Security on z/OS: Comprehensive, current, and flexible

by R. Guski
J. C. Dayka
L. N. Distel
W. B. Farrell
K. A. Gdaniec
M. J. Kelly
M. A. Nelson
L. H. Overby
L. G. Robinson

In this paper, we summarize and explain the security functions available to a typical enterprise computing installation using the IBM z/OS™ operating system and Security Server. The discussion is at a high level, aimed at enterprise decision makers and application architects. The intent is to explain the comprehensive security componentry within z/OS and to show how these techniques and functions are exploited by modern distributed and Internet applications. Both z/OS and the IBM e server zSeries™ product family have a rich heritage and significant presence in the evolving computing marketplace. Consequently, this discussion includes some computer security history and projections of the relevant future.

It is generally believed that about 70 percent of all large enterprise business data reside on mainframe computing system server platforms running the IBM z/OS* operating system or its predecessor, OS/390* (Operating System/390). From a security perspective, mainframe computer operating systems are the *de facto* intellectual standard to which newer operating systems are compared.

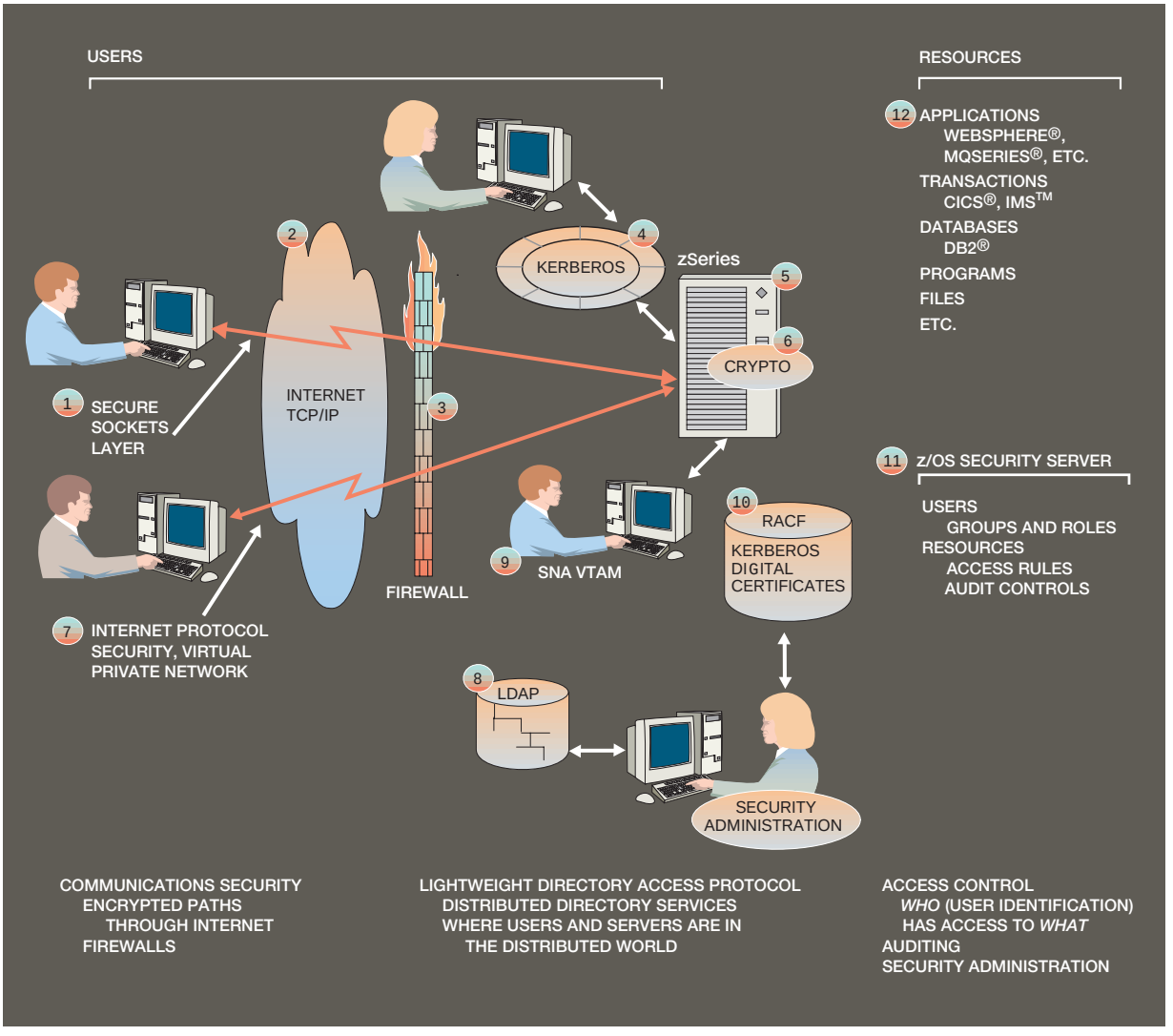
As architects, designers, and marketers of IBM mainframe computing system server products, the authors of this paper are often asked to explain why z/OS and the IBM e server zSeries* family of computing system platforms enjoy this reputation for strong security. The question comes not just from customers and potential customers, but also from architects and designers of other server platforms who are considering the application of similar security componentry to their products.

Actually, the answer already exists in print, much of it available on the Web, in volumes of highly specific technical documentation. Such material is necessary and effective in documenting the details of day-to-day operations but is not suited to quickly gaining the high-level understanding that is often needed by a decision maker or application architect. We have worked hard to make this paper relatively easy to read. Because it is fairly long, we have organized the material into sections; we hope that the sections will be revisited as reference material. Also, to aid our readers in navigation, we have included a high-level overview chart that illustrates the elements of z/OS security as they relate to one another, to end users, and to applications executing within z/OS (Figure 1).

Figure 1 shows users interacting with an enterprise server via various communication protocols, including the Transmission Control Protocol/Internet Protocol (TCP/IP) (2) and Systems Network Architecture/Virtual Telecommunications Access Method (SNA/VTAM) (9). The z/OS Security Server (11) and its components—the Resource Access Control Facility (RACF*) (10) and the Lightweight Directory Access Protocol (LDAP) server (8)—are providing security and directory services for the users who are accessing sensitive enterprise applications, information, and services (12). In this paper, we explain how the elements of

0018-8670/01/$5.00 © 2001 IBM

Figure 1 z/OS security overview



z/OS security fit together cooperatively and comprehensively and why they are effective.

We promise to use a minimum amount of technical jargon to present our message, although we assume that our readers have a reasonable understanding of modern large enterprise distributed and centralized computing. When we use a technical or marketing term, we will explain it, give its generally recognized acronym, and then use either the complete term or the acronym through the remainder of the paper.

Background

The z/OS operating system, from its beginnings as Multiple Virtual Storage (MVS*) in the 1970s, was built on System/360* (S/360), and now zSeries, a strong hardware architecture that provides for the safe execution of multiple applications on one system. Working storage for each application is separated from the storage of other applications, as well as from the storage used by the supervisor functions of the operating system. In addition, applications run in a different hardware "state" than supervisor func-

tions and therefore do not have direct access to privileged hardware instructions. On top of this is layered a common point of user authentication and access control, both as a way to reduce the need for each application to provide these services uniquely, and as a way to reduce administrative burden.

Twenty years ago, of course, all this security structure needed only to support an environment of isolated batch processing systems or systems with terminal access through private corporate networks. Over time, and especially in the last decade, our customer's businesses have changed dramatically, reaching out to their suppliers, distributors, and customers through a variety of networking techniques. Businesses are now truly "worldwide" operations and rife with merger activity. In short, the nature of business has changed and so have the security requirements.

For example, until recently banking was done either through a human bank teller who had specific security capabilities, or with an automatic teller machine (ATM), where personal identification numbers (PINs) secured those transactions. Today, almost every bank has moved into home and Internet banking, and many banks are becoming Internet trust brokers, extending the trust structure that they maintain with credit cards into a trust structure built around X.509 version 3 digital certificates and encryption technology.

As MVS became OS/390 in the mid-1990s, we saw the beginning of changes in operating system security function to support the business changes that were starting to happen. For example, there have been some profound changes in the way business applications are built. One change is that a single application, which can now start in the Internet, is expected to run across multiple platforms. As this has happened, OS/390 has further transformed into z/OS, and we have had to marry the security infrastructure with many open security standards. A second change is a rise in the importance of encryption. IBM enterprise servers have had a long history with encryption, having introduced optional hardware-based encryption in 1991. (See label 6 in Figure 1.) Since 1997, hardware encryption has been standard in S/390 processors and (now) zSeries processors. This is because we understand that encryption is the basis of security when using the Internet, and that our customers require the performance and security that our tamper-resistant hardware encryption provides. Later, when we discuss the Secure Sockets Layer (SSL), which has become the dominant technique for enterprises to communicate securely with their customers via Internet browsers, we include numbers to show the dramatic increases achieved in the performance of the z/OS cryptographic functions that SSL requires.

As distributed applications have emerged based on open computing standards, z/OS has merged its proven security architecture with the best of the new technologies. For example, the LDAP server offers a commercially popular implementation of a subset of the X.500 Directory Services Open System Interconnection standard as defined by the International Standards Organization/International Electrotechnical Commission. Although users can still identify and authenticate themselves to computing facilities with LDAP using traditional user identifiers (IDs) and passwords, modern third-party authentication techniques such as Kerberos[1] and X.509 version 3 digital certificates are now supported options. Each of these technologies and their implementation on z/OS will be explained.

Before we discuss the elements of security on z/OS, we need to explain the packaging of z/OS. It is rather simple. In the mid-1990s we took the existing MVS operating system and nearly 200 separately installed functions, and combined them into one tested, prepackaged operating system, OS/390. We included data access methods, communication and networking functions, workload management functions and, of course, security functions. Some security functions are provided in the base OS/390 and z/OS products, while others are packaged in an optional Security Server feature. Chief among the functions packaged within the Security Server is the Resource Access Control Facility (RACF).

RACF incorporates various elements of security, such as user *identification and authentication* and *access control*, which will be discussed in detail later. Some customers develop their own security manager software (most unusual) while some others use security manager software offered by other vendors. Computer Associates CA-ACF2** and CA-Top Secret** are well-known alternatives to RACF. While some equivalence between RACF and the alternatives offered by Computer Associates can be expected, this paper is written from a RACF perspective. In 2001, the System/390* family of computers was replaced by the zSeries hardware family, which is based on a new hardware architecture. Likewise, the OS/390 operating system was replaced by an enhanced oper-

ating system, z/OS, which continues this packaging philosophy. Unless specifically stated otherwise, security support discussed here exists in both OS/390 and z/OS. In many respects, this paper tells the story of how we have transformed the security functions of IBM's most powerful computer operating system, now z/OS, so that e-business applications can take advantage of the large amounts of business data that are secured by RACF.

These functions are discussed in the following order: (1) cryptography, (2) user identification and authentication, (3) basic authentication (user ID and password), including authenticated credentials, password management, and trusted third-party user identification and authentication (digital certificates, the Secure Sockets Layer [SSL], and Kerberos), (4) System Authorization Facility (SAF), (5) access control, (6) security service exploitation, (7) auditing and logging, (8) directory services, (9) networking and communications security, and (10) ethical hacking and certifications.

Cryptography

We start our discussion with the topic of cryptography, which we consider to be critical for security on the Internet, for banking and finance, for e-business, and for the emerging "intelligent" infrastructure.

Cryptography is the algorithmic "scrambling" or *enciphering* of information based on a *key* such that the information may be unscrambled or *deciphered* later with a complementary algorithm and decryption key. Typical uses of cryptography include ensuring the confidentiality and integrity of data composing large financial transfers between institutions, protecting the confidentiality of personal identification numbers that flow between automated teller machines and issuing banks, and generating electronic signatures that more and more often have the same legal validity as personal written signatures. The Secure Sockets Layer, which is the core technology used to secure Internet transactions, uses cryptography both for authentication of clients and servers and for data confidentiality. SSL is discussed in detail in a later section.

Since 1991, our integrated hardware encryption has consistently been the industry leader both in level of security provided and in performance. Hardware encryption devices provide a tamperproof security boundary that can be an absolute requirement for financial applications. When compared to application software implementations of computationally intensive public key cryptography, hardware devices can provide huge performance advantages.

The zSeries hardware now has two distinctly different cryptographic hardware engines that are supported under z/OS; the CMOS (complementary metal oxide semiconductor) Cryptographic Coprocessor and the newer PCI (peripheral component interface) Cryptographic Coprocessor (PCICC). The PCICC provides the capability for rapid response to customer requirements that was sometimes difficult to achieve with the CMOS Cryptographic Coprocessor alone.

The CMOS Cryptographic Coprocessor is the CMOS technology follow-on to the original bipolar technology ICRF (Integrated Cryptographic Feature) that first shipped on S/390 in 1991. It is a pure hardware implementation; no microcode executes within the secure cryptographic boundary. It has achieved a FIPS (Federal Information Processing Standard) 140-1 Level 4 security rating from the U.S. government's National Institute of Standards and Technology (NIST). The CMOS Cryptographic Coprocessor provides very fast Data Encryption Standard (DES) encryption, message authentication code checking (MACing), key management, and PIN functions. These functions have extensive customer use throughout the world, particularly in the financial community. The CMOS Cryptographic Coprocessor also has a secure, fast RSA (Rivest, Shamir, Adleman) signature and key distribution capability. The RSA key distribution functions are becoming critical to the enablement of SSL implementations on z/OS.

While the CMOS Cryptographic Coprocessor offers spectacular performance for DES functions and spectacular reliability, it is somewhat inflexible in that it is difficult to add new function that executes within the secure hardware boundary. The RSA performance—while good—has not kept pace with the exponentially growing demands of SSL.

IBM's answer to both problems was to incorporate the PCI Cryptographic Coprocessor as an optional feature on zSeries and S/390 systems. The PCICC feature is built around IBM 4758-2 PCI Cryptographic Coprocessor cards with special enhancements and adaptation packaging. The IBM 4758-2 also has a FIPS 140-1 Level 4 security rating. The card has an operating system and an essentially standard C programming environment for the implementation of new secure functions. It is even possible for custom-

ers to create their own unique functions within the secure boundary of the PCICC card and to have these functions accessed via extensions to the Integrated Cryptographic Services Facility (ICSF), which is the z/OS host access method to all zSeries and S/390 cryptographic hardware. The facility to provide customer-unique function is known as user-defined extensions (UDX). Finally, the PCICC feature has excellent RSA performance, especially for the RSA private key operation that is used in the SSL handshake. A single card can support about 135 RSA handshakes per second, while a single CMOS Cryptographic Coprocessor can support about 75. ICSF routing algorithms for RSA functions scale well as PCICC features are added, so that in an environment where 16 PCICC coprocessors and two CMOS cryptographic coprocessors are active, support for over 2000 SSL handshakes per second is possible.

The CMOS Cryptographic Coprocessor and the PCICC feature implement IBM's Common Cryptographic Architecture (CCA). CCA is an architecture consisting of a set of well-designed cryptographic functions and a method for secure separation of cryptographic keys to ensure that keys can only be used in specific functions. Such a design is fundamental to any good implementation of cryptography and is necessary to comply with cryptographic standards such as ANSI (American National Standards Institute) X9.24. CCA is also an application programming interface (API) that provides access to these functions via a set of terse procedural high-level-language function calls. The CCA API is a true cross-platform API, supported on z/OS and OS/390 via the zSeries, S/390 CMOS Cryptographic Coprocessor, and the PCICC features; OS/400* via the iSeries and AS/400* PCICC feature; AIX* via the pSeries and RS/6000* PCICC feature; Microsoft Windows NT**, Microsoft Windows** 2000, and OS/2 via the IBM 4758 PCI Cryptographic Coprocessor.

On z/OS, two additional APIs have been successfully layered on top of the CCA API. The first is the BSAFE[2] hardware API (BHAPI), which allows applications that use the BSAFE toolkit, developed by RSA Security Inc., to make use of the zSeries and S/390 hardware by making very minor modifications to their cryptographic application program. Hardware cryptographic functions can also be accessed through the Open Cryptographic Services Facility in z/OS. Open Cryptographic Services Facility is an implementation of the Common Data Security Architecture (CDSA) for applications running in the UNIX** services environment. Access to cryptographic functions on z/OS is subject to access controls, which will be discussed later.

As stated previously, cryptography is a core technology that supports several elements of security on z/OS. One of these elements, which we discuss now, is user identification and authentication.

User identification and authentication

We define *identification and authentication* as the act of *identifying yourself* to the computing system and then *proving that you are who you claim to be*. Users identify themselves to a computing system application with a user ID and then prove that they are who they claim to be by correctly entering the password that has been associated (previously) with that user ID. Refer to Figure 1, labels 1, 4, 8, and 10, to see where identification and authentication are implemented within the z/OS security elements. Note that labels 8 and 10 identify user registries—RACF and LDAP—that incorporate identification and authentication functions in the security services they provide. The identification and authentication functions make use of the information within the registries. Identification and authentication technology, in one form or another, is implemented within several components of z/OS, using multiple security technologies. One of these forms is "basic authentication," which depends on passwords for authenticating users, and this is our starting point.

Authorization via password. Passwords were first used in computing to validate (or authenticate) a user's authority to access a particular data file. That is, the file was "protected with a password" and user(s) who knew the password could access the file. The file service routines installed on the computer actually provided the security, because they would allow access to the file only when the correct file password was supplied. This technique, which still has some use today, has fundamental drawbacks: (1) multiple files require multiple passwords and (2) if multiple users require access to a common file, they all need to know the password, thus barring individual accountability for use of the file.

Basic authentication

Instead of authorizing access to individual files, a different approach is used by most modern multiuser computer operating systems, such as the z/OS operating system. With the RACF function of the Security Server for z/OS, the password authenticates the us-

er's authority to use an "identity." This technique is known as *basic authentication*.

Identities used with basic authentication, and with other identification and authentication alternatives to be discussed later, are preassigned within the user registry, for example the RACF registry, by a trusted security administrator. The administrator may assign an identity to an individual user, or to a set of users—in which case the identity is really a form of functional "role"—or to an application server daemon. At the same time, for administrative convenience, the security administrator may associate the user identity with one or more logical groups of users, all of whom then share authority to resources that (as we will show later) can be associated with the group. As with groups, identities associated with roles can be used to conveniently authorize multiple users, who have similar jobs, to the resources that they need to do their jobs.

Authentication occurs when the user accesses the computing system and claims to be a particular user by specifying a user ID. The Security Server then uses the user ID as a search key to look up the predefined identity record associated with the claimed user ID within the user registry. If a record for the claimed user ID is found, the password that has also been stored in the record is compared with the password supplied by the entering user, and if the two match, the user is considered to be authenticated.

The authenticated credential. Once authenticated, the individual process that the user has initiated—whether an interactive session, a batch job, a transaction, a Java** bean, or another function—is associated with the user's preassigned identity by means of a data construct called a "credential." In z/OS we call one form of the credential an *accessor environment element* (ACEE). The ACEE follows the process logically within the operating system, and it is available to identify the authenticated user during access control authorization checking and for auditing purposes.

The technique of combining basic authentication with an authenticated credential is used in z/OS in multiple ways; RACF and LDAP are important examples. We will resume our discussion of authentication when we introduce digital certificates and Kerberos. For now, we diverge to discuss the management of user passwords.

Password management. Password user authentication has been in use since the late 1960s, with the first interactive applications running on centralized computing facilities. Several improvements have been made over the years. For security of the password when stored within the user registry, most modern security managers, such as RACF, encrypt the password with a one-way encryption algorithm so that the actual clear-text password can never be recovered. This one-way encryption occurs first when the user chooses his or her password, and the resulting one-way encrypted value is stored in the security registry. The one-way encryption occurs again whenever the user is entering the system and specifies his or her password for authentication; the entered clear-text password is encrypted (refer to label 3 in Figure 1) as before, and the result is compared with the value stored in the registry.

Although the password is more secure than if it were stored in clear text in the registry, it is still subject to a "brute-force" hacker attack if the hacker is allowed to try many possible passwords until the correct one is found. On centralized systems, where the physical security of the registry is reliable, this form of attack is defeated by allowing only a small number of invalid attempts to be made against an individual user ID, within a reasonable time period, then revoking[3] the user ID. If the hacker can obtain an off-line copy of the registry, he or she can attack a user ID this way for a long time with many possible passwords until the correct one is found, so controlling access to the registry is vital. Encrypting the registry itself can help, but this just moves the hacker's problem to that of obtaining the encrypting key of the registry. If the hacker can get to the security registry in the first place, either physically or remotely, he or she can probably get to the encrypting key of the registry as well, and encrypting the registry complicates installation administration and disaster recovery operations, so this is not common practice.

One shortcoming of the user ID and password (basic authentication) model is that the password has to be passed from the end point, where the user is, to the physical location where the application—and security registry—reside, which may be far away. Security of the password in flight is very important and tricky to ensure. On z/OS, possible technologies for encrypting the password while in flight include the SSL protocol, and the use of a *virtual private network* (VPN) TCP/IP-encrypted "tunnel." Another technology used on z/OS is not to use the password at all for authentication processing. This approach is sup-

ported with RACF, which includes the ability to authenticate users with password substitutes called *PassTickets*, which flow within traditional password-based protocols. Since PassTickets can be generated dynamically by a trusted server, based on a secret encryption key that is shared with the target RACF-secured server, they can also be used to forward an authentication from one trusted server to another. This function is exploited by several products, such as Tivoli Global Sign-On and IBM WebSphere* Host On-Demand, to provide a form of single sign-on support that includes RACF secured servers at the back end.

Another problem exists with traditional passwords when the user is defined to multiple registries, possibly with multiple different user IDs. Even when the user IDs are the same, the problem is user password synchronization. That is, when the user changes the password on one of the systems, the change should be propagated to the other systems, so that the user does not have to make the change separately on each system or live with multiple passwords, perhaps writing them down and attaching them to one's workstation. RACF deals with these problems nicely today, in homogeneous RACF-secured systems within an enterprise, with the RACF Remote Sharing Facility (RRSF). RRSF can be configured, if desired, by enterprise security administration, so that end-user RACF passwords, as well as other security information, will be copied to other RACF systems automatically and securely according to preset installation policy. RRSF user-password propagation can even be established for a user who happens to have multiple different user IDs, with the same password—on the same system—and wants to be able to change his or her password(s) only once.

Installations often have policy governing the content and structure of user passwords and how often users must be encouraged to change them. Rules that say, for example, that "passwords must contain at least six characters, and that at least two of them must be numeric" are common. There are many variations of such rules and z/OS security manager products are rich in these functions. Finally, in regard to password management, a history of the passwords that have been used by each user should be maintained to defeat the ill-conceived practice of changing the password to what it is now or what it was for a recent period.

Trusted third-party user identification and authentication. Although basic authentication, with user ID and password, has served well and is still in widespread use, the technique has limitations when used with modern Internet applications. For example, basic authentication password registries are practical and common for enterprises where the total user population is counted in the thousands. But user-ID registries that total several millions can consume huge amounts of direct-access storage and create performance bottlenecks during run-time use. For these and other reasons, basic authentication is giving way to "trusted third-party" identification and authentication techniques.

Basic authentication is still in widespread use, but it has limitations when used with Internet applications.

The use of X.509 version 3 digital certificates with an associated public key infrastructure (PKI), and Kerberos are two examples of modern trusted third-party identification and authentication techniques that are in common use. But, what is meant by "trusted third party"? The following analogy illustrates the principle.

During airport check-in, a traveler identifies himself to the airline gate agent by stating "I am Mr. Soandso." He then proves that he is indeed Mr. Soandso by displaying his driver's license, which includes a picture of his face. The image on the license authenticates that this license, which is a form of "credential," was issued to the person presenting it and that this person is indeed Mr. Soandso. The gate agent, who does not know or trust Mr. Soandso directly, trusts that the state Department of Motor Vehicles (DMV)—the well-known and trusted authority that issued the driver's license—has indeed validated Mr. Soandso's identity before issuing him the license. In this analogy, the state DMV is the *trusted third party*.

Digital certificates. A digital certificate can be used to identify and authenticate one user to another user (or server) and as the basis for generation of cryptographic keys for secure communication between these parties. Digital certificates that are created ac-

cording to the popular X.509 version 3 Internet standard are based on what is known as public key cryptography and are the subject of this section.

Public key cryptography uses two keys, called a "key pair," which are mathematically related to each other such that data enciphered with either key of the pair can only be deciphered with the other key of that pair. This basic technology can be used, not only to provide the foundation for private communication between parties, but also for the parties to be identified and authenticated to each other.

Although X.509 version 3 digital certificates may be issued by a server to its users for the users to identify and authenticate the server when accessing it, the certificates can also be issued by a third party that is mutually trusted by the communicating parties. In this case, the communicating parties do not necessarily have to trust, or even know, each other. Such a trusted third party is known as a *certificate authority*, and certificate authorities convey their trust by using public key technology to digitally "sign" the individual user certificates. X.509 version 3 digital certificates that are signed by the trusted third party, the certificate authority, are used by communicating parties to trust each other, in the degree that they each trust the third party.

Because the certificate authority is computationally involved with only the process of signing the certificates and not with the actual identification and authentication process, which happens only between the communicating parties (user and server, for example), it is normal practice for a single certificate authority to support *very large numbers* of users and servers. For this reason, X.509 version 3 digital certificate identification and authentication technology, and related public key infrastructure, is technology that is fundamental to Internet business.

The Security Server for z/OS supports X.509 version 3 digital certificates by allowing applications (the IBM HTTP [HyperText Transfer Protocol] server for z/OS, for example) to pass authenticated X.509 version 3 digital certificates into RACF, then have RACF translate them, using a mapping process, into appropriate RACF identities. The result is similar to what happens when a user signs on to the z/OS Customer Information Control System (CICS*), for example, using a user ID and password (as discussed earlier). That is, the user process is assigned an ACEE credential, which embodies the "identity" that has previously been assigned to the user by security administration. The z/OS Security Server also supports an end-user Web-initiated administrative process for the creation and automated distribution of user (and server) X.509 version 3 digital certificates in large numbers, providing security appropriate to the needs of e-business.

Secure Sockets Layer. Refer to label 1 in Figure 1. Secure Sockets Layer was developed by Netscape and has emerged as key technology in support of secure e-commerce. SSL is a public key cryptography-based extension to the TCP/IP "socket" interface that can be implemented at the application level. SSL has several characteristics that can be used by an e-commerce application to communicate with large numbers of users via common Internet browser software:

1. SSL can be used to identify and authenticate (with a high degree of trust) server applications to users of the application. An example of the importance of this function is a Web marketing application requesting a customer's credit card number. The customer wants to be assured that the application is the one intended and not a "Trojan horse" imposter that is stealing credit card numbers. SSL provides this highly useful security function.

2. The second function provided by SSL is establishment of a private cryptographic communication channel through the Internet between the communicating parties, for example the customer on the Internet and the Web application server. The value, obviously, is that now credit card numbers may be passed from the customer to the marketing application without being observed by a hacker. Note that virtual private networks (label 7 in Figure 1), which we will discuss later, also provide Internet communication privacy. The difference is that VPNs are generally used for point-to-point communication sessions that may be shared by multiple applications and that continue for a longer duration than those typically seen with common browsers.

3. The third function provided by SSL is the ability for the users of applications to be identified and authenticated (with a high degree of trust) to Web application servers. This function is most useful when the application provides function that requires the user to be known to the application. An example is when the application provides administrative support functions that need to be individually authorized for execution by only specified individuals, such as payroll functions.

SSL has emerged as the preferred method for securing Web-based transactions on the Internet. It is widely supported in browsers and most Web servers. The first two functions are in common use worldwide, while the third—user identification and authentication using X.509 version 3 digital certificates—is being applied in an increasing number of cases.

User identification via SSL with X.509 version 3 digital certificates on z/OS brings with it some highly advanced function. For example, although it is possible to have an individual X.509 version 3 digital certificate (representing an individual user) that maps to (is associated with) only one RACF user ID, it is equally possible to have multiple digital certificates (representing multiple individual users) map to a single RACF user ID. In this case the RACF user ID is shared by all of the users who are mapped to it. Individual accountability is not lost, however, because the individual user-identifying information from the X.509 version 3 digital certificates is retained in the ACEE credential and is included in any auditing (or logging) records created by the process. In addition, the individual identifying information is available to the application should it be required. RACF user IDs that are shared in this way, perhaps by a large number of users accessing the computing system via the Internet, should be restricted in access authority such that these user IDs can access only resources to which they have been specifically authorized and access nothing by default. RACF provides such capability via its restricted-user attribute on user ID definitions.

SSL represents the single most important user of cryptography in the spectrum of secure e-business applications. Over the past few years we have focused on encryption performance, especially in support of SSL, and we have added a performance-optimized system SSL service in z/OS that utilizes S/390 and zSeries integrated cryptographic hardware to handle complex cryptographic operations. As a result we have increased z/OS SSL performance 154-fold (from 13 SSL session handshakes per second in 1998, to 2000 per second at the beginning of 2001).

Kerberos on z/OS. Another form of trusted third-party user identification and authentication is Kerberos (see label 4 in Figure 1). The fundamental difference between Kerberos and trusted third-party identification and authentication with digital certificates is the method of encryption used. As previously discussed, X.509 version 3 digital certificates technology is based on public key technology—more specifically on "asymmetric" cryptography in which two mathematically related keys are used. Kerberos, instead, is based on "symmetric" cryptography, in which the same key is used to both encipher and decipher information. Therefore, before the user can communicate privately with any server, the user must first approach the Kerberos server, be authenticated, and be passed a cryptographic key with which to speak to the target server. Thus Kerberos depends on the active participation of a Kerberos server, called the key distribution center (KDC), during the actual identification and authentication process.[4] Since the KDC must be available for identification and authentication to occur between two Kerberos parties, Kerberos implementations work most efficiently when configured as part of a local area network (LAN) of users and servers.

Recent enhancements to the Security Server for z/OS include an implementation of the Kerberos user identification and authentication functions as well as support for related secure information-passing cryptographic tools implemented according to the GSS-API[5] (Generic Security Services Application Programming Interface). A unique feature of the z/OS Security Server Kerberos support is that the required Kerberos registry of users, along with the user passwords, is built on top of the existing RACF registry of users. That is, the required Kerberos information for each user is added to the existing RACF information for existing users, so an entirely new registry (another list of the same users) is not required, and administrators do not have to learn the administration user interface of another system. In the same vein, a user's Kerberos and RACF passwords are the same, so there are no password synchronization problems.

System Authorization Facility

Approximately 25 years ago IBM developed the Resource Access Control Facility to provide centralized security functions such as user identification and authentication, resource access control, and auditing for both the operating system and applications running on the system. RACF initially provided these functions via the operating system's "supervisor call" facility, enabling programs written in basic assembler language to invoke security functions via assembler macros. RACF existed as a separate product, which customers could choose to license and install if they wanted to provide better security for their systems. Other vendors provided competitive products that a customer might install instead, though some

customers chose not to install a separate security product.

Approximately 16 years ago, in an effort to improve the usability of the system's security interfaces and thus encourage more applications to use consistent system-level security rather than implement their own separate security mechanisms, IBM implemented the System Authorization Facility (SAF) within the MVS operating system. SAF combined the various security function invocations into a single extensible security mechanism using a new macro instruction named RACROUTE, which provided subfunctions for each of the previously separate assembler language security macro instructions. Thus, a programmer could use a RACROUTE instruction with the REQUEST=VERIFY subfunction to authenticate a user's identity, or with REQUEST=AUTH to see if an authenticated user is allowed to access a requested resource.

This new security structure provided several benefits to encourage use by application programs and components of the operating system itself:

1. SAF made it easier for application developers to find information about the available security functions, because instead of trying to locate information on several (six, at that time) separate security macro instructions, the application developer could instead look at information about one instruction.

2. SAF made it easier for an application developer to interpret the results of a security operation. Rather than needing to handle many different results from many different security functions, with SAF and RACROUTE an application usually needed to handle only three consistent results from any security request. The request would return either a value of "0" to indicate a success, a value of "8" to indicate a failure (lack of authority), or a value of "4" to indicate either that the system did not have a security product such as RACF installed, or that the security administrator had not provided the necessary information within the security product to handle the application's request.

3. SAF and RACROUTE made it easier to design applications. Applications no longer needed to check for the presence of a security product before making a request for security services, nor did applications need as much code to operate correctly with different, possibly somewhat incompatible, levels of the operating system or security product.

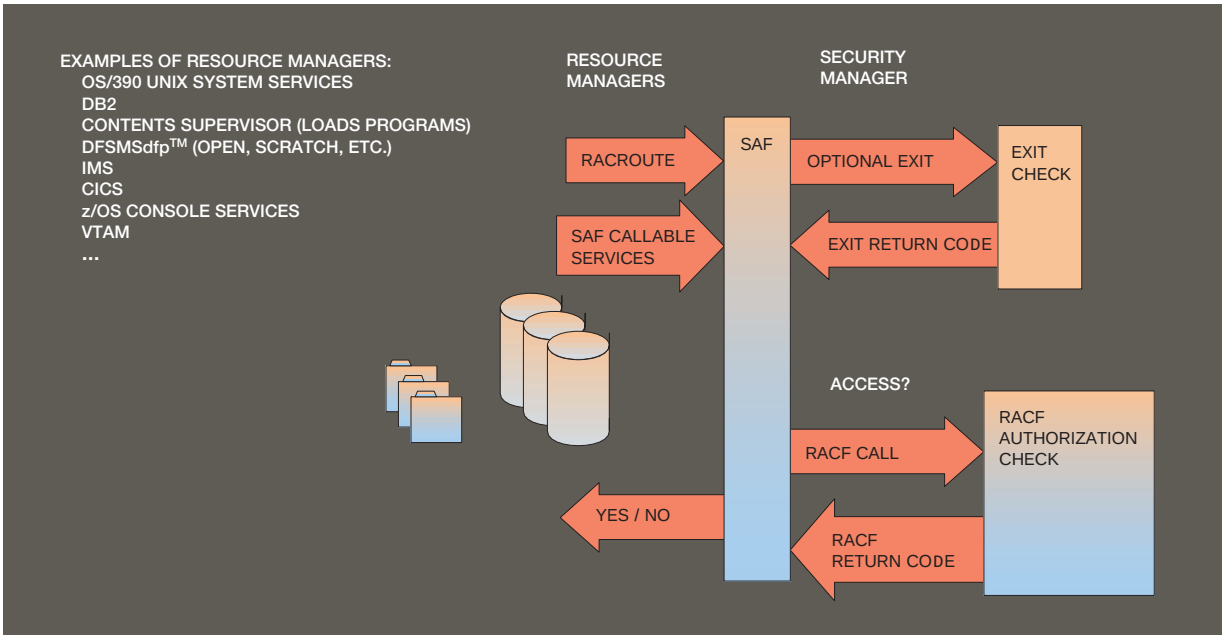
SAF also provided direct benefits for IBM's MVS customers by providing a central mechanism (an exit point or hook within SAF itself) for customized security-request processing. For example, with customer-written exit code, they could process requests in their own way, assisting their security product to make the proper decision or replacing the results that the security product would otherwise provide. SAF also provided indirect benefits for customers, because security management became simpler as more applications used centralized security services rather than provide their own security mechanisms.

Figure 2 illustrates how SAF relates to resource managers executing within z/OS and a security manager such as RACF. SAF is invoked by resource managers at system security control points, then SAF invokes the installation-selected security manager, for example RACF. The authorization decision is passed back to the resource manager by RACF through SAF.

Initially SAF provided only the RACROUTE macro and a thin layer of framework code to route security requests first to the optional customer-provided security exit and next to the optionally installed security product, RACF or another. In later years, as MVS and its security capabilities improved further, IBM and RACF provided several additional security functions via the RACROUTE interface and actually implemented some of these new security functions within the SAF framework. Thus, SAF might satisfy some security requests by itself or handle some jointly with the security product, though for most functions SAF merely routed control to the security product to provide the security services needed by applications.

Most recently, IBM has again modified SAF to provide additional security interfaces for the UNIX environment on z/OS. Again, SAF acts primarily as a routing facility, allowing customers to provide an exit routine that may examine or modify an application's security request, and then as needed route the request to RACF or another security product. However, for this new function SAF no longer uses the RACROUTE macro, but rather provides various callable services usable by applications written in a wider variety of programming languages, such as C, C++, and Java, extending the centralized security aspects of z/OS to accommodate newer kinds of applications, such as servers written for UNIX environments.

Figure 2 System Authorization Facility (SAF)



Access control

We have so far explained how the identification and authentication of users is implemented within the Security Server optional component of z/OS, and we have discussed the role of the System Authorization Facility. Having prepared the groundwork, we turn our attention to *access control* (see label 11, Figure 1), which is the discipline of controlling access to valuable and sensitive enterprise computing resources, and therefore the central mission of any logical access control manager.

Stated simply, access control is the process of determining *who* has access to *what*. Identification and authentication of users provides the "who"; the "what" is provided by information about protected resources, including the rules for access, which are maintained within the data store of the access control manager. During run time, when a user attempts to access a protected resource, a data file for example, the system component (resource manager) that supports use of the resource invokes the system security manager (i.e., RACF), requesting an access control decision. The security manager uses information about the user and the protected resource to determine if the user is authorized to access the resource, and returns the decision to the requesting resource manager, which subsequently allows or denies access to the resource.

In making the access control decision, the security manager will often consider additional factors, such as how the user intends to use the resource, to *read* the information, for example, or perhaps to modify (*update*) the information. In addition, modern security managers like RACF support a variety of access intentions including *execute* (which is functionally different from read) and *scratch* (z/OS jargon for delete).

Access intentions can be thought of as a *condition* that exists at the time the access control authorization check event occurs. But there are other conditions that may be in effect at the time that may influence the decision. One example of such a condition is *time*, represented as time of day, day of the week, or before or after a pre-established point in time. Another example is the physical location of the user when the access attempt is made. In addition, the method by which the user accesses the system and identifies himself or herself may determine the user ID on which the access control decision is based. Depending on the resource, these capabilities are implemented in one form or another with the z/OS Security Server.

With modern security managers, access rules and associated conditions are stored in data entries known as *access control lists* (ACLs). ACLs are usually stored in the data repository of the security manager in informational records, which, in the case of the RACF component of the z/OS Security Server, are called *profiles*. In general, such profiles are logically associated with the actual resource by way of the name structure of the profile. That is, the name of the resource is used as a key to look up the associated profile in the access control list profile registry. This level of abstraction, between the actual resource and the access control list profile associated with it, is important because it supports the creation of a profile that exactly matches the name of one and only one resource, called a discrete profile, and also supports the creation of a so-called generic profile, which can cover multiple resources that share a common naming structure.

Generic profiles are valuable administrative aids, because with them many similar resources can be controlled with a minimum of administrative effort. Generic profiles also benefit overall system efficiency by allowing the total number of individual entries in the profile registry to be kept to a minimum for a given number of protected resources.

The RACF component of the z/OS Security Server also supports the concept of *user groups*, which are logical structures in which multiple individual users are associated within a single named user group. The named user group, rather than each individual user, is then represented within the access control list of either a discrete or a generic profile. The result is administrative efficiency, because many users can be assigned a common access level to a resource (or generic set of resources) by a single access control list entry. A recent enhancement of the group concept is the notion of a set of users who share a common function or *role* within the enterprise. The role is a special case of the user group, in which the criterion for associating a given population of users with a particular group is the specific job that the users do.

Still another important aspect of access control, specifically the administration of access control lists, is the separation of *access rights* that users have to the resources from *administrative rights* that administrators may have to the access control lists. A user with access privilege to a resource may or may not have privilege to the access control lists that define privilege to that resource. Conversely, administrators do not have access privilege to a resource just because they have administrative privilege. The RACF component of the z/OS Security Server sorts out this complex area by allowing system administrators to be specifically denoted at both system and group levels. These administrators have the power to create and modify access control lists according to predefined and documented rules. Auditing of administrative action, which will be discussed in a later section, discourages administrators from giving themselves improper access to resources, because such access would be automatically recorded and made available for later scrutiny by separate personnel.

Another fundamental aspect of access control technology that is supported by the RACF component of the z/OS Security Server involves the ability to make access control decisions based on the compartmentalization of computer resources and information. Compartmentalization is sometimes referred to as *multilevel security* or MLS, and can be very important to government groups, especially military-related computing installations and networks. MLS involves the association of certain abstract qualities to both resources and users. These can be hierarchical in nature, in which case they are referred to as *security levels*, or flat (not hierarchical), in which case they are referred to as compartments or *security categories*. Further, security levels and categories can be grouped together into constructs called *security labels*. In short, resources that are assigned (by security administration) particular security level(s), category(s), or label(s) can only be accessed by users who also have been assigned the same security level(s), category(s), or label(s).

Now that we have discussed the basic concept of access control lists and access checking, we should consider two additional key points about the security architecture provided by z/OS and its predecessor OS/390. We briefly discussed "resource managers" or "access control managers" at the beginning of this section, and we have referred repeatedly to the concept of "resources" without defining exactly what constitutes a resource. We now describe resource managers and their resources more completely and indicate how they function in the system.

System resource managers perform access checking automatically as users try to access the resources they support. z/OS, the z/OS Security Server, and the system-supplied resource managers delivered with z/OS can protect many different IBM-defined categories (classes) of resources (over 125 as shipped by IBM to its customers). These include MVS data sets, UNIX

files, CICS and Information Management System (IMS*) transactions, executable MVS programs, terminals and workstations, VTAM (Virtual Telecommunications Access Method) logical units, printers, direct access storage device (DASD) volumes, etc. Some of these resource classes hold physical objects (e.g., a data set or a terminal), whereas others may be logical objects (e.g., an administrative authority allowing a user to reset some other users' passwords, or the ability for a system operator to issue a particular command during the operation of z/OS). For all of these classes of resources, the system-supplied resource manager will automatically check the user's authority to the resource, without the customer needing to do any programming.

SAF services handle relationships between users, groups, and resources, removing this complexity from the application.

In general, customer application programmers do not need to concern themselves with implementing security within their applications. Most applications will run in an application environment (batch, TSO [Time Sharing Option], CICS, WebSphere Application Server, HTTP server, etc.) where the environment itself, or the system-supplied resource managers, will perform all the necessary identification and authentication functions and all the necessary resource access-checking functions. The customer application normally just runs on behalf of an identified user and accesses system resources as it operates on its data, and all the security functions happen automatically.

However, in the rare cases where a customer or a software vendor does need to write an application that will make some security decisions (though still subject to overriding control by the system-supplied resource managers, of course, when dealing with system-supplied resources), the application can make use of the SAF services to make those decisions. The application programmer can determine a naming convention for the resources controlled by the application and a name for the entire class of resources. Then, using the functions of the z/OS Security Server, the security administrator can define the resource class to the system. At that point, the security administrator(s) can define resource profiles in the class, using the resource names determined by the application programmer, and the application program can use the RACROUTE REQUEST=AUTH service to request that the z/OS Security Server make an access check to determine if the user running the application has the requested authority to operate on the resource. (Note: In some application environments, such as CICS, IMS, Java, or WebSphere Application Server, the application must use different services ["wrapper" functions] to make the security check. Those services, then, will invoke the SAF RACROUTE REQUEST=AUTH function "under the covers" as necessary to perform the actual authorization check.)

In even rarer cases, application programmers may create a server application that, in addition to acting as a resource manager and performing authorization checks, also needs to authenticate its clients (users). SAF also makes that easy to do, via RACROUTE REQUEST=VERIFY and a number of wrapper functions that make it easier to perform user authentication in various application environments.

Security services exploitation

With the introduction of the System Authorization Facility suite of services, a centralized point was created within the operating system infrastructure through which both operating system components and applications could invoke the services of the z/OS resident security manager. This security framework allowed application developers to focus their attention on the placement of the appropriate security-related calls within their application code, and to examine a small set of return codes (status) of a call to determine the result of the security-related request.

The SAF services can be thought of as an encapsulation layer in which the complexities of the relationships between users, groups, and resources that users access are effectively managed by the security manager, removing this complexity from the application. This programming model has been adopted by major z/OS subsystems, such as the CICS Transaction Server (TS)[6] for z/OS, which provides high volume on-line transaction management. CICS uses the z/OS security services offered by SAF and the RACF product to identify and authenticate the originator of a transaction, and to authorize the use of customer-implemented transactions. Similarly, IMS,[7] a hierarchical database and transaction manager, uses SAF and RACF services for user identification and authen-

tication and to mediate a user's access to an IMS transaction.

The transaction processing environment and many components of the z/OS operating system have strict performance requirements; in many cases security-related decisions and functions are in the critical path to completing the transaction or operating system function. The SAF and RACF services have been tuned for efficiency through path-length studies to meet critical performance objectives. The scalability of these services, in both frequency of requests and volume of users and resource definitions, also has been honed, which is in part reflected in the scale of transactions that are sustainable by the z/OS transaction processing environment.

Strategies such as caching and data structure optimization, while maintaining coherency of cached security data, have been effectively used in meeting performance objectives without compromising the integrity of the security decisions provided by the security infrastructure. The performance and scalability goals are examples of the quality of service offered by the mature z/OS security components. The focus on scale has allowed z/OS applications to use the z/OS security infrastructure to centralize the security administration and security policy enforcement within the security product and the enterprise security administrators.

With the advent of UNIX services, a new class of "ported" applications has emerged on z/OS. These applications are typically developed in C, and the use of assembler language macro interfaces to security services would be inappropriate. Responding to the security needs of ported applications, the security services exposed by SAF have been enveloped by C language functions, giving the distributed application developer a familiar programming model. These services provide the same security capability to ported applications as the native security services do for applications implemented using macro interfaces.

As the application development world began to adopt Java as a portable application development language, z/OS and the security services provided by z/OS continued to evolve. As was done for ported UNIX applications, z/OS provided Java class libraries that "wrap" the native platform security services. We expect that as the Java environment continues to mature, with Java application run-time environments provided by WebSphere Application Server and Java virtual machines, the security services provided by the SAF component will keep in step, providing the basic infrastructure that allows mixed application workloads on z/OS to benefit from the centralized security services for run-time security decisions, user identification and authentication, auditing, and administration.

Auditing and logging

One of the most important features of a centralized authentication and access control mechanism such as RACF is the ability, from a single focal point, to record and analyze security information. This information, called audit data, is essential for ensuring that the customer's installation security policy is being followed. Proper monitoring of the installation requires the analysis of two types of audit information: static information, such as the user registry and access information, and dynamic information, which is a record of security-relevant information such as system or resource access.

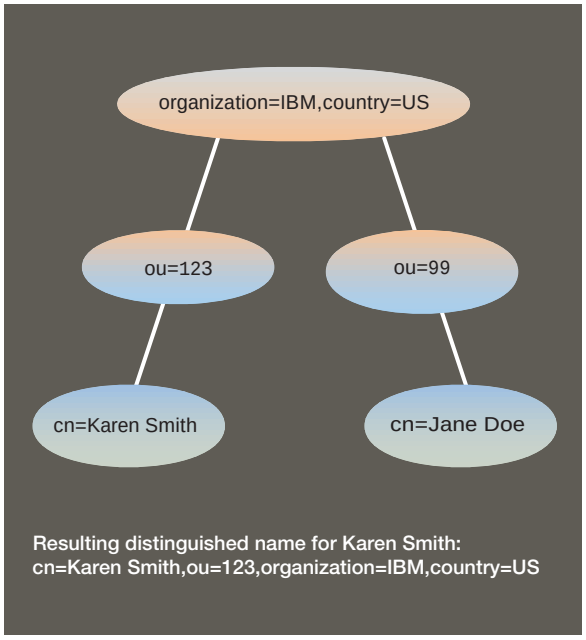
The RACF option of the z/OS Security Server provides multiple ways to specify what security-relevant events are recorded in the audit stream and how that information is reduced and analyzed.

In the case of resources, the RACF security administrator or a resource owner (as designated by RACF) may request the auditing of events related to a specific RACF-controlled resource (or set of resources). Similarly, the RACF auditor may request auditing of events related to resources, but independently of the security administrator and the resource owner, that is, entirely at the discretion of the auditor and without the owner necessarily being aware of the audit. Auditors may request the auditing of entire RACF classes of resources.

In auditing users, the RACF security administrator or the RACF auditor can cause the activities of a specific user to be logged. Such users need not be aware that their activities are being monitored.

Finally, RACF can enforce a separation of duties with respect to security administration and auditing by allowing the administrator to change access control rules and user definitions but not to change certain global auditing controls. Conversely, the auditor can change auditing controls but cannot make administrative changes. In other words, the administrator cannot prevent the auditor from auditing the administrator, while the auditor cannot authorize himself or herself to resources. The result, when this capa-

Figure 3 Sample directory tree



bility is properly exploited, is that collusion between the two is required to cheat the system.

RACF provides several utilities to assist security administrators and auditors in the analysis of security information. These include:

- The RACF database unload utility, which translates the RACF database (user, group, and resource registry) into a format that can be loaded into a relational database, such as IBM's DB2*, or processed using a report writer, such as the ICETOOL facility in IBM's DFSORT* program product
- The RACF SMF unload utility, which translates the RACF event log—Systems Management Facility (SMF) data—into a format that can be loaded into a relational database or processed using a report writer
- The RACF data security monitor utility, which interrogates and reports on the configuration of security within the z/OS environment
- The RACF remove ID utility and RACF cross-reference utility, which find references to users in the RACF registry

Each of these utilities provides the auditor with data analysis and reduction capabilities to ensure that users are adhering to the installation security policy.

Directory services

While not always thought of as such, there are several databases on z/OS that can be used as directories. The RACF registry is itself a directory for RACF information, including users and groups. The Distributed Computing Environment (DCE) on z/OS also has a directory that supports DCE definitions. And, the Lightweight Directory Access Protocol (LDAP) server is a general-purpose distributed directory server implemented within z/OS (see label 9 in Figure 1) that can contain many different types of information, from distributed application descriptions, to configuration information, to user and group definitions. The z/OS implementation of LDAP is designed to be complementary to RACF and to interoperate with it in support of the integration of the centralized computing model, traditionally supported by RACF, into the emerging distributed computing models, such as those defined by the Common Object Request Broker Architecture (CORBA**) and the Enterprise JavaBeans** (EJB**) environment provided by WebSphere.

Work has been done in recent releases to bring together some of the information associated with users. For example, RACF allows DCE principal information for a user to be kept in the RACF registry. In addition, it supports mappings between a user's RACF user ID and other identities, including Kerberos principal name and X.509 version 3 digital certificates. Some middleware applications have been adapted to use the information stored in RACF, and others are expected to do so in the future.

LDAP adds another element for consideration. While identities in RACF are based on user ID, identities in LDAP are based on distinguished names. A distinguished name includes a specific identifier for an entity in the directory, as well as identifiers that determine that entry's position within the directory. An LDAP directory is a hierarchical representation. Figure 3 shows a simple sample tree to demonstrate how a distinguished name is built.

The distinguished name for an entry is created by starting with the entry's own name and adding to it all of the names of entries in the hierarchy that have to be traversed to reach that entry. So, a person, Karen Smith, who works in department 123 in IBM in the United States, might have a distinguished name that looks like this:

Figure 4 RACF example using LDAP command

```
ldapsearch -h 127.0.0.1 -p 389 -D bindDN -w passwd
  -b "racfid=kareng,profiletype=user,cn=plex1,o=IBM,c=US"
  "objectclass=*"

racfid=kareng,profiletype=USER,cn=plex1,o=IBM,c=US
objectclass=racfUser
...
racfid=kareng
racfauthorizationdate=99.134
racfdefaultgroup=racfid=GOODGUYS,profiletype=GROUP,cn=plex1,o=IBM,c=US
racfattributes =SPECIAL
racfrevokedate=NONE
safaccountnumber=75932
racfomvsuid=0
racfomvshome=/u/kareng
....
```

cn=Karen Smith,ou=123,organization=IBM,
country=US

while her coworker in department 99 might have a distinguished name that looks like this:

cn=Jane Doe,ou=99,organization=IBM,
country=US

The information that can be kept in an LDAP server for a user is based on definitions of what can be stored in a person entry. There are several definitions available for person entries, some defined by industry standard bodies, such as the Internet Engineering Task Force (IETF), and others defined by companies or applications. These definitions are referred to as "schema." Although other databases use the concept of a schema to define their data, these definitions tend to be fixed, not customer-definable, and not modifiable.

The advent of the LDAP server and various middleware and other applications that use that server give rise to another identity for a user. On z/OS, the LDAP server provides read and update access both to its own database, where "person"-style data are stored, and to RACF's database, where "user ID"-style data are stored. There is some overlap between these definitions: the LDAP person schema defines a "UID" attribute that is commonly used to store a user ID, and it also allows for a password in the user-password attribute. However, authentication to the LDAP server is based on the combination of the distinguished name and password rather than on the UID and password.

The remainder of the data kept by LDAP and RACF is very different. As a result, a schema has been developed and is provided by the z/OS LDAP server for the user and group information in RACF. The "racfuser" definition represents a RACF user's base segment,[8] with additional schema definitions for each RACF segment that can be associated with a user. The "racfgroup" definition does the same for a RACF group. With these definitions, or schema, and support in the z/OS LDAP server, it is possible to administer RACF user and group information using LDAP API calls. There are several Web-based LDAP directory tools that allow both browsing and updating of the RACF users and groups. Access to this information is controlled using an LDAP distinguished name that corresponds to a RACF user ID; in this way, inappropriate accesses are avoided while the use of a distinguished name identity for authentication to LDAP is retained. Figure 4 shows an example search.

In addition to the RACF complementary identification and authentication functions in support of distributed applications that we have just described, the LDAP server can also be used to store many different types of information. Also, the LDAP protocol allows

the data definitions (of what an LDAP server stores) to be updated dynamically, which makes it easy to adapt an LDAP-capable directory to store information needed for many applications. As a result, many middleware offerings and applications, such as the IBM WebSphere Application Server, IBM Host On-Demand, and Tivoli Policy Director, rely upon an LDAP-capable directory server to store information necessary for configuration, run-time operations, or both. The z/OS LDAP server can be used as the data store for these offerings.

The LDAP server's role in the security environment is one of an easily accessible data store for security information ranging from X.509 version 3 digital certificates to access control policy information. The LDAP protocol provides an industry-standard access mechanism to these data. On z/OS, the LDAP server provides the ability to extend the native security services provided by RACF with distributed security capabilities provided by cross-platform applications and services.

When the information is stored in the LDAP server, access to that information is controlled by the authentication identities discussed earlier. The LDAP server includes an access control list capability that is used to administer and control access to information stored by the server. These access control lists are separate and distinct from RACF's access control. The LDAP server's access control lists are built using distinguished names for individual users or groups, and their scope is limited to controlling access to information accessed through the LDAP server. In contrast, RACF's access control, which is based on user IDs, provides control over a wide range of resources on the z/OS system, as well as control over access to the data stored in RACF's own database.

Because the z/OS LDAP server supports RACF-style distinguished names as well as LDAP's ePerson-style distinguished names in access control lists, the security definitions established in RACF can be extended to the LDAP server. Access control in the LDAP server is established for each individual entry within the server. Access control lists can be set at any level within the hierarchy of information and can be propagated down through the tree to apply to all entries below. Access is controlled at the attribute level within each entry by assigning each attribute to an access class, where the access classes are defined as normal information, sensitive information, or critical information. These classifications are part of the data definition (schema) for the information stored by the LDAP server. For each access class, various permissions can be established: read, write, search, and compare. So, for any given entry ("object") within the hierarchy, an access control list can be established by an administrator that lists the distinguished name of a user or group entry ("subject") and the subject's permissions ("rights") to the data represented by the object.

By using the RACF-style distinguished name that represents a group defined in RACF within the access control list for entries within LDAP, the RACF group definitions can be extended into the LDAP server to control access to the entries stored by LDAP.

Figure 5 shows an example using the simple sample directory shown in Figure 3. The access control list-related attributes within the entry are highlighted (in blue). Access to an entry and the entries below it in the hierarchy are controlled by these attributes. Figure 5 highlights the access control for organization unit (ou) 123, which corresponds to department 123.

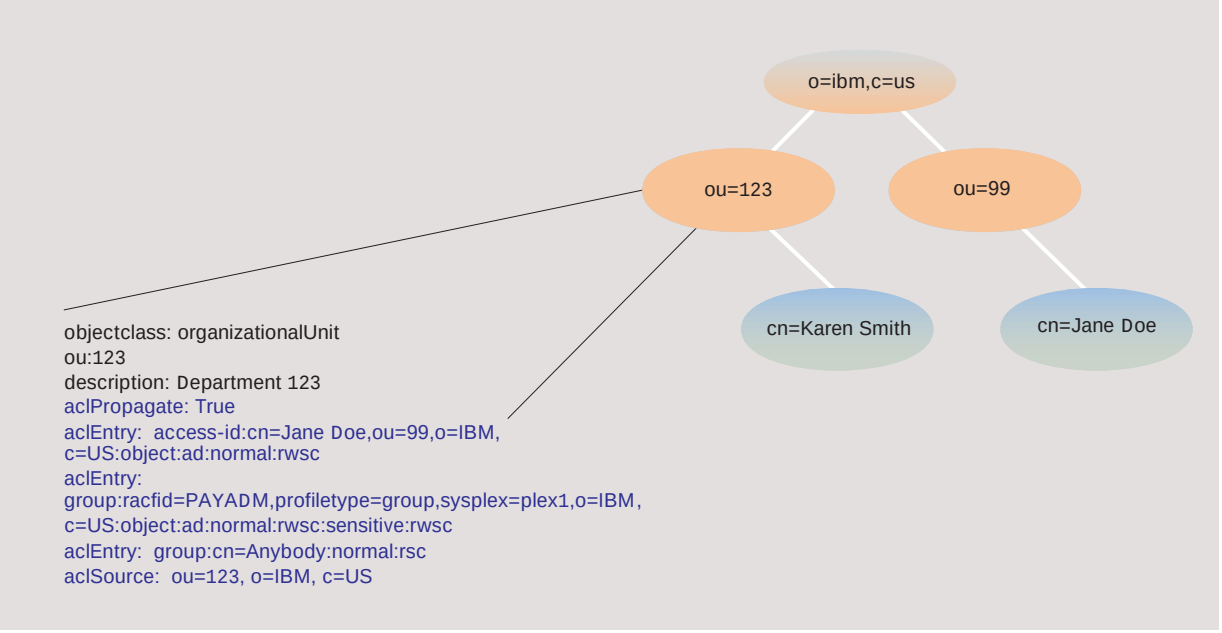
First, note that the "aclPropagate" value is set to "true." This means that the access control established for this entry will apply to all entries below it in the tree, including Karen Smith's entry as well as any other people in department 123.

The "aclEntry" attribute is where a subject's rights are defined with respect to this entry. In this example, two specific subjects, Jane Doe and PAYADM, and one special subject are given access rights. Suppose that Jane Doe in department 99 needs to be able to update the directory to add people to department 123, but she can only enter nonsensitive data. The aclEntry for access-id:cn=Jane Doe establishes just those rights for Jane Doe.

Suppose also that there is an already-established RACF group called PAYADM that contains the RACF user IDs of several payroll administrators. They need to be able to add new members to department 123; also, they need to update both normal and sensitive information, but not critical information. The aclEntry for group:racfid=PAYADM provides these rights. When a subject with user ID PAY1, who is a member of group PAYADM, authenticates to the LDAP server, the group PAYADM will be associated with PAY1's authentication. When access control checking occurs, PAY1 will get the rights established for the PAYADM group.

The special subject defined here is aclEntry:access-id:cn=Anybody. This provides anonymous or "any-

Figure 5 ACL example for sample directory tree



body" access to the entry. By default, a caller that does not authenticate itself to the server receives anonymous access to data within the server. The default access for "cn=Anybody" is read, search, and compare of only normal attributes. This default can be modified to remove all anonymous access to the data in the LDAP server.

From the examples, one can see that it is possible to "mix and match" distinguished name styles within access control information in the z/OS LDAP server. This provides the opportunity to extend already-established RACF administration rules to the LDAP directory.

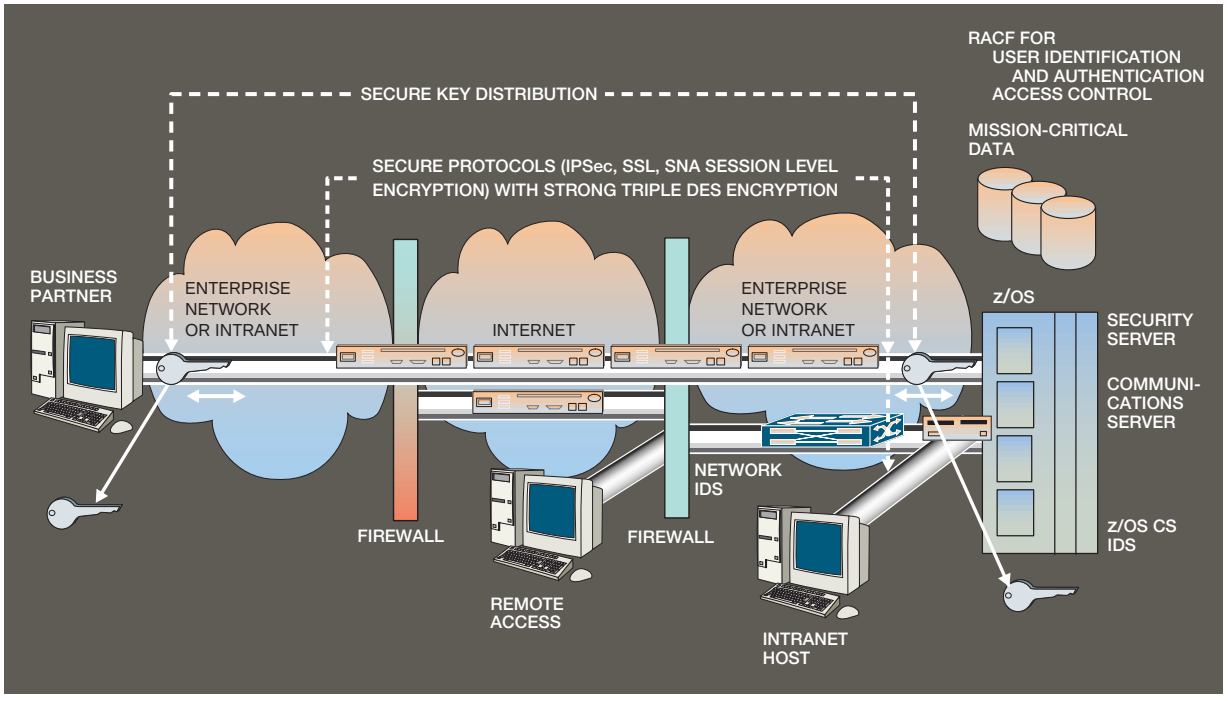
Networking and communications security

Networking and communications security on zSeries and S/390 systems is provided by the *Communications Server* element of z/OS and OS/390. The Communications Server provides networking and communications services for accessing z/OS applications over both SNA[9] and IP networks.

Traditionally, SNA networks have been considered very secure. With subarea SNA networking,[10] every remote resource in the network had to be predefined to VTAM before it could engage in communications, giving administrators tight control over the use of network resources. Advanced Peer-to-Peer Networking (APPN)[11] changed this by adding more dynamics with reduced network definitions. Techniques such as LU 6.2[12] session security, LU 6.2 conversation security, and CP-CP[13] session security were developed to improve peer-to-peer authentication. In order to protect sensitive data in the network, VTAM provides privacy services using SNA session level encryption, and data authentication and integrity services using SNA message authentication code checking. Both of these services use the Data Encryption Standard (DES) and Triple DES[14] symmetric encryption algorithms.

With the advent of the Internet and TCP/IP, the network security requirements have become more stringent and complex. Because many transactions come from untrusted networks such as the Internet, and sometimes from unknown users, increased attention is paid to host and user authentication, data privacy, data origin authentication, and data integrity, as well as denial-of-service attacks. In addition, there are certain applications shipped with TCP/IP such as File Transfer Program (FTP) that, without proper configuration and access controls in place, could allow

Figure 6 Providing network security



unauthorized users access to system resources and data. The Communications Server, along with other elements of z/OS, provides security functions to address these TCP/IP security concerns (see Figure 6). These include the following:

- *Protecting data in the network*. The Communications Server protects data in the network using secure protocols based on cryptography, such as IP Security, SSL, and SNA session level encryption.
- *Protecting system resources and data from unauthorized access*. Communications Server applications and the TCP/IP protocol stack protect data and resources on the system using standard RACF services.
- *Protecting the system from the network*. The Communications Server is responsible for protecting the system against denial-of-service attacks from the network. The Communications Server has built-in defenses and also provides several services that an installation can optionally deploy to protect against these attacks.

Protecting data in the network. As SNA networks are replaced by IP networks, the TN3270 protocol provides a means for client workstations to continue to access z/OS SNA applications over an IP network. With TN3270, it is transparent to the application that an IP network is being used. In order to protect data over the IP portion of the network, the Communications Server has enabled its TN3270 server to use Secure Sockets Layer. The SSL-enabled TN3270 server identifies and authenticates end users using digital certificates, then using RACF services, maps their X.509 version 3 digital certificates to a corresponding RACF user ID. Once the RACF user ID is known, the end user's authority to access the TN3270 server is verified. The IBM Host On-Demand program product, which provides an SSL-enabled TN3270 client, exploits this function to make traditional CICS- and IMS-based interactive applications available to users over the Internet.

As TCP/IP applications are rolled out on z/Series or S/390 processors within the enterprise, security in the IP network may be required to protect data traffic, depending on the sensitivity of the data and the level of trust that the installation has in the IP network. If the z/Series or S/390 application is not enabled to use SSL or Kerberos, IP Security (IPSec)[15] can be

used. Furthermore, as enterprises seek to engage in business-to-business or same-business communications using the Internet as a portion of the data path, IPSec can be used to protect not only the application data but also the IP header information. IPSec can be used to build a virtual private network to support these e-business configurations.

IPSec, an IETF-defined standard, provides data privacy, data origin authentication, and data integrity services. Since IPSec is implemented at the network layer rather than in the application (as is the case for SSL), it can transparently protect all applications without requiring applications to be changed. IPSec protects data traffic using security associations, which provide cryptographic security services for the data traffic they carry. Security associations and associated cryptographic keys can be manually defined, or they can be dynamically and securely created using the Internet Key Exchange (IKE) protocol.

In addition to IPSec, the Communications Server supports network service programs that have security built into the application protocol, such as Simple Network Management Protocol version 3 (SNMP v3).[16] In an announced future release of the z/OS Communications Server, additional secure network services such as Open Shortest Path First (OSPF) MD5 Authentication[17] and Secure Domain Name System (DNS)[18] will be supported.

Protecting system resources and data from unauthorized access. Communications Server applications use RACF for identification, authentication, and access control decisions. Some applications allow access by an end user that is not identified and authenticated. These applications must be specifically configured to permit anonymous access. When such access is allowed, the resources that can be accessed are limited by server configuration, or by using a specific RACF user ID (with limited access privilege) assigned to the anonymous user(s). By using a RACF user ID to represent the anonymous user, the application can access only the resources permitted according to the installation-defined RACF policy for anonymous users. Figure 7 shows, for a representative set of applications, whether user identification is required and whose credentials are used when resources are accessed.

In addition to the TCP/IP application support of RACF, the Communications Server TCP/IP protocol stack uses RACF to control local z/OS or OS/390 users' access to TCP/IP resources, such as a specified TCP/IP system, TCP or UDP (User Datagram Protocol) port, or IP network resource (or group of IP network resources). Other protections are available, such as syslogd[19] isolation, which provides a method for segregating system- and user-level syslog records based on user ID and job name. It also records the real user ID and job name in the syslog record to prevent undetected "spoofing."

Protecting the system from the network. The z/OS Communications Server has built-in defenses to ensure high availability of the system against denial-of-service attacks from the network. The installation can request further protection by configuring for IP packet filtering. IP packet filtering can permit or discard inbound or outbound packets by matching configured selectors such as IP address, port, and protocol with information in the IP packet. A traffic regulation management daemon (TRMD) can be configured to control the number of connections allowed on a TCP port. The number of connections can affect system resource consumption, such as memory and number of address spaces. TRMD protects the system against a spike in the number of connection requests.

Intrusion detection systems (IDSs) are used to detect potential intrusions and attacks on the network or a host in the network. There are many types of intrusion detection systems, each with its own advantage. In general, an installation should incorporate multiple IDS types from multiple vendors in order to broaden intrusion-detection coverage. In an announced future release the Communications Server will provide intrusion-detection services that are integrated with the TCP/IP stack. This approach has a number of strengths when compared to intrusion-detection systems deployed in the network. These strengths are based on the IDS's exploitation of its location, which is the communications endpoint. The IDS can examine data encrypted end-to-end by using IPSec after decryption. The IDS also has access to information unavailable to a network IDS, such as memory and CPU usage, connection state information, internal data queue lengths, and packet-discard rates and reasons. Using this information, the integrated IDS can detect some attacks that are not detectable using a network-based approach. In the first phase of support, the integrated IDS will focus on attacks against the TCP/IP stack. This approach could be extended in the future to include detection of attacks against applications. The integrated IDS, which will be policy-driven using an LDAP-defined schema, can detect scans, single-packet attacks, and floods.

Figure 7 User Indentification and access control for Communications Server applications

SERVER	END-USER IDENTIFICATION	RESOURCE ACCESS
FTP	OPTIONAL (1)	END-USER ID OR CONFIGURED ANONYMOUS USER ID (2)
LINE PRINTER DAEMON	OPTIONAL (1)	SERVER USER ID OR END-USER ID
TRIVIAL FILE TRANSFER PROGRAM	NO	SERVER USER ID (2)
MVS REMOTE EXECUTION DAEMON (REXECD)	REQUIRED	END-USER ID
MVS REMOTE SHELL DAEMON (RSHD)	REQUIRED (PASSWORD OPTIONAL) (1)	SURROGATE USER ID OR END-USER ID
UNIX REXECD	REQUIRED	END-USER ID
UNIX RSHD	REQUIRED (PASSWORD OPTIONAL) (1)	END-USER ID OR SERVER USER ID (EXIT ROUTINE TO VERIFY REQUEST)
UNIX SHELL (TELNET/RLOGIN)	REQUIRED	END-USER ID

(1) All optional items are installation controlled and all can be configured to require full end-user identification.
(2) Files that are accessible can be configured on a server basis to limit access.

Figure 8 illustrates the architecture of the integrated IDS. When an attack occurs, it is recognized by existing error-handling logic in the stack. In the error logic, the IDS policy is checked, and action can be taken based on this policy. For example, event records can be written to syslogd or to the local console, or a sampling of packet traces can be taken. Events written to syslogd are available for off-line analysis, and events written to the local console and syslogd can be used by automation programs to trigger actions and inform policies. It is possible, through future extensions, to send the IDS events directly to an IDS management system without requiring the event records to be recorded to z/OS syslogd or written to the console.

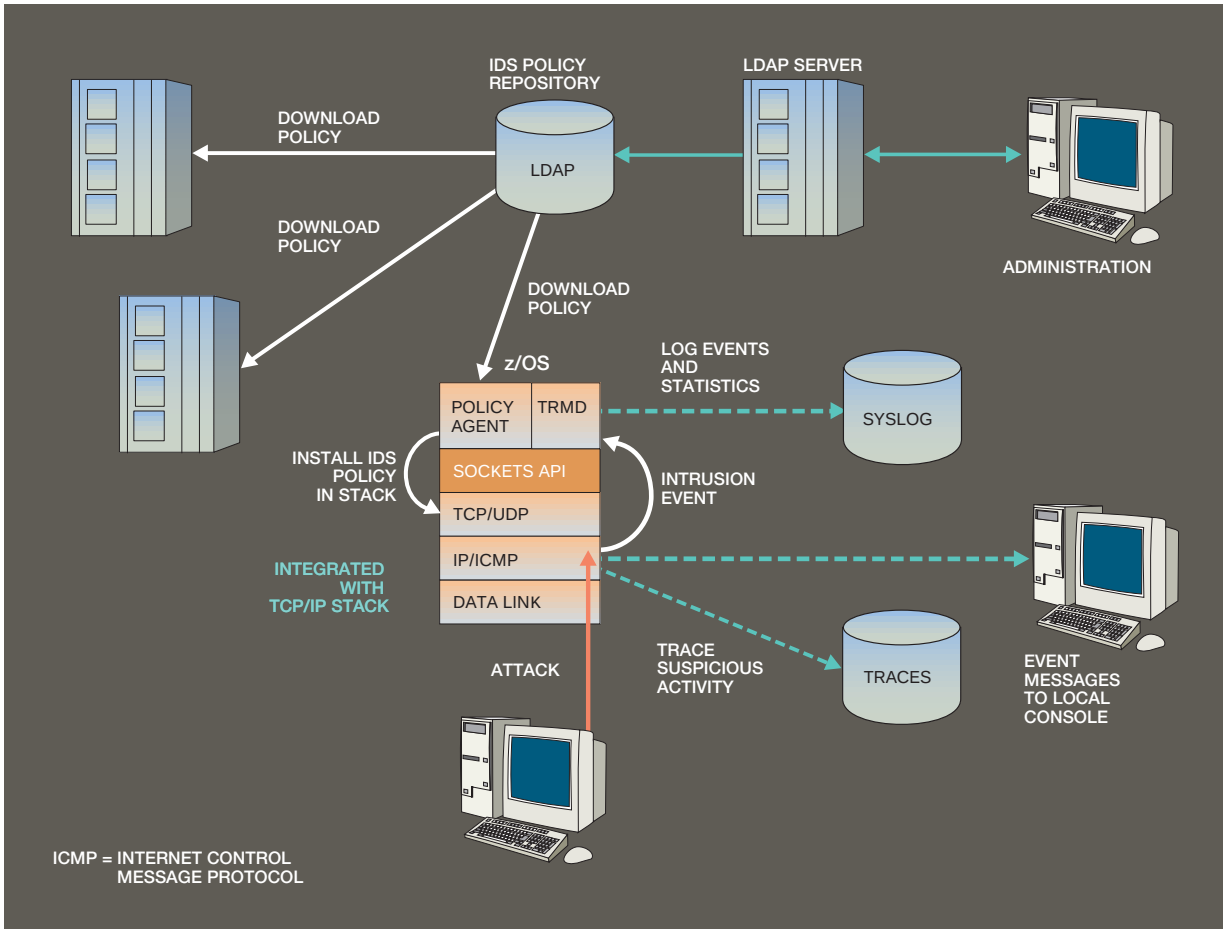
Ethical hacking and certifications

As our customers hooked up their zSeries and S/390 systems to the Internet for the first time, new security issues were emerging. Denial of service and distributed denial of service were key focus areas for us, and to help us, we enlisted the industry experts at IBM Research. With their help, we established a process whereby every release of the z/OS operating system would undergo "ethical hacking" tests. This additional testing provides an added level of confidence for our customers.

We also received an "E4" certification for our S/390 logical partitioning (LPAR) support in 1995 based on the European Commission's Information Technology Security Evaluation Criteria (ITSEC).[20] E4 certification was important; it means that an independent body assessed the isolation of workload from one zSeries computer logical partition to another, and that the isolation is the equivalent of separate physical servers. This gives our customers assurance that one logical partition running a Web environment, and another logical partition running production work, can be configured to be truly separate and isolated while sharing common hardware resources on a single physical server.

Another focus area for us has been encryption hardware certification. As encryption has become a key security tool, industry and country requirements have driven us to provide or work toward these certifi-

Figure 8 Architecture of the integrated IDS



cations. The U.S. government required FIPS 140-1 level 4, and the German digital certificate law required E4 certification. These levels of certification provide confidence to customers that their core encryption keys cannot be captured by a hacker or even an internal systems programmer.

Looking over the horizon

Whether an enterprise has legacy or distributed computing resources, the fundamental mission of the security mechanisms is the same; controlling *who* (*users*) has access to *what* (*resources*). When stated this way, security sounds easy. However, defining the users when there may be large numbers of them, providing methods and mechanisms to authenticate them, and then authorizing their access to potentially large numbers of different kinds of resources during run time, is difficult to do in practice.

It is clear that security function is evolving and application processing environments are changing. For example, the distributed environment adds complexity to the intuitively simple objective of maintaining user registries and resource access policy in centralized repositories, which was easy when RACF was securing a single System/390 MVS operating system image. Likewise, the identification and authentication of users in highly distributed e-business scenarios is driving the implementation of modern distributed identification and authentication techniques, like Kerberos and X.509 version 3 digital certificates, as practical replacements for the long-serving user ID/password logical model. These technologies are

here and available. But what can computer installation managers and users expect in the future? What are some of the driving forces and technologies that are likely to be the focus of tomorrow's security solutions?

A key driving force will clearly be the ever-increasing number of e-business transactions initiated by enterprise Internet customers and business-to-business relationships. The increasing number of transaction requests per unit of time will drive demand for ever-increasing cryptographic system performance and software scalability.

This paper has focused on z/OS and its predecessor OS/390, and with them the Security Server and RACF. Our objective was to make our readers aware of the security strengths of the centralized computing model. We recognize that the real world increasingly consists of a distributed computing model that in the future will be based largely on WebSphere and Java implementations. Tivoli Policy Director is emerging as a popular product for the distributed computing model, providing support for various security disciplines that we have discussed here. Our customers are already asking for integration of Policy Director administration and services with existing z/OS Security Server function. Clearly, requirements in this area will demand the attention of IBM.

Another driving force will be the need for advanced administration capabilities, as the raw number of e-business customers reaches many millions. Banks and insurance companies and similar large e-business enterprises will demand highly scalable implementations of public key infrastructure function in order to manage very large numbers of deployed digital certificates.

Another festering problem is the proliferation of user and security registries. Consider just the IBM operating system platforms of z/OS, AIX, and OS/400. Each platform has its own individual registry with its own format, syntax, and administrative user interface. Other platforms are in wide use, such as Microsoft Windows NT and Windows 2000 with its proprietary and functionally unique directory server. In addition to the operating system platforms, there are middleware and application-unique user and security registries. Policy Director and LDAP are examples of middleware, and SAP AG and Baan provide applications. Customers we speak with agree that this situation is chaotic and, unless somehow addressed, presents a very important and significant inhibitor to sustained industry e-business growth. But how can this problem be addressed? We see by looking at the recent past that when a new registry mechanism is introduced, the existing registries do not disappear, even if the new registry is demonstrably better than the existing ones. In fact, what happens is that enterprises simply have yet another registry to work with. The new one brings with it additional costs of personnel and education. Clearly, new thinking is required, and we expect increased focus on this problem in the days ahead.

Conclusion

In this paper we have described how security for enterprise computing function and information is implemented on IBM OS/390 and @server zSeries computing systems. We have shown that security for these systems is a *comprehensive* integrated set of support elements that address *multiple security disciplines* consisting of user identification and authentication, access control, auditing, distributed directory, networking security, and security administration. We have shown how z/OS, formally the OS/390 operating system, supports these disciplines by starting with hardware—System/390 and now zSeries—that provides integrity, process isolation, and cryptographic capability. On top of this solid hardware foundation, z/OS and OS/390 implement the various elements of security, with the Security Server and Communication Server components, in a flexible way that supports customization and allows various functional options. Further, we have shown how the z/OS Security Server has evolved and how it has been modernized in support of e-business through the adoption of advanced technologies, such as Kerberos and X.509 version 3 digital certificates for user identification and authentication and LDAP as the storage foundation for distributed user registries and resource access control policy. Finally, we have looked to the future and identified problems that will demand operating system and middleware design and development attention in the near future.

This is a large and complex topic. In order to survey the topic we needed to keep the discussion at a high level and to organize it into individual sections. Complete explanations and details on any of these topics can be found by referring to the reference section and to specific product documentation. We would like to call the attention of our readers to specific Internet Web sites.[21] Evolution of the industry and security products will continue, but we expect that the Web sites will continue to keep pace.

*Trademark or registered trademark of International Business Machines Corporation.

**Trademark or registered trademark of Computer Associates International, Inc., Microsoft Corporation, The Open Group, Sun Microsystems, Inc., or Object Management Group.

Cited references and notes

1. Kerberos was developed by the Massachusetts Institute of Technology (MIT).
2. "BSAFE," or more formally RSA BSAFE Crypto-C, is one of a family of security toolkits developed and marketed by RSA Security Inc. The toolkit provides a wide selection of cryptographic software engines and algorithms and is commonly used within IBM security products.
3. A revoked user ID may be reactivated easily through administrative action after appropriate investigation.
4. Remember, with digital certificates, the trusted third party (the certificate authority) needs to be actively involved only during the issuing of the X.509 version 3 digital certificates, which is usually arranged to occur rarely.
5. The Generic Security Services Application Programming Interface (GSS-API) offers a standard interface for application programmers to access security services that are supported by lower-level functions, such as the operating system.
6. CICS is an application server that provides industrial-strength, on-line transaction management for mission-critical applications.
7. The IMS family of products includes the IMS Hierarchical Database Manager, the IMS Transaction Manager, and a growing set of tools for application development, business intelligence, systems and data management, and the deployment of e-business applications.
8. RACF profiles are arranged in data areas known as profile "segments." The base segment, which is always present, contains basic information such as the user ID and password. Optional segments can be added, via RACF administrative support, to contain additional information. Two examples of additional user profile segments are the CICS segment and the OS/390 UNIX system services segment.
9. SNA is Systems Network Architecture. The Communications Server SNA support is provided by the VTAM component (Virtual Telecommunications Access Method).
10. Subarea SNA functions in a hierarchical manner. Each subarea node provides services for and control over peripheral nodes. In a subarea network, VTAM serves as a type 5 node, which is the highest-level node in the subarea hierarchy. Peripheral nodes require the services of a VTAM subarea node to communicate with other peripheral nodes and subareas nodes in the subarea network.
11. APPN functions in a peer-to-peer manner. A network node provides network services for its own end users and end nodes. A network node can be implemented on multiple platforms and does not require VTAM involvement in setting up communications between peers.
12. A logical unit (LU) represents an end user to the SNA and APPN network. End user sessions are called LU-LU sessions. LU 6.2 is an LU type that is used for application-to-application communications. LU 6.2 uses the Advanced Peer-to-Peer Communications protocol (APPC).
13. A control point (CP) is a component of an APPN node. It is responsible for managing the node and its resources.
14. Triple DES is a symmetric cryptographic algorithm. It provides stronger encryption than DES by applying the DES algorithm three times, using either two or three 56-bit cryptographic keys.
15. IPSec is defined by the IETF by Request for Comments (RFC) 2401–2406, 2409, 2410.
16. SNMP v3 is a secure network management protocol. It provides data origin authentication, integrity, and privacy for SNMP messages, as well as access control to SNMP resources. It is defined by RFCs 2271–2275.
17. OSPF (open shortest path first network routing protocol) MD5 authentication is defined by RFC 2328.
18. DNS (Domain Name System) provides numerous TCP/IP directory services, including a mapping of host names to IP addresses. Using cryptographic authentication, Secure DNS ensures that DNS replies are not spoofed, and that they are from a trusted system. Secure DNS is defined by RFC 2535.
19. The syslog daemon provides a system logging facility available to applications. Syslog records can be logged to a variety of destinations, such as files and devices.
20. See http://www.itsec.gov.uk/info.
21. The following are some Web sites relevant to this paper: http://www.ibm.com/servers/eserver/zseries, http://www.ibm.com/servers/eserver/zseries/zos, http://www.ibm.com/servers/eserver/zseries/zos/racf, and http://www.ibm.com/servers/eserver/zseries/zos/security.

Accepted for publication May 15, 2001.

Richard Guski *IBM Server Group, 2455 South Road, Poughkeepsie, New York 12601 (electronic mail: guski@us.ibm.com).* Mr. Guski is an IBM Senior Technical Staff Member focusing on zSeries security design and architecture. He began his career in 1971 with the Mountain Bell Telephone Company (now QWEST) in Denver, Colorado. He joined IBM in 1981 as a programmer in RACF development, where he participated in the design and development of many versions of RACF. He holds a patent on security-related software technology and he has other patents pending. He is a CISSP (certified information systems security professional), certified by the International Information Systems Security Consortium, Inc. Mr. Guski received a B.S. degree in computer technology from the New York Institute of Technology in 1971.

John C. Dayka *IBM Server Group, 2455 South Road, Poughkeepsie, New York 12601 (electronic mail: dayka@us.ibm.com).* Mr. Dayka is an IBM Senior Technical Staff Member. He has worked on the S/390 platform since joining IBM in 1985. He is currently a design team leader and system architect with primary focus on security for both zSeries and z/OS. Mr. Dayka is a graduate of Mount Saint Mary College in Newburgh, New York. He holds a bachelor's degree in computer science.

Linda N. Distel *IBM Server Group, 2455 South Road, Poughkeepsie, New York 12601 (electronic mail: ldistel@us.ibm.com).* Ms. Distel is the program director for cross brand technologies within the Enterprise Server Group. This position includes security strategy and investment planning for z/OS and zSeries platform security. She joined IBM as a programmer in 1983 and has held a number of technical and management positions in S/390 hardware and software development. She is a graduate of the State University of New York at Albany, with a bachelor's degree in computer science and mathematics. Ms. Distel holds a master's degree in computer science from Marist College.

Walter B. Farrell *IBM Server Group, 2455 South Road, Poughkeepsie, New York 12601 (electronic mail: wfarrell@us.ibm.com).*

Mr. Farrell has worked for IBM since 1984, first in design and development for the RACF security product for MVS/ESA and VM/ESA, and subsequently for its follow-on, the Security Server (SecureWay Security Server) for OS/390 and z/OS. He has recently received CISSP certification. Prior to joining IBM, he spent several years as a lead systems programmer for a large IBM banking customer that used both MVS and RACF, and he worked closely with the data security department helping to implement the necessary security controls using RACF. Mr. Farrell holds a master's degree in computer science from Rensselaer Polytechnic Institute in Troy, New York. His bachelor's degree is in mathematics, from the California Institute of Technology in Pasadena, California.

Karen A. Gdaniec *IBM Server Group, 1701 North Street, Endicott, New York 13760 (electronic mail: kgdaniec@us.ibm.com).* Ms. Gdaniec works for IBM within the DCE and Ecommerce Development group and has been the team leader for OS/390 directory development since October 1996. Her team is responsible for DCE's cell directory service as well as the LDAP products. She has worked for IBM since 1983 in a number of positions, including application development, management, marketing support, and product development. Ms. Gdaniec holds a master's degree in computer science from Marist College in Poughkeepsie, New York. Her bachelor's degree is also in computer science, from Indiana University of Pennsylvania.

Michael J. Kelly *IBM Server Group, 2455 South Road, Poughkeepsie, New York 12601 (electronic mail: kellymj@us.ibm.com).* Mr. Kelly works for IBM on S/390 cryptographic software design, development, and service. He joined IBM in 1980 and has held assignments in JES2/MVS system test and in product development for cryptographic products. He is currently a senior software engineer in the Integrated Cryptographic Support Facility (ICSF) development organization, where he has been a lead designer since the product was started in 1987. Prior to joining IBM, Mr. Kelly worked as a cryptologic mathematician for the U.S. Department of Defense. He holds a master's degree in mathematics from Syracuse University.

Mark A. Nelson *IBM Server Group, 2455 South Road, Poughkeepsie, New York 12601 (electronic mail: markan@us.ibm.com).* Mr. Nelson works in RACF development focusing on design and special projects. He has recently received CISSP certification. His career with IBM began in 1982 and has included assignments in RACF management and marketing support. He was design owner for the RACF component of the OS/390 release 8 Security Server. He was designer and developer for the RACF Database Unload Utility, the RACF SMF Unload Utility, and designer for the RACF Remove ID Utility. His other design assignments have included work on RACF's support for IBM's Component Broker product. He is an active speaker and writer on RACF, presenting talks to IBM employees and customers on four continents and publishing numerous articles in *Technical Support* magazine. Mr. Nelson received his bachelor's degree in computer science and electrical engineering from the Polytechnic Institute of New York and his master's degree in information systems from Marist College.

Linwood H. Overby *IBM Application Integration Middleware Division, 4205 South Miami Boulevard, Research Triangle Park, North Carolina 27709 (electronic mail: overbylh@us.ibm.com).* Mr. Overby is a senior software engineer in the z/OS Communications Server strategy and design area, focusing on SNA and TCP/IP security. He joined IBM in 1978 and worked on several IBM networking products before his current assignment. He has been the lead designer for the Communications Server on numerous security functions, including IPSec, TN3270 SSL, and Intrusion Detection Services. He holds five U.S. patents and currently has 17 additional patent applications pending. Mr. Overby received a B.A. in economics and business administration from North Carolina State University in 1977.

Linwood G. Robinson *IBM Server Group, 999 Waterside Drive, Norfolk, Virginia 23510 (electronic mail: linwood@us.ibm.com).* Mr. Robinson's current assignment is with OS/930 Security Marketing Programs, where he is the marketing manager for Enterprise Systems security, cryptography and directory. His responsibilities include preparing marketing information for customers and IBM sales teams, meeting with customers to understand their requirements, and helping validate future development directions. Prior to this assignment, he worked in IBM sales covering large systems hardware, software, storage, systems management, and security. Mr. Robinson has worked for IBM since 1967 in a variety of positions, including operating systems development, management, sales, and marketing support. He holds a bachelor's degree in electrical engineering from Massachusetts Institute of Technology in Cambridge, Massachusetts.