Preface

This issue contains papers on recent advances in cryptography that are embodied in the Common Cryptographic Architecture (CCA) and associated technology for managing single-key cryptographic systems. We are indebted to R. C. Elander of the IBM Kingston Programming Center in Kingston, New York, and C. L. Symes of IBM Enterprise Systems in Poughkeepsie, New York, for their contributions to the planning and preparation of this issue.

Single-key cryptographic systems are those in which the sender and receiver share a private key that allows them to encode and decode a message transmitted between them. In contrast, public-key systems have different keys for the sender and receiver, and neither can derive the other's key. The single-key approach to message-level secure communication was selected by the National Security Agency and the National Bureau of Standards, as it was then known, in the mid-1970s. In 1977, they adopted as a standard an IBM-designed single-key security algorithm, which became known as the Data Encryption Algorithm (DEA) and Data Encryption Standard (DES). DEA and DES remain today as the standard single-key technology, having resisted concerted efforts to find general methods for cracking the resulting codes.

In the intervening years there have been many advances in single-key secure communications and in the implementation of such security systems. All of this activity has created a requirement for an overall architecture for secure communications programming. IBM has answered that requirement with the Common Cryptographic Architecture (CCA) and has built products based on that architecture. This issue has six papers on single-key cryptography and its architectural and technical support.

Johnson, Dolan, Kelly, Le, and Matyas discuss the CAA Cryptographic Application Programming Interface (Cryptographic API), which has been announced by IBM. It simplifies use of cryptographic services specified by the CCA. The authors describe advanced user features and provide typical application scenarios to show how easily these new callable key-management services can be used. Symes has provided a sidebar that gives background information for those not familiar with general product features and support for cryptography and computer security.

Much of the current technology for cryptography is based on the management of security keys that are associated with resources and people. Matyas describes how CCA implements keys through control vectors that define permitted uses of that key in the context of a DEA-based system. The key is cryptographically linked to the control vector, and the resulting token is the basic unit for secure processing and distribution. He also contrasts control vectors with another means of key management, namely variants.

Matyas, Le, and Abraham contribute a paper on management of keys using a scheme that is part of the new IBM Transaction Security System. That system is supported by special instructions that form the secure hardware basis of the Transaction Security System cryptographic facility (CF). The advantage of the control vector approach is shown to be programs that are simpler and that operate on a robust data structure.

Within the context of the CCA, there is a requirement for a new approach to cryptography—one that is implemented through special instructions in general-purpose computers. The result is the ESA/390™ Integrated Cryptographic Facility (ICRF), which directly supports key management and control vectors in the hardware. Yeh and Smith describe the ICRF, its objectives, applications, key management, and physical security.

Abraham, Dolan, Double, and Stevens discuss in detail the IBM Transaction Security System as an

implementation of the CCA. Since there is no complete and final algorithmic solution for computer and data security, they show how engineering and common sense can be used to form the basis for the development of practical cryptographic systems, and especially for the Transaction Security System in CCA.

The Transaction Security System, as discussed above, fully implements the CCA. But, as Johnson and Dolan show in the final paper of this issue, it also provides extensions to CCA to satisfy additional customer requirements. Extensions described in this paper include programmer support through intuitive security mechanisms, parametric enhancements to the Cryptographic API, and masking of complexity through a layered approach, among others.

The next issue of the *Journal* will contain several papers related to computer communications, along with papers on such subjects as software quality, REXX partial compilation, and multimedia for education.

Gene F. Hoffnagle
Editor