Chapter 2

Trust and Protection in the Illinois Browser Operating System

2.1 Summary

Illinois Browser Operating System (IBOS) is a new OS and web browser presented by Tang, Mai and King, which improves on integrity and confidentiality of web data from past approaches by removing typically shared OS components and system services from the browser's trusted computing base (TCB). These components, include device drivers, network protocol implementations, the storage stack and window management software. In the IBOS implementation, all of these components run above a trusted reference monitor which enforces security policies. These components operate on browser level abstractions, allowing the reduction of the TCB by mapping browser security policies down to the lowest level hardware directly and removing drivers and system services from the TCB.

Each visited URL of the IBOS browser has a unique process in a unique namespace. The name of the process is the same as the host of the URL. Traditional application and web plugin processes are labeled as "localhost" and communicate via L4's IPC mechanism. Process labels are generated by listening to URL related system calls rather than allowing the untrusted application to assign its own label. When web data is requested, a system call is made which traps into the IBOS kernel. IBOS looks for and existing process with a label matching the host of the requested URL. Where an existing process is not present, a new process is created. By keeping components in separate processes, process protection domains are utilized to protect against unauthorized access. Further to this, each component is restricted to only performing it's designated task. For example, the user interface can never ask for cookie data and the storage manager cannot impersonate a network process to send a synthesized HTTP data attack to a web page instance.

Although IBOS currently does not prevent deletion or replay attacks, it provides a secure web browser service with proven security enhancement in an area where security is an ever increasing concern.

2.2 Evaluation

2.2.1 Pros

- Mention of availability as an important aspect to browser security (details left for additional publications)
- Video frame buffers used per process/web page to prevent a malicious program from reading the screen data.
- Push to formal verification is mentioned in the paper.
- Vulnerabilities are categorized and explained before presenting results.
- Shortcomings are clearly mentioned. Self or external improvements can be easily recognized and implemented.
- Excellent system under test description with hardware, software and version numbers clearly defined.
- Anomalies in results are identified and explored.

2.2.2 Cons

- Inadequate introduction. In particular, the summary of sections and their content.
- Limitation of replay attack protection mentioned but not explained.
- The unit of "lines of code" seems inappropriate for TCB size comparison for an under developed browser and popular browsers. The higher functionality of popular browser may warrant the additional TCB size.
- "Misc vulnerabilities" provides a substantial section of vulnerabilities. It represents the second largest category without any description of the types of vulnerabilities within this section.

2.2.3 Criticism

L4 spin-lock contention has been blamed for bad performance when testing wikipedia. I find great strength in the paper for investigating rather than speculating the causes of these anomalies, however, spin-lock contention is most likely caused by the chosen CPU allocation. Security functionality of the IBOS is run on CPU0 while the userland interface is run on CPU3. Each context switch will require spinning on a lock until the other CPU is busy. I speculate that because the other evaluated browsers do not rely heavily on OS services, these context switches are reduced.

2.2.4 Minor issues

A few spelling and grammatical errors have been identified as follows: On page 2, left column, last line:

> "Our goal is to build a system where a user can visit a trusted web site safely, even one or more of the components on the system have been compromised"

Should read as:

> "Our goal is to build a system where a user can visit a trusted web site safely, even if one or more of the components on the system have been compromised"

On page 5, right column, line 7:

> "a HTTP request"

Should read as:

> "an HTTP request"

On page 5, right column, line 11:

> "sending it to network manager"

Should read as:

> "sending it to the network manager".

2.3 Questions

Questions I have about the IBOS system relate to the large scale integration of the system. Do the security properties of the browser produce any limitations on functionality? Is some functionality of the browser not yet implemented such that the browser artificially runs faster than the browsers used for comparison? Will the size of the trusted code base explode once functionality similar to the evaluated browsers is implemented? Questions which I feel should definitely have been answered in the paper relate to the complexity of process management as each host is assigned a unique process. Is There a limit to the number of hosts which can be concurrently accessed? To benefit from memory protection domains, connections must be assigned their own address space and scheduled by the kernel. How is starvation of other services avoided when the user opens many URLs from unique hosts?