UND __guard@GLIBC_2.3.2 (4)
    21: 00000000   720 FUNC    GLOBAL DEFAULT  UND __stack_smash_handler@GLIBC_2.3.2 (4)
   109: 00000000     4 OBJECT  GLOBAL DEFAULT  UND __guard@@GLIBC_2.3.2
   111: 00000000   720 FUNC    GLOBAL DEFAULT  UND __stack_smash_hahndler@@GL


 10:25:12 [/space/chroots/chroot002:2229.pts-0.papillon]papillon ~
 # readelf -s /lib/libc-2.3.4.so | egrep "__guard|__guard_setup|__stack_smash_handler"
   285: 000153bf   720 FUNC    GLOBAL DEFAULT   11 __stack_smash_handler@@GLIBC_2.3.2
   702: 00108324     4 OBJECT  GLOBAL DEFAULT   29 __guard@@GLIBC_2.3.2
   865: 0001531c   163 FUNC    GLOBAL DEFAULT   11 __guard_setup@@GLIBC_2.3.2
  7163: 000153bf   720 FUNC    GLOBAL DEFAULT   11 __stack_smash_handler
  7580: 00108324     4 OBJECT  GLOBAL DEFAULT   29 __guard
  7743: 0001531c   163 FUNC    GLOBAL DEFAULT   11 __guard_setup


 # rm ./hello; CFLAGS="-static" make hello
gcc -static    hello.c   -o hello
 10:29:17 [/space/chroots/chroot002:2229.pts-0.papillon]papillon ~
 # readelf -s hello | egrep "__guard|__guard_setup|__stack_smash_handler"
  1218: 08048a09   679 FUNC    GLOBAL DEFAULT    2 __stack_smash_handler
  1409: 080ac114     4 OBJECT  GLOBAL DEFAULT   15 __guard
  1481: 0804897c   141 FUNC    GLOBAL DEFAULT    2 __guard_setup
```

```
        .file   "libssp.c"
.globl __guard
        .data
        .align 4
        .type   __guard, @object
        .size   __guard, 4
__guard:
        /* stock value: 0xFEEDFEED */
        .long   -17957139
        .text
.globl __guard_setup
        .type   __guard_setup, @function
__guard_setup:
        pushl   %ebp
        movl    %esp, %ebp
        /* initialized value: 0xDEADBEEF */
        movl    $-559038737, __guard
        popl    %ebp
        ret
        .size   __guard_setup, .-__guard_setup
.globl __stack_smash_handler
        .type   __stack_smash_handler, @function
__stack_smash_handler:
        pushl   %ebp
        movl    %esp, %ebp
        nop
.L3:
        jmp     .L3
        .size   __stack_smash_handler, .-__stack_smash_handler
        .section        .note.GNU-stack,"",@progbits
        .ident  "GCC: (GNU) 3.3.4 20040623 (Gentoo Linux 3.3.4-r1, ssp-3.3.2-2, pie-8.7.6)"
```

```
---- libssp-x86.s       2004-12-06 22:26:57.801893232 +0100
+++ libssp-x86-PIC.s    2004-12-06 22:29:05.778437840 +0100
@@ -13,8 +13,11 @@
 __guard_setup:
        pushl   %ebp
        movl    %esp, %ebp
+       call    __i686.get_pc_thunk.cx
+       addl    $_GLOBAL_OFFSET_TABLE_, %ecx
+       movl    __guard@GOT(%ecx), %eax
        /* initialized value: 0xDEADBEEF */
-       movl    $-559038737, __guard
+       movl    $-559038737, (%eax)
        popl    %ebp
        ret
        .size   __guard_setup, .-__guard_setup

@@ -27,5 +30,12 @@
 .L3:
        jmp     .L3
        .size   __stack_smash_handler, .-__stack_smash_handler
+       .section        .gnu.linkonce.t.__i686.get_pc_thunk.cx,"ax",@progbits
+.globl __i686.get_pc_thunk.cx
+       .hidden __i686.get_pc_thunk.cx
+       .type   __i686.get_pc_thunk.cx, @function
+__i686.get_pc_thunk.cx:
+       movl    (%esp), %ecx
+       ret
        .section        .note.GNU-stack,"",@progbits
        .ident  "GCC: (GNU) 3.3.4 20040623 (Gentoo Linux 3.3.4-r1, ssp-3.3.2-2, pie-8.7.6)"
```

useful for long gdb sessions: dummy LD_PRELOADable libssp.so:

```
22:40 pappy@papillon pappy $ cat /space/chroots/master/tmp/libssp.c

unsigned long __guard = 0xFEEDFEEDUL;
void __guard_setup (void) { __guard = 0xDEADBEEFUL; }
void __stack_smash_handler (char func[], int damaged) { for(;;) {;} }
```