Entrapment: Tricking Malware with Transparent, Scalable Malware Analysis

Paul Royal

paul@gtisc.gatech.edu

Agenda

- Modern Malware
  - Obfuscations, Server-side Polymorphism, Collection Volume
- Malware Analysis Detection
  - Commoditization, Popularity, Transparency
  - Detecting QEMU, VMware, KVM
- Baremetal Malware Analysis
  - Hardware, Technologies
  - Non-Virtual Machine Trace (NVMTrace), a PoC Baremetal Malware Analysis Tool
- Conclusion/Future Work

Modern Malware

Modern Malware

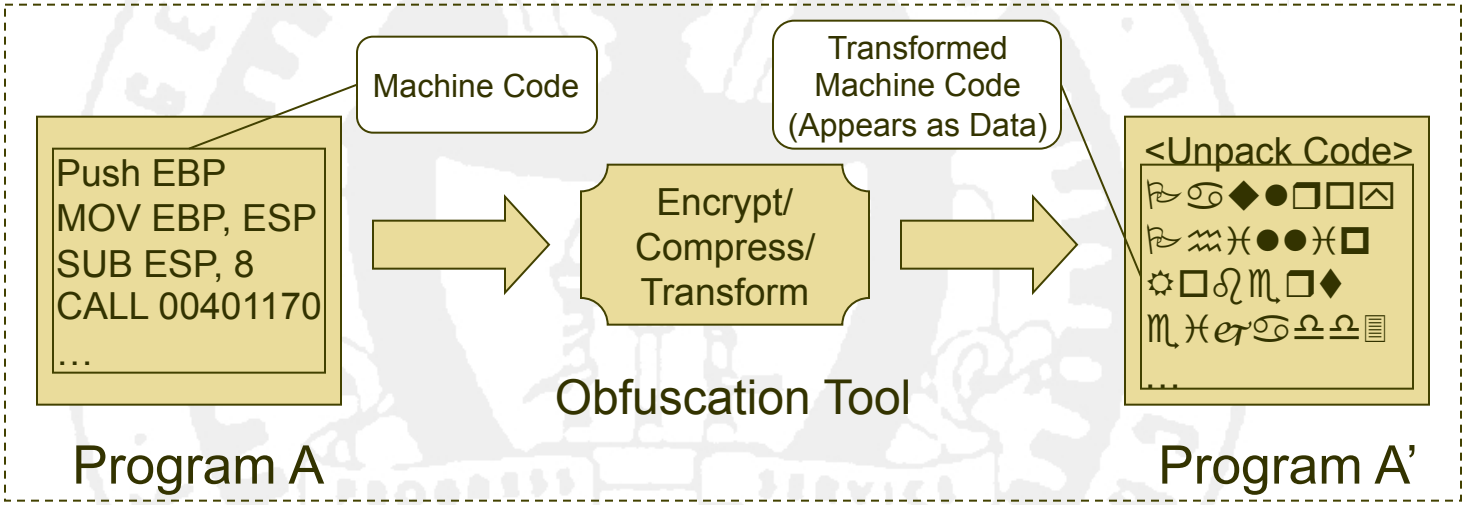
- **The centerpiece of current threats on the Internet**
  - **Botnets (spamming, DDOS, etc.)**
  - **Information Theft**
  - **Financial Fraud**
- **Used by Real Criminals**
  - **Criminal Infrastructure**
  - **Domain of Organized Crime**

Malware Cont'd

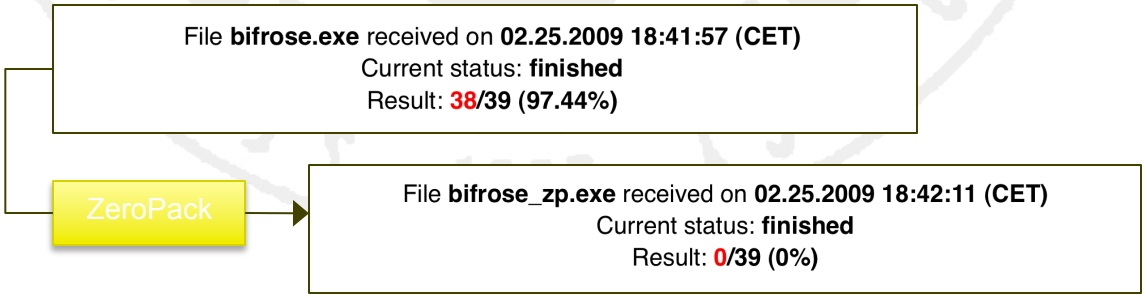
- **There is a pronounced need to understand malware behavior**
  - **Threat Discovery and Analysis**
  - **Compromise Detection**
  - **Forensics and Asset Remediation**
- **Malware authors make analysis challenging**
  - **Direct financial motivation**

Malware Obfuscations

- **Pictorial Overview**

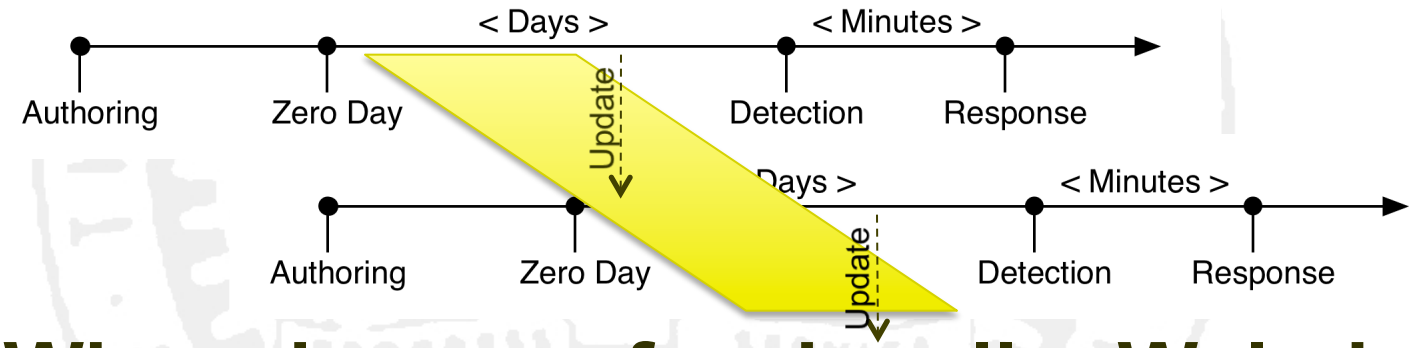


- **Project ZeroPack**



Obfuscations Cont'd

- **Server-side Polymorphism**
  - **Automate mutations**



- **When done professionally: Waledac**

Collected on 12/30/2008

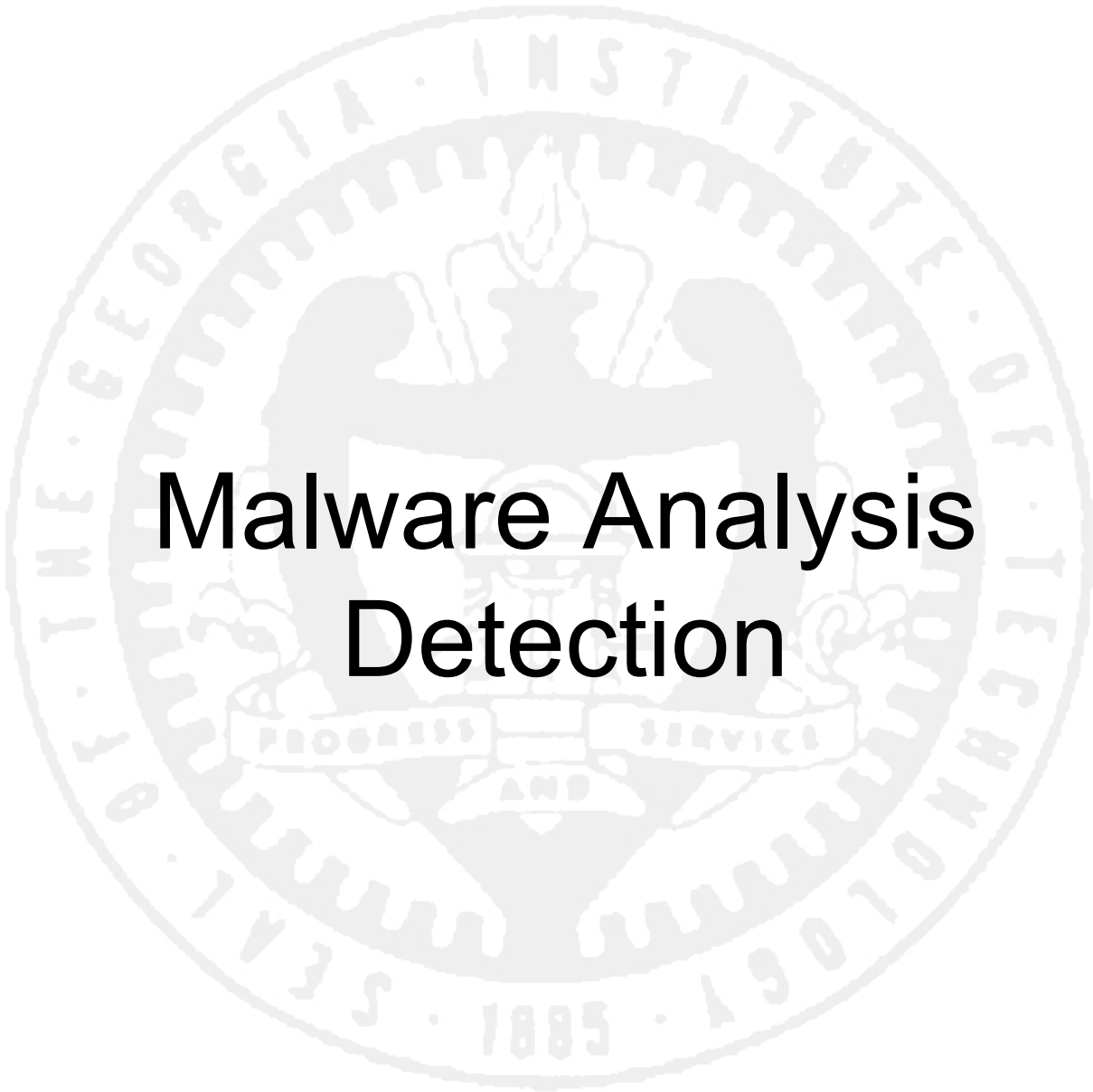
File **postcard.exe** received on **02.25.2009 22:03:16 (CET)**
Current status: **finished**
Result: **35/39 (89.75%)**

Collected on 2/25/2009

File **disc.exe** received on **02.25.2009 21:53:13 (CET)**
Current status: **finished**
Result: **11/39 (28.21%)**

Why Automation?

- **Vastly increased volume of samples**
- **GTISC averages 1M new samples/ month**
  - **Higher for commercial security organizations**
- **Volume makes manual analysis untenable**

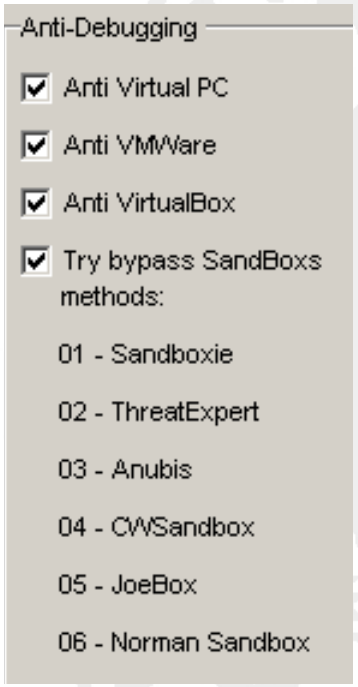


Malware Analysis Detection

Malware Analysis Detection

- **Environment-aware Malware**
  - **Conficker**
    - **Checks for relocated LDT**
  - **TDL4**
    - **Checks for device emulation via WQL**
  - **Bredolab**
    - **Checks for for device emulation via DeviceIoControl()**

Detection Cont'd



- **Analysis tool/environment detection is a standard, inexpensive option**

Transparency Requirements

- **Higher Privilege**
- **No Non-privileged Side Effects**
- **Same Instruction Execution Semantics**
- **Identical Exception Handling**
- **Identical Notion of Time**

Requirements Cont'd

- **In-Guest Tools**
  - **No higher privilege**
  - **Non-privileged side effects**
  - **Exception handling issues**
- **Reduced Privilege Guests (VMware, etc)**
  - **Non-privileged side effects**
- **Emulation (QEMU, Simics)**
  - **No identical instruction execution semantics**

Detecting QEMU

- **IRETD with 0x26 prefix**

```
#include <stdlib.h>
#include <stdio.h>
#include <windows.h>

int seh_handler(struct _EXCEPTION_RECORD
                    *exception_record,
        void *established_frame,
        struct _CONTEXT *context_record,
        void *dispatcher_context)
{
    printf("Malicious code here.\n");
    exit(0);
}
```

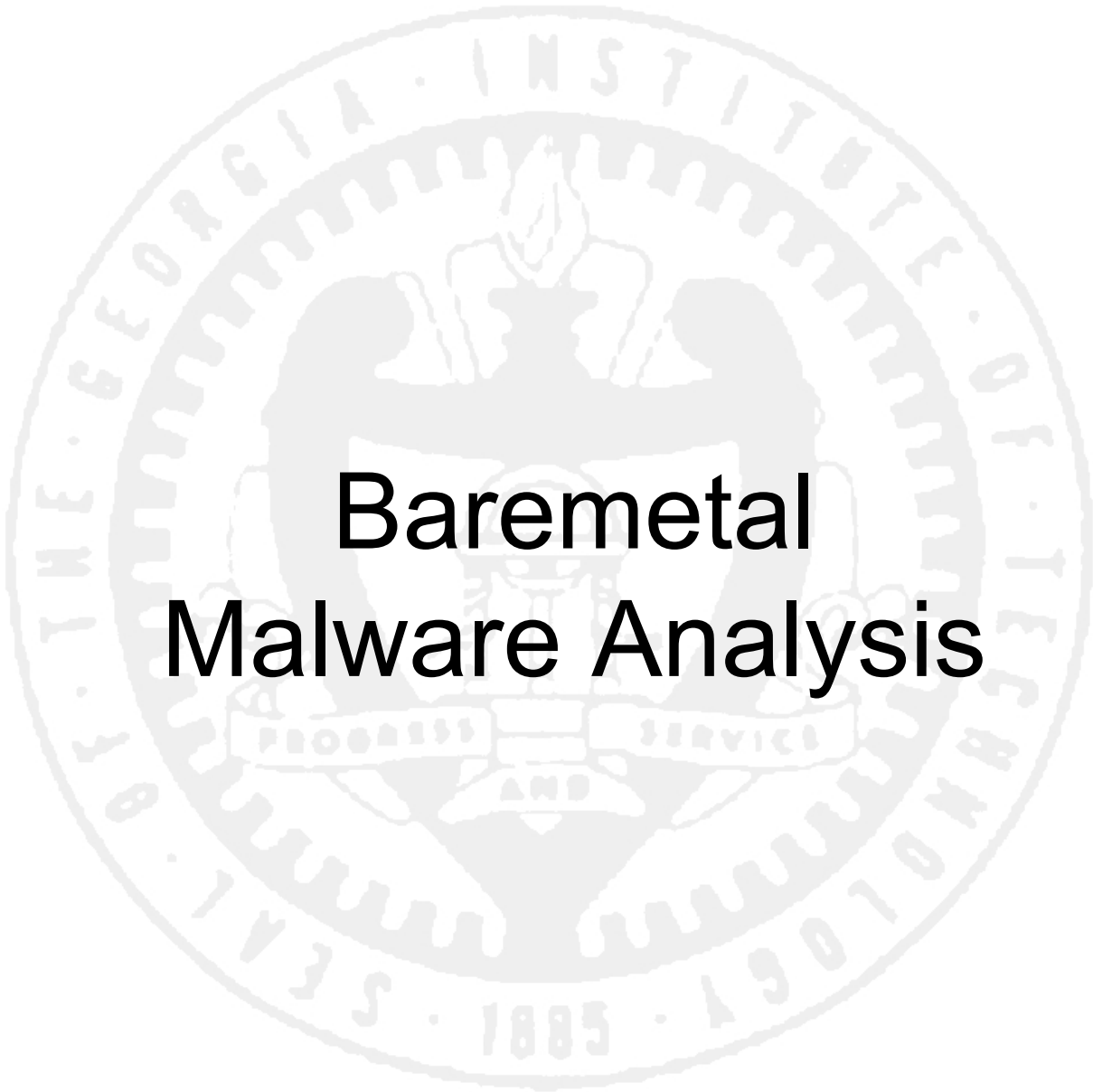
```
int main(int argc, char *argv[]) {

    unsigned int handler =
        (unsigned int) seh_handler;

    printf("Attempting QEMU detection.\n");

    __asm("movl %0, %%eax\n\t"
        "pushl %%eax\n\t"::
        "r" (handler): "%eax");

    __asm("pushl %fs:0\n\t"
        "movl %esp, %fs:0\n\t");

    __asm(".byte 0x26, 0xcf");

    __asm("movl %esp, %eax");
    __asm("movl %eax, %fs:0");
    __asm("addl $8, %esp");

    return EXIT_SUCCESS;
}
```

Detecting VMware, KVM

- **VMware**
  - **Older versions primarily use binary software translation**
    - **SYSRET treated as NOP when executed in ring 3**
- **KVM**
  - **Uses hardware virtualization extensions**
    - **Certain instructions cause VMExits**
    - **Older versions terminate with unhandled exit on guest execution of VMREAD**

Why Transparency?

- **Analysis environment detection commoditized, popular**
- **Detection vulnerability trend does not suggest decrease over time**
- **Certain types of detection vulnerabilities automatically discoverable**



Baremetal Malware Analysis

Baremetal Challenges

- **Conceptual**
  - **Physicalizing virtual machine**
- **Scalability**
  - **Cost of hardware**
  - **Efficiency of processing**
- **Automation**
  - **Managing system state**
  - **Ensuring longevity of hardware**

Baremetal Cluster Hardware

- **Baremetal Controller**
  - **Supermicro 5016I-MTF**
    - **X3430 Processor, 8GB RAM, 4 x 250GB disks**
- **Baremetal Non-Virtual Machine (NVM)**
  - **Supermicro 5015A-PHF**
    - **Integrated Atom processor, 1GB RAM**
- **Cluster Networking**
  - **Cisco WS-C2960-24TC-S**
    - **24 10/100Mb, 2 1Gb Ethernet ports**

Baremetal Cluster Technologies

- **Linux Device Mapper**
  - **Create Copy-on-Write (CoW) block device**
- **ATA-over-Ethernet (AoE)**
  - **Make CoW device available over network**
- **g Preboot eXecution Environment (gPXE)**
  - **Boot NVM into OS on network CoW device**
- **Intelligent Platform Management Interface (IPMI)**
  - **Manage NVM system state**

NVMTrace

- **Software controller for automated baremetal malware analysis**
  - **Executes each sample in its own sterile, isolated non-virtual machine**
- **Provides access to NVM disk contents and network traffic**
  - **Use with your favorite network traffic and disk forensic tools**

NVMTrace Corner Cases

- **System Clock**
  - **Sample can modify system time**
  - **Modify gPXE to set sane value, sync immediately prior to sample execution**
- **NVM PSU Lifetime**
  - **Turning NVM on, off hundreds of times each day quickly destroys PSU**
  - **Use resets instead**

Conclusion

- **Analysis environment detection commoditized, increasingly popular**
  - **Virtualization still a valuable analysis tool, but can be supplemented**
- **Advances in hardware make scalable baremetal malware analysis possible**
- **NVMTrace facilitates automated baremetal malware analysis**

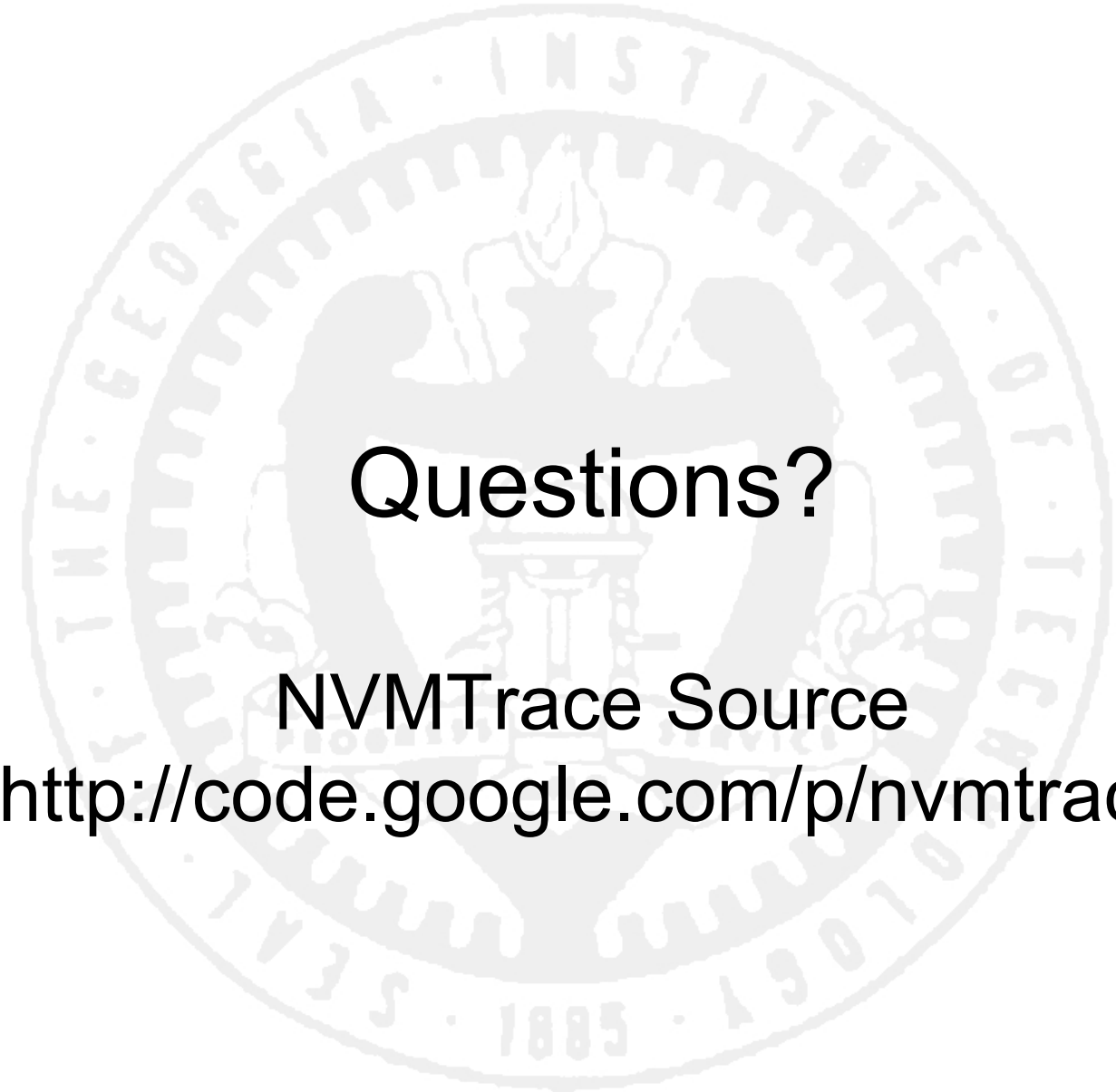
Future Work

- **AoE Disk Forensics**
  - **Examine controller-NVM AoE network traffic**
  - **Record disk-level events as they occur**
- **Arduino Boards**
  - **Connect to NVM via USB**
  - **Inject keyboard/mouse events**
  - **Activate trigger-based malware**

Acknowledgements

- **Robert Edmonds**
  - **System design**
- **Michael Lee**
  - **System implementation**
- **Artem Dinaburg**
  - **Environment detection**
- **David Dagon**
  - **System concept**

Please fill out your feedback forms.



Questions?

NVMTrace Source
http://code.google.com/p/nvmtrace