ARO Planning Workshop Security of Embedded Systems and Networks

Hardware-Based Security: Trouble and Hope

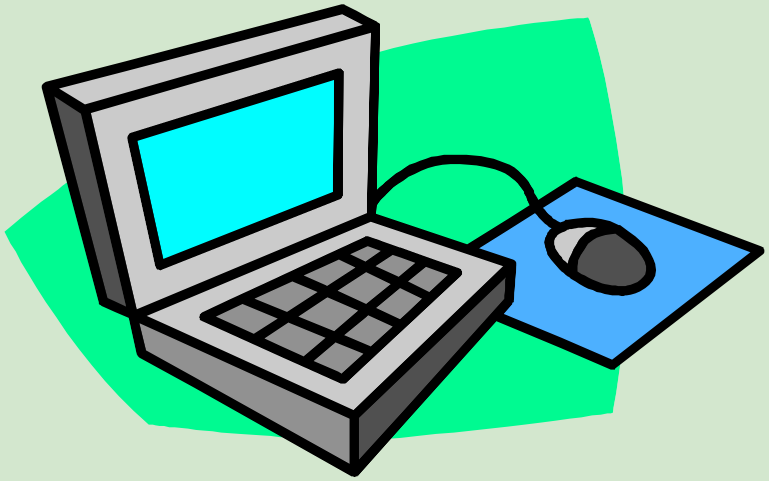
Sean W. Smith
Department of Computer Science
Dartmouth College
Hanover, NH USA

http://www.cs.dartmouth.edu/~sws/

February 22, 2007

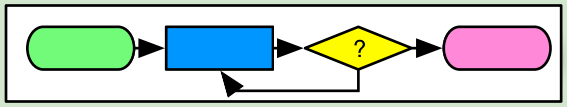
Trouble Area #1:

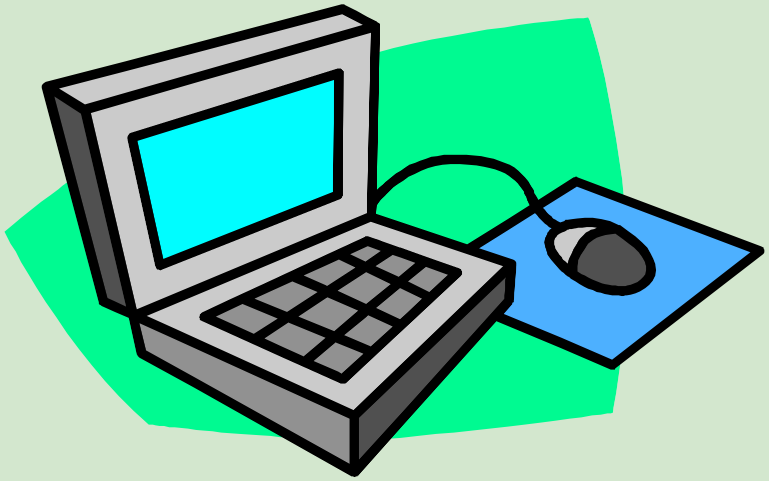
Not Thinking about Enough Levels



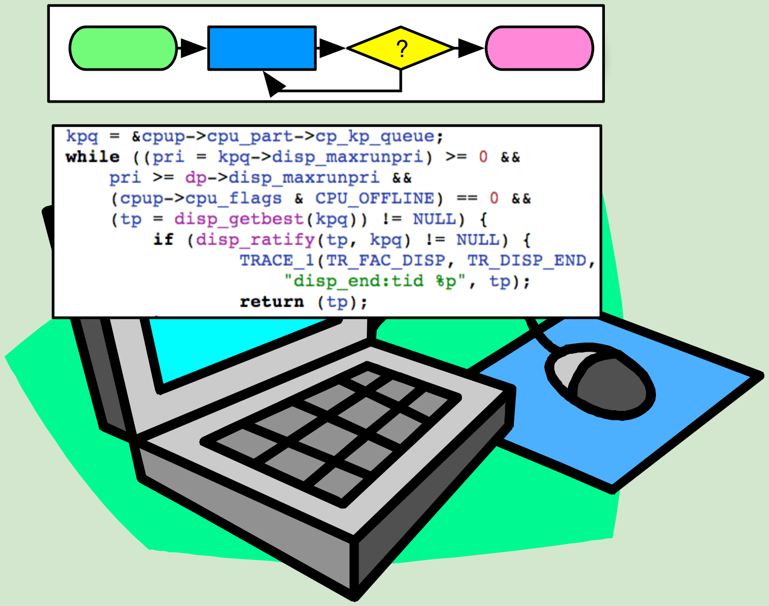
Trouble Area #1:

Not Thinking about Enough Levels





Trouble Area #1:
Not Thinking about Enough Levels

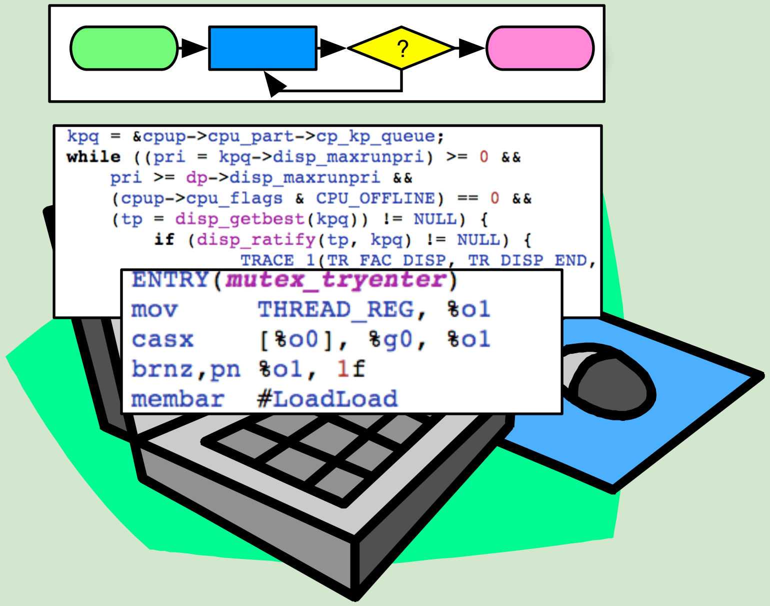


```
kpq = &cpup->cpu_part->cp_kp_queue;
while ((pri = kpq->disp_maxrunpri) >= 0 &&
    pri >= dp->disp_maxrunpri &&
    (cpup->cpu_flags & CPU_OFFLINE) == 0 &&
    (tp = disp_getbest(kpq)) != NULL) {
        if (disp_ratify(tp, kpq) != NULL) {
                TRACE_1(TR_FAC_DISP, TR_DISP_END,
                    "disp_end:tid %p", tp);
                return (tp);
```

Vox Clamantis in Deserto

Trouble Area #1:

Not Thinking about Enough Levels



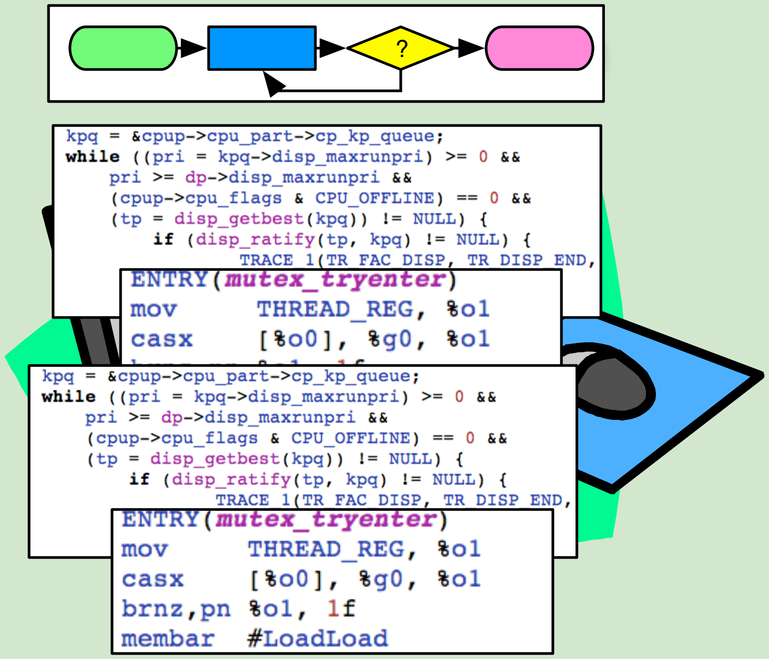
```
kpq = &cpup->cpu_part->cp_kp_queue;
while ((pri = kpq->disp_maxrunpri) >= 0 &&
    pri >= dp->disp_maxrunpri &&
    (cpup->cpu_flags & CPU_OFFLINE) == 0 &&
    (tp = disp_getbest(kpq)) != NULL) {
        if (disp_ratify(tp, kpq) != NULL) {
                TRACE_1(TR_FAC_DISP, TR_DISP_END,
```

```
ENTRY(mutex_tryenter)
mov       THREAD_REG, %o1
casx      [%o0], %g0, %o1
brnz,pn %o1, 1f
membar    #LoadLoad
```

Vox Clamantis in Deserto

Trouble Area #1:

<u>Not Thinking about Enough Levels</u>



```
kpq = &cpup->cpu_part->cp_kp_queue;
while ((pri = kpq->disp_maxrunpri) >= 0 &&
    pri >= dp->disp_maxrunpri &&
    (cpup->cpu_flags & CPU_OFFLINE) == 0 &&
    (tp = disp_getbest(kpq)) != NULL) {
        if (disp_ratify(tp, kpq) != NULL) {
                TRACE_1(TR_FAC_DISP, TR_DISP_END,
```

```
ENTRY(mutex_tryenter)
mov      THREAD_REG, %o1
casx     [%o0], %g0, %o1
```

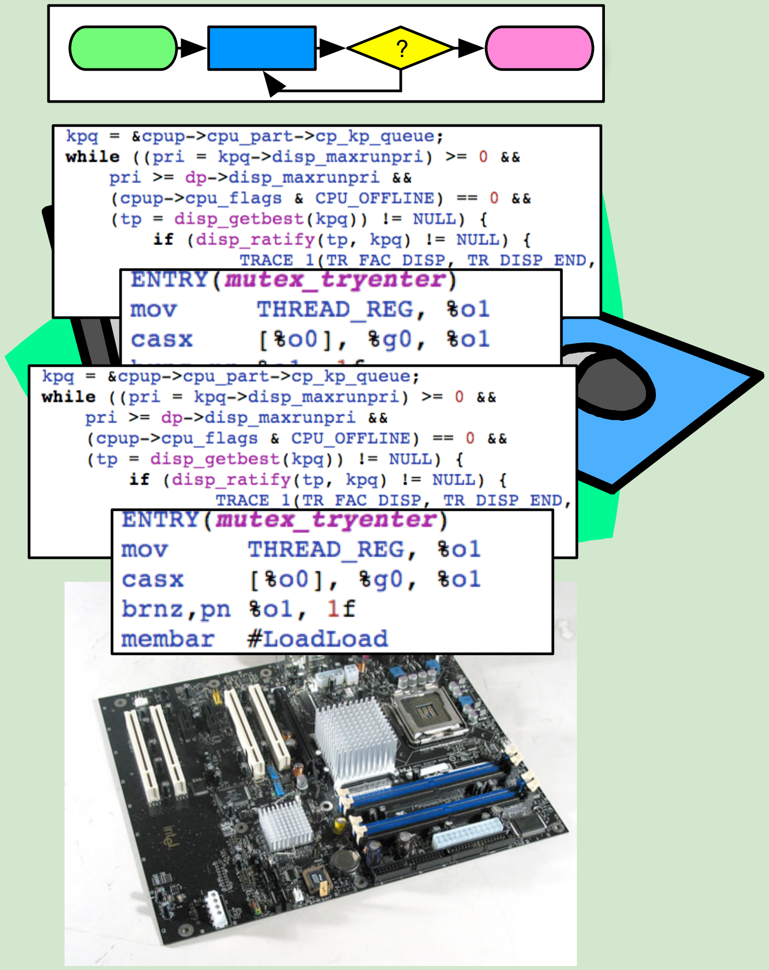
```
kpq = &cpup->cpu_part->cp_kp_queue;
while ((pri = kpq->disp_maxrunpri) >= 0 &&
    pri >= dp->disp_maxrunpri &&
    (cpup->cpu_flags & CPU_OFFLINE) == 0 &&
    (tp = disp_getbest(kpq)) != NULL) {
        if (disp_ratify(tp, kpq) != NULL) {
                TRACE_1(TR_FAC_DISP, TR_DISP_END,
```

```
ENTRY(mutex_tryenter)
mov      THREAD_REG, %o1
casx     [%o0], %g0, %o1
brnz,pn %o1, 1f
membar   #LoadLoad
```

Vox Clamantis in Deserto

Trouble Area #1:

Not Thinking about Enough Levels



```
kpq = &cpup->cpu_part->cp_kp_queue;
while ((pri = kpq->disp_maxrunpri) >= 0 &&
    pri >= dp->disp_maxrunpri &&
    (cpup->cpu_flags & CPU_OFFLINE) == 0 &&
    (tp = disp_getbest(kpq)) != NULL) {
        if (disp_ratify(tp, kpq) != NULL) {
                TRACE_1(TR_FAC_DISP, TR_DISP_END,
```

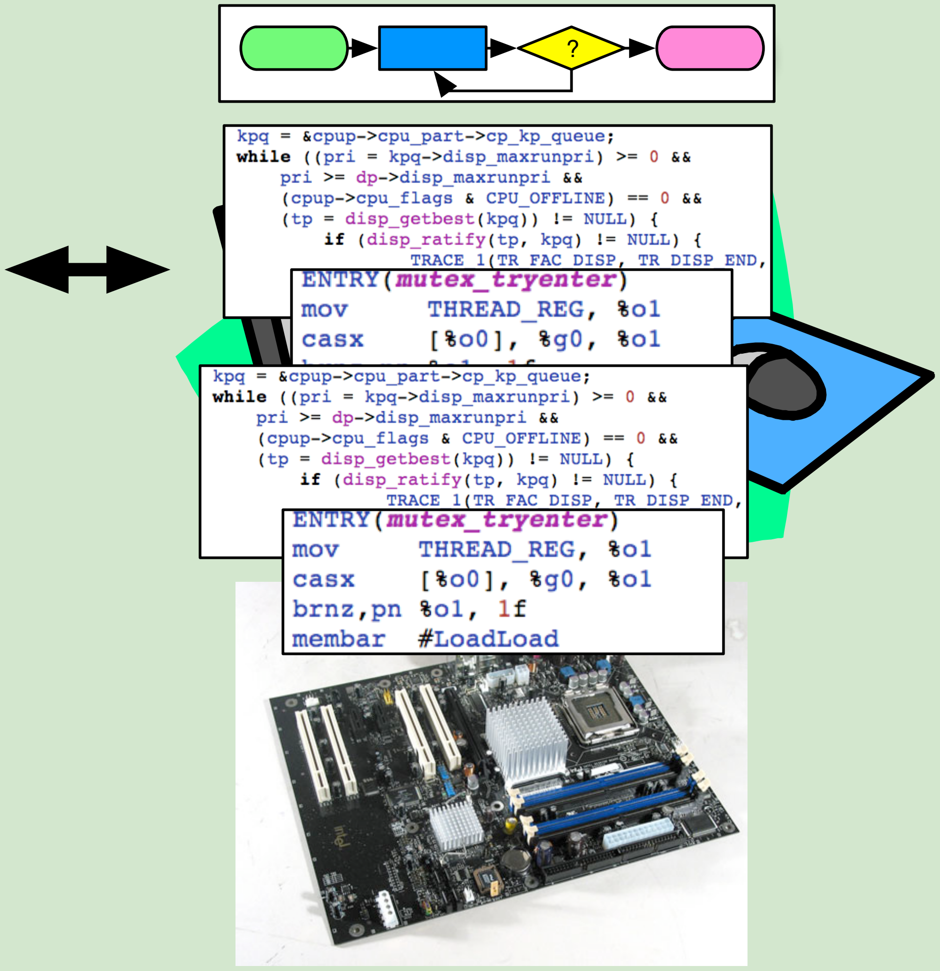
```
ENTRY(mutex_tryenter)
mov       THREAD_REG, %o1
casx      [%o0], %g0, %o1
```

```
kpq = &cpup->cpu_part->cp_kp_queue;
while ((pri = kpq->disp_maxrunpri) >= 0 &&
    pri >= dp->disp_maxrunpri &&
    (cpup->cpu_flags & CPU_OFFLINE) == 0 &&
    (tp = disp_getbest(kpq)) != NULL) {
        if (disp_ratify(tp, kpq) != NULL) {
                TRACE_1(TR_FAC_DISP, TR_DISP_END,
```

```
ENTRY(mutex_tryenter)
mov       THREAD_REG, %o1
casx      [%o0], %g0, %o1
brnz,pn   %o1, 1f
membar    #LoadLoad
```

Trouble Area #1:

Not Thinking about Enough Levels



```
kpq = &cpup->cpu_part->cp_kp_queue;
while ((pri = kpq->disp_maxrunpri) >= 0 &&
    pri >= dp->disp_maxrunpri &&
    (cpup->cpu_flags & CPU_OFFLINE) == 0 &&
    (tp = disp_getbest(kpq)) != NULL) {
        if (disp_ratify(tp, kpq) != NULL) {
                TRACE_1(TR_FAC_DISP, TR_DISP_END,
```

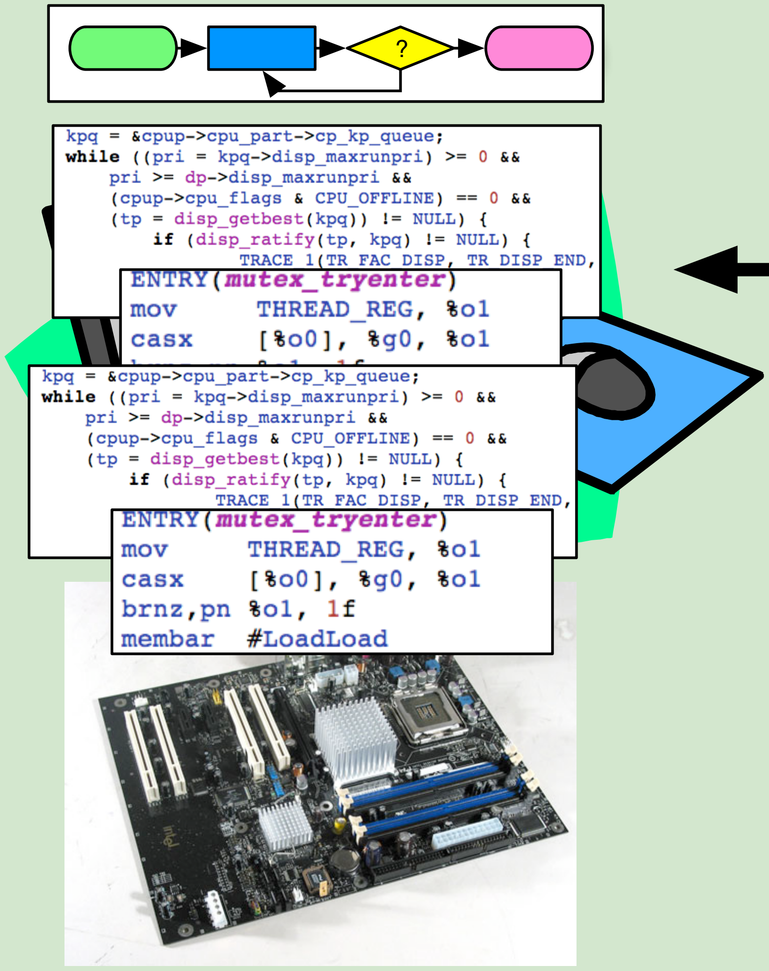
```
ENTRY(mutex_tryenter)
mov       THREAD_REG, %o1
casx      [%o0], %g0, %o1
```

```
kpq = &cpup->cpu_part->cp_kp_queue;
while ((pri = kpq->disp_maxrunpri) >= 0 &&
    pri >= dp->disp_maxrunpri &&
    (cpup->cpu_flags & CPU_OFFLINE) == 0 &&
    (tp = disp_getbest(kpq)) != NULL) {
        if (disp_ratify(tp, kpq) != NULL) {
                TRACE_1(TR_FAC_DISP, TR_DISP_END,
```

```
ENTRY(mutex_tryenter)
mov       THREAD_REG, %o1
casx      [%o0], %g0, %o1
brnz,pn %o1, 1f
membar    #LoadLoad
```

Trouble Area #1:

Not Thinking about Enough Levels



```
kpq = &cpup->cpu_part->cp_kp_queue;
while ((pri = kpq->disp_maxrunpri) >= 0 &&
    pri >= dp->disp_maxrunpri &&
    (cpup->cpu_flags & CPU_OFFLINE) == 0 &&
    (tp = disp_getbest(kpq)) != NULL) {
        if (disp_ratify(tp, kpq) != NULL) {
                TRACE_1(TR_FAC_DISP, TR_DISP_END,
```

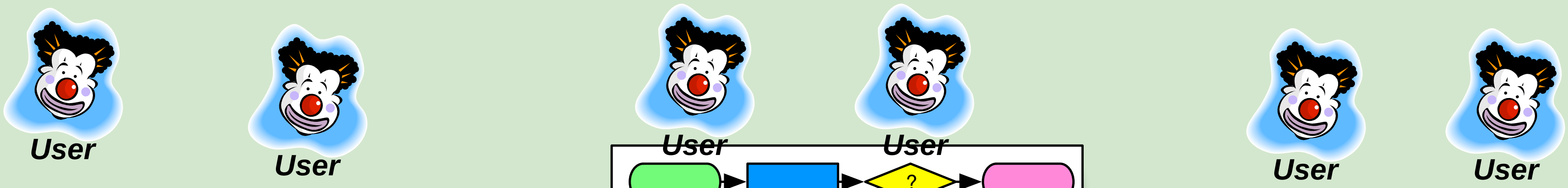
```
ENTRY(mutex_tryenter)
mov      THREAD_REG, %o1
casx     [%o0], %g0, %o1
```

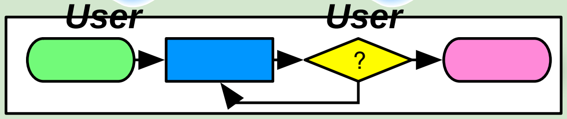
```
kpq = &cpup->cpu_part->cp_kp_queue;
while ((pri = kpq->disp_maxrunpri) >= 0 &&
    pri >= dp->disp_maxrunpri &&
    (cpup->cpu_flags & CPU_OFFLINE) == 0 &&
    (tp = disp_getbest(kpq)) != NULL) {
        if (disp_ratify(tp, kpq) != NULL) {
                TRACE_1(TR_FAC_DISP, TR_DISP_END,
```

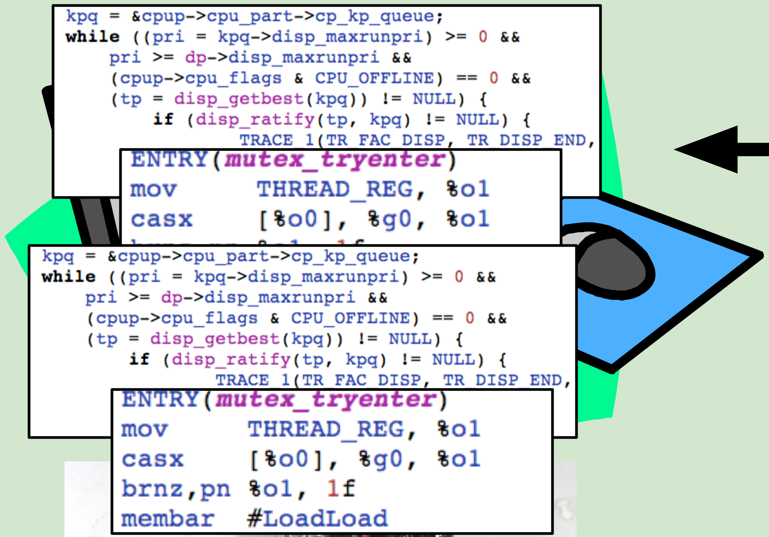
```
ENTRY(mutex_tryenter)
mov      THREAD_REG, %o1
casx     [%o0], %g0, %o1
brnz,pn  %o1, 1f
membar   #LoadLoad
```

Trouble Area #1:

Not Thinking about Enough Levels





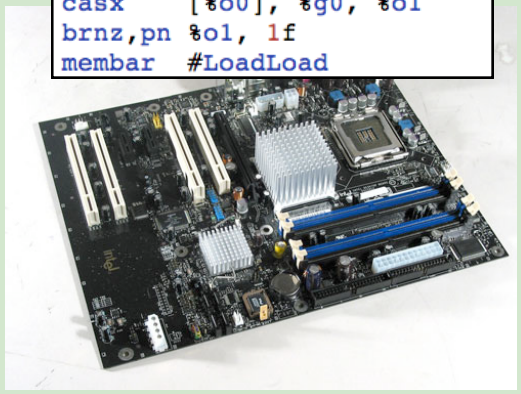


```
kpq = &cpup->cpu_part->cp_kp_queue;
while ((pri = kpq->disp_maxrunpri) >= 0 &&
    pri >= dp->disp_maxrunpri &&
    (cpup->cpu_flags & CPU_OFFLINE) == 0 &&
    (tp = disp_getbest(kpq)) != NULL) {
        if (disp_ratify(tp, kpq) != NULL) {
                TRACE_1(TR_FAC_DISP, TR_DISP_END,
```

```
ENTRY(mutex_tryenter)
mov     THREAD_REG, %o1
casx    [%o0], %g0, %o1
```

```
kpq = &cpup->cpu_part->cp_kp_queue;
while ((pri = kpq->disp_maxrunpri) >= 0 &&
    pri >= dp->disp_maxrunpri &&
    (cpup->cpu_flags & CPU_OFFLINE) == 0 &&
    (tp = disp_getbest(kpq)) != NULL) {
        if (disp_ratify(tp, kpq) != NULL) {
                TRACE_1(TR_FAC_DISP, TR_DISP_END,
```

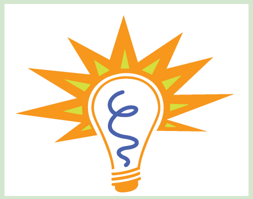
```
ENTRY(mutex_tryenter)
mov     THREAD_REG, %o1
casx    [%o0], %g0, %o1
brnz,pn %o1, 1f
membar  #LoadLoad
```



Vox Clamantis in Deserto

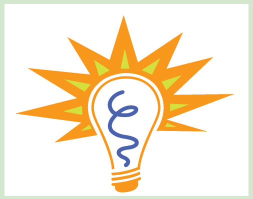
Examples

- Java type-safety vs. light bulbs



Examples

- Java type-safety vs. light bulbs



- Type-safe C variant for kernel coding vs. kernel coders

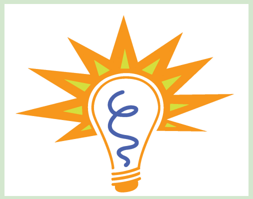
Examples

- Java type-safety vs. light bulbs

- Type-safe C variant for kernel coding vs. kernel coders
- hardware-based attestation vs. the computational entity
- hardware-based attestation vs. the operating system

Examples

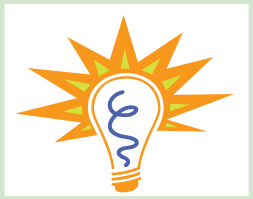
- Java type-safety vs. light bulbs



- Type-safe C variant for kernel coding vs. kernel coders
- hardware-based attestation vs. the computational entity
- hardware-based attestation vs. the operating system

- IBM 4758 platform vs. API flaws in the CCA app

Examples



- Java type-safety vs. light bulbs
- Type-safe C variant for kernel coding vs. kernel coders
- hardware-based attestation vs. the computational entity
- hardware-based attestation vs. the operating system

- IBM 4758 platform vs. API flaws in the CCA app
- Cyber-Manhattan project vs. economic roll-out

More Trouble Areas

2. Cryptography's questionable future

More Trouble Areas

2. Cryptography's questionable future

More Trouble Areas

2. Cryptography's questionable future

3. Keeping and using secrets

- *effectively*
- *affordably*

More Trouble Areas

2. *Cryptography's questionable future*

3. *Keeping and using secrets*

- *effectively*
- *affordably*

Reasons for Hope

1. Industry is open to designing ***and deploying*** hardware-based techniques to enhance security.

TRUSTED COMPUTING GROUP™

Reasons for Hope

1. Industry is open to designing ***and deploying*** hardware-based techniques to enhance security.

TRUSTED COMPUTING GROUP™

Reasons for Hope

1. Industry is open to designing ***and deploying*** hardware-based techniques to enhance security.

TRUSTED COMPUTING GROUP™

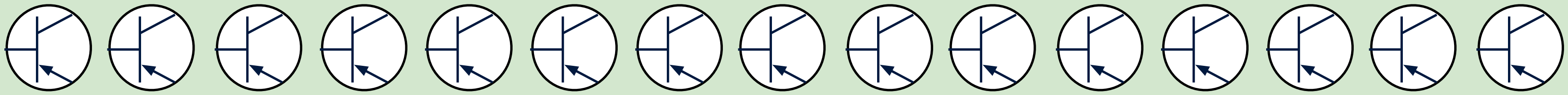
2. The consequences of ***keeping up*** with Moore's Law

Reasons for Hope

1. Industry is open to designing ***and deploying*** hardware-based techniques to enhance security.

TRUSTED COMPUTING GROUP™

2. The consequences of ***keeping up*** with Moore's Law



Reasons for Hope

1. Industry is open to designing ***and deploying*** hardware-based techniques to enhance security.

2. The consequences of ***keeping up*** with Moore's Law

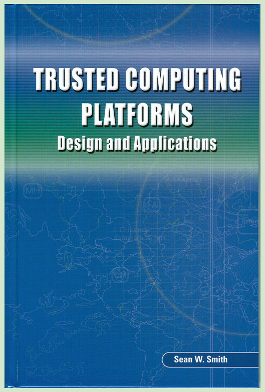
3. Repeated calls for ***principled revolution***

- CRA, I3P, "Cyber-Manhattan Project,"....
- ***This workshop***

Thanks

Sponsors:

- NSF CAREER, DoJ/DHS, Mellon, Internet2/AT&T
- Sun, Intel, Cisco (and IBM Research)



For more information:

- http://www.cs.dartmouth.edu/~sws/
- *Trusted Computing Platforms: Design and Applications*. Springer, 2005.