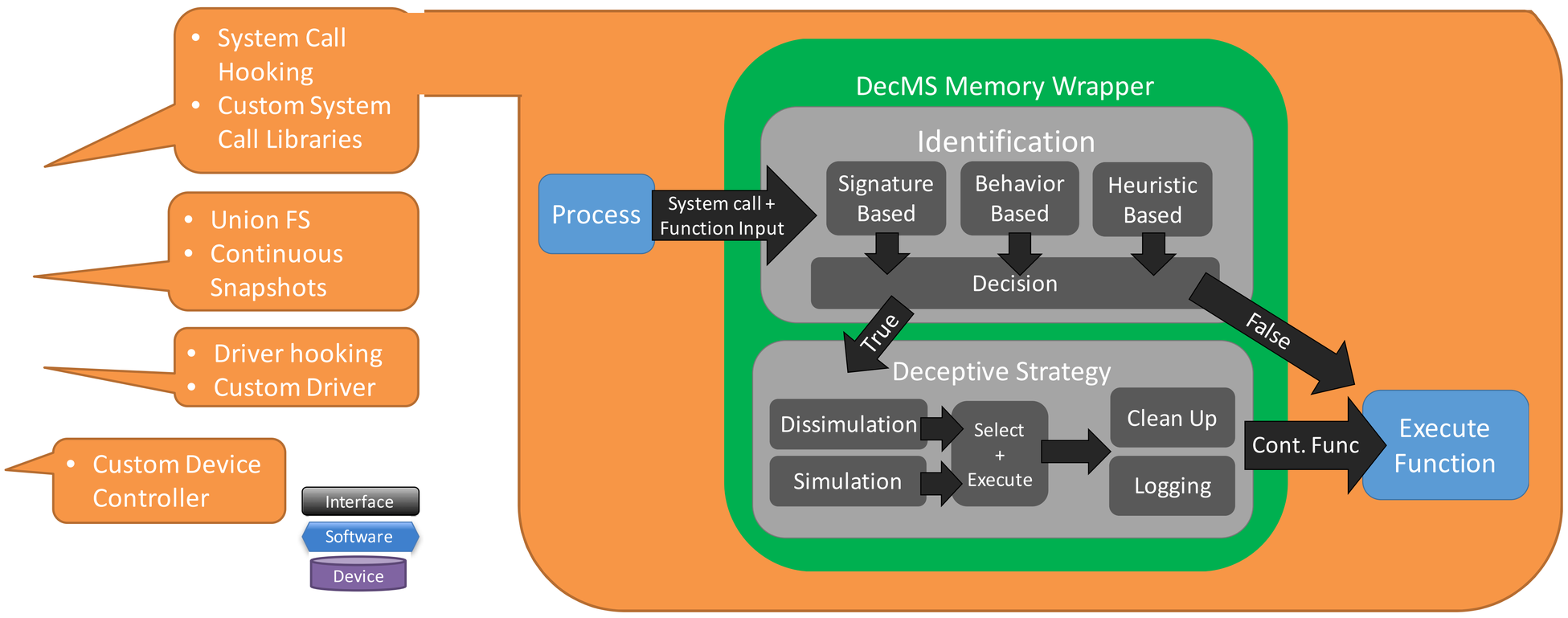
# Deceptive Memory Systems – Countering Anti-Forensics with Deception

Christopher N. Gutierrez, Eugene Spafford, Saurabh Bagchi

**Abstract:** The identification and preservation of digital evidence are crucial to uncovering the truth in digital crime. The computing systems that criminals compromise may store forensically valuable information. However, a sophisticated attacker can also compromise the integrity or availability of forensically valuable information. This work explores the use of deception to enhance the preservation of forensically valuable data objects through Deceptive Memory Systems (DecMS). When an attacker attempts to purge or modify evidence, DecMS migrates the evidence into a container and tricks the adversary into believing that their malicious action was successful. A forensic examiner may then run additional analysis on the potential evidence stored in the container.




## Dissimulation (Hiding the Real)

## Simulation (Showing the False)

| | Masking | Repackaging | Dazzling |
|---|---|---|---|
| Mimicking | **Dissimulation:** Isolate external logging and preservation.<br><br>**Simulation:** Mimic write latencies for secure data deletion. Produce data that looks like secure deletion. | **Dissimulation:** Write-back encrypted data items with secret key. Hide metadata within file via steganography.<br><br>**Simulation:** Encrypted data "looks like" what the secure delete produces. Produce hashes for old files prior to introducing steganography. | **Dissimulation:** Strategic failure - partial execution of secure deletion left in a recoverable state.<br><br>**Simulation:** Mimic write failure, produce messages, logging, and system behavior |
| Inventing | **Dissimulation:** Isolate external logging and Preservation.<br><br>**Simulation:** Fake CPU and/or I/O wait to allow time for isolation and preservation. | **Dissimulation:** Fragment, encrypt, and scatter deleted data items or metadata in unused memory space.<br><br>**Simulation:** Fake CPU and/or I/O to allow evidence to be placed elsewhere in the system. Secure delete produces expected results. | **Dissimulation:** Strategic failure - partial execution of secure deletion left in a recoverable state<br><br>**Simulation:** External cause of failure: power failure, hanging process, etc. |
| Decoying | **Dissimulation:** Isolate external logging and Preservation.<br><br>**Simulation:** multiple sources of data item: back-ups, RAID disks, NFS, overlays, etc. | **Dissimulation:** Fragment, encrypt, and scatter deleted data items or metadata in unused memory space.<br><br>**Simulation:** Fragment, encrypt, and scatter deleted items or metadata in multiple places. For each fragment, produce "decoy" fragments. | **Dissimulation:** Strategic failure - unknown causes, random failures, unstable system.<br><br>**Simulation:** Produce decoy log entries, notifications, kernel panics messages, etc. |

Portions of this research are funded by National Science Foundation Grants CPS-1329979, Science and Technology Center CCF-0939370, and EAGER-1548114.

PURDUE UNIVERSITY