C27 Proving Database Tampering Through In-Memory Object Reconstruction

Aisha Ali-Gombe, PhD, Towson University, Towson, MD 21252; Sneha Sudhakaran, MTech, Louisiana State University, Baton Rouge, LA 70808; Andrew Case, MS, Louisiana State University, Baton Rouge, LA70808; Golden G. Richard III, PhD, Louisiana State University, Baton Rouge, LA 70808*

Learning Overview: After attending this presentation, attendees will gain an understanding of in-memory object recovery from userland process address spaces on Android™ devices and how the retrieved objects and their metadata can be leveraged to reconstruct database queries in system-wide content providers and prove database access and manipulation.

Impact on the Forensic Science Community: This presentation will impact the forensic science community by illustrating how memory forensics can be used to generate causal relationships between applications and native system database accesses. Reconstructions with real applications of allocated objects from process runtime memory will be demonstrated and their metadata will be used to trace database activities. This approach will illustrate how vital this detection process is in proving attribution on a multi-app platform, such as Android™.

The Android™ operating system stores system-wide sensitive user data, such as text messages, call logs, and calendars, in SQLite databases, which are accessible through a management service called a native content provider. Each provider is associated with a single SQLite database file. To access any SQLite data object, an application makes a Create, Read, Update, and Delete (CRUD) request to the provider using a content resolver object. Each CRUD function corresponds to one of the SQL data manipulation queries (insert, update, delete, and select).[1] The most important object in a CRUD function is the Universal Resource Identifier (URI), which tells the resolver which provider to contact and collectively with the remaining parameters informs the provider about which query should be performed.

In this research work, the objective was to recover important objects allocated by user processes at runtime, then use their metadata to reconstruct events. On the new Android™ Runtime (ART), dynamic objects are allocated using the RosAlloc allocator. The RosAlloc provides an efficient method of allocating sequential memory space to objects of the same size.[2] Every allocated object is created based on its class specification and contains a pointer within its memory space to its definition. Also, it has an associating lock that holds the ArtMethod for the object. To recover database activities, individual threads in the process runtime are identified, then all the allocated runs per thread and their associated non-free slots are recovered. Based on the object class definition, URI class objects are enumerated and parsed. The metadata of those objects will be traced to identify the "locking" CRUD function. Finally, the resulting data will then be correlated with a low-level parcel recovered from registered system-wide Inter-Process Communication (IPC) threads.

Reference(s):

1. Ali-Gombe A., Richard G.G. III, Ahmed I., Roussev V. Don't Touch that Column: Portable, Fine-Grained Access Control for Android's™ Native Content Providers. *Proceedings* of the 9th ACM Conference on Security & Privacy in Wireless and Mobile Networks; July 2016, Darmstadt, Germany, (pp. 79-90). ACM.
2. Bhatia R., Saltaformaggio B., Yang S.J., Ali-Gombe A., Zhang X., Xu D., Richard G.G. III. Tipped Off by Your Memory Allocator: Device-Wide User Activity Sequencing from Android™ Memory Images. *Proceedings* of the 2018 Network and Distributed System Security Symposium (NDSS 2018); February 2018, San Diego, CA.

Android™, Memory, SQLite