Vulnerability Use Cases

The CLASP Vulnerability Use Cases depict conditions under which security services are vulnerable to attack at the application layer. The Use Cases provide CLASP users with easy-to-understand, specific examples of the relationship between security-unaware design and source coding and possible resulting vulnerabilities in basic security services.

CLASP defines a security vulnerability as a flaw in a software environment — especially in an application — that allows an attacker to assume privileges within the user's system, utilize and regulate its operation, compromise the data it contains, and/or assume trust not granted to the attacker.

The CLASP Vulnerability Use Cases are based on the following common component architectures:

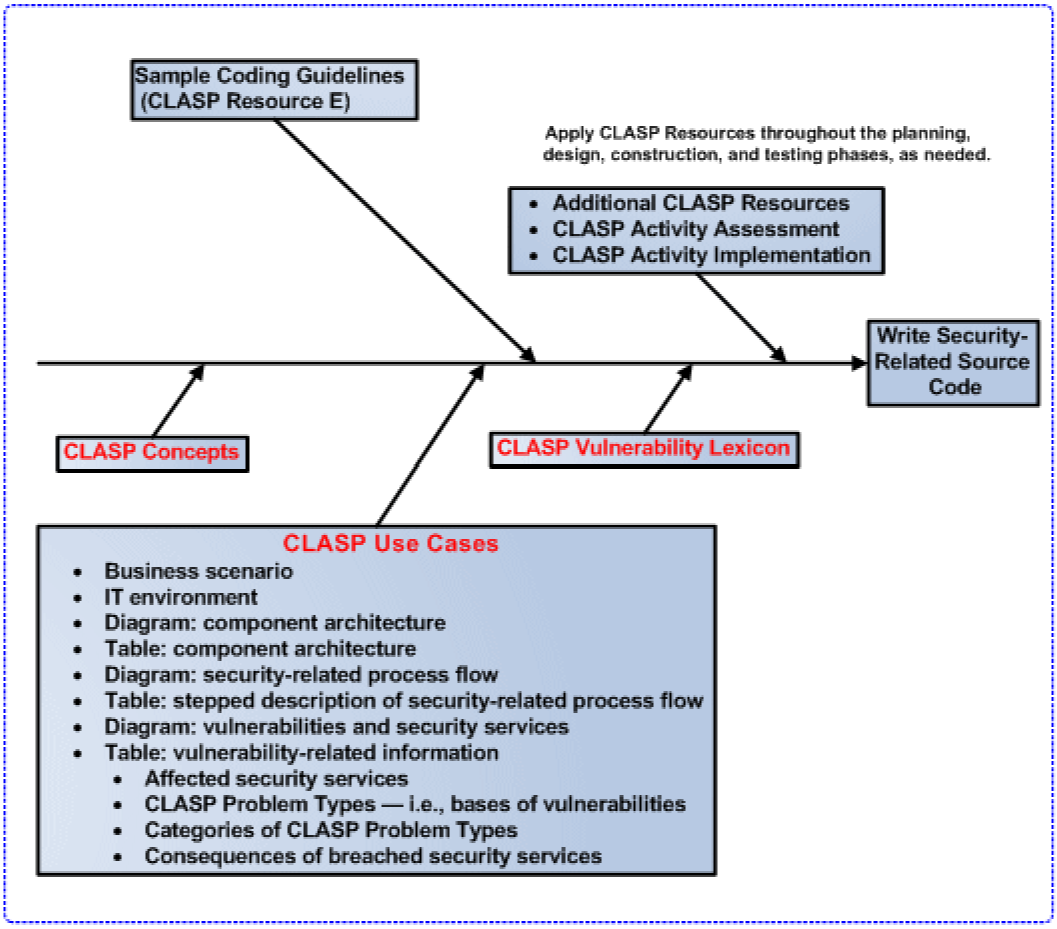
- Monolithic UNIX
- Monolithic mainframe
- Distributed architecture (HTTP[S] & TCP/IP)

Position of Use Cases within CLASP

This page describes a recommended sequence within which to apply the CLASP Vulnerability Use Cases during a security-related software development project. It is recommended to apply the CLASP Use Cases as a bridge from the Concepts View of CLASP to the Vulnerability Lexicon (in the Vulnerability View), since the Use Cases:

- Exemplify CLASP concepts in security-related contexts;
- Provide an overview of the CLASP "Problem Types" within the Vulnerability Lexicon;
- Specify the basic security services which vulnerabilities can cause to fail;
- Show the specific points within a security-related process-flow where vulnerabilities can occur within an application.

The following diagram depicts a recommended position of the Use Cases within the CLASP process:


Sample Coding Guidelines (CLASP Resource E)
Apply CLASP Resources throughout the planning, design, construction, and testing phases, as needed.
Additional CLASP Resources
CLASP Activity Assessment
CLASP Activity Implementation
Write Security-Related Source Code
CLASP Concepts
CLASP Vulnerability Lexicon
CLASP Use Cases
Business scenario
IT environment
Diagram: component architecture
Table: component architecture
Diagram: security-related process flow
Table: stepped description of security-related process flow
Diagram: vulnerabilities and security services
Table: vulnerability-related information
Affected security services
CLASP Problem Types — i.e., bases of vulnerabilities
Categories of CLASP Problem Types
Consequences of breached security services

CLASP Vulnerability Lexicon

A security vulnerability occurs in a software application when any part of the application allows a breach of the security policy governing it.

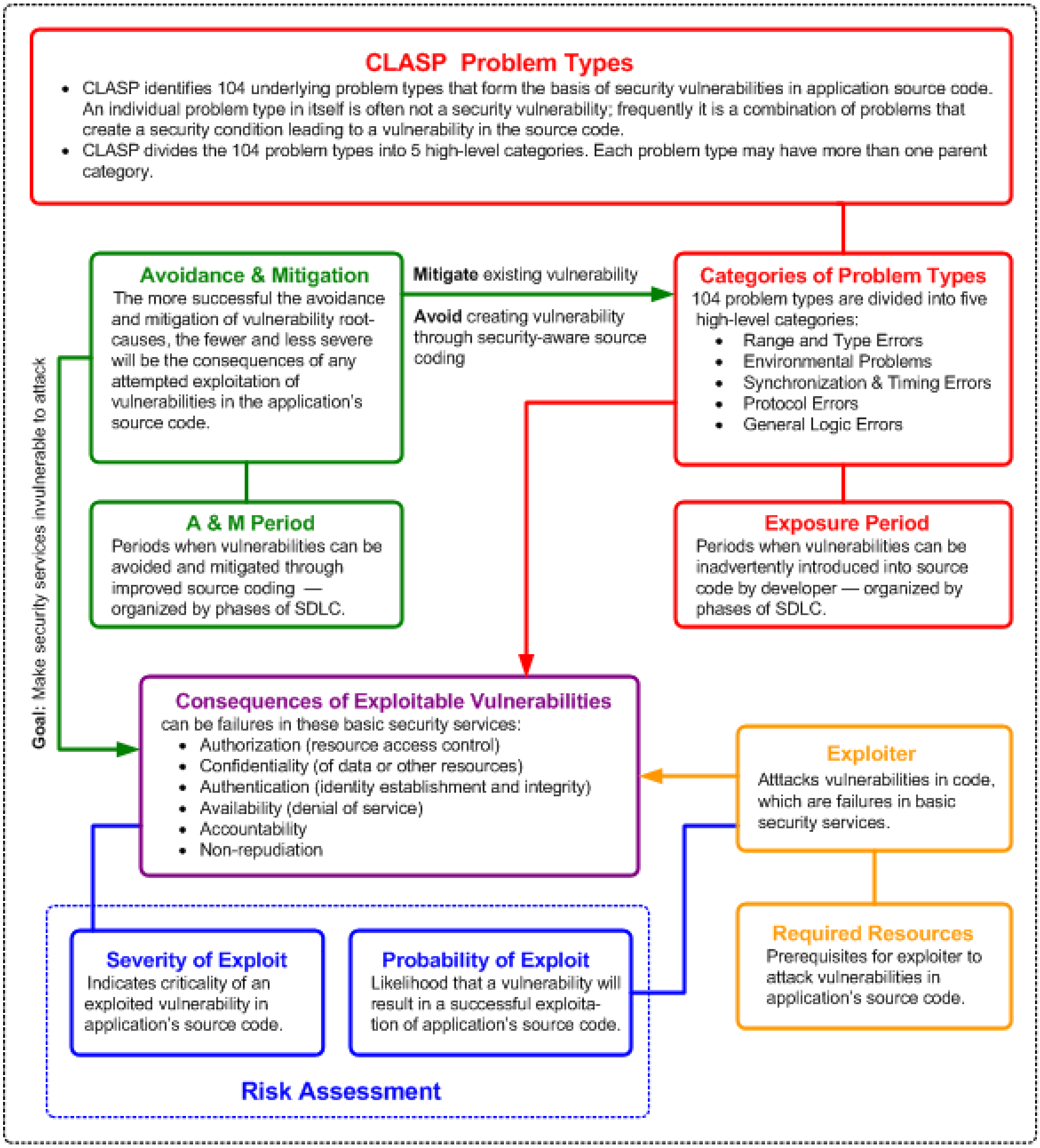
CLASP identifies 104 underlying **problem types** — i.e., bases of security vulnerabilities — that allow security vulnerabilities to occur in application source code. An individual problem type in itself is often not a security vulnerability; frequently it is a combination of problems that create a security condition leading to a vulnerability in the source code.

CLASP divides the 104 problem types into 5 high-level categories. Each problem type may have more than one parent category. The CLASP categories are:

- Range and type errors
- Environmental problems
- Synchronization & timing errors
- Protocol errors
- General logic errors

The following diagram is a taxonomy of CLASP. The taxonomy is a high-level classification of the CLASP process, divided into classes for better evaluation and resolution of security vulnerabilities in source code. For the CLASP Use Cases, this diagram depicts:

- The position of the 104 problem types within CLASP;
- The division of the problem types into five high-level categories;
- The consequences of exploitable security vulnerabilities for basic security services.



CLASP Problem Types

- CLASP identifies 104 underlying problem types that form the basis of security vulnerabilities in application source code. An individual problem type in itself is often not a security vulnerability; frequently it is a combination of problems that create a security condition leading to a vulnerability in the source code.
- CLASP divides the 104 problem types into 5 high-level categories. Each problem type may have more than one parent category.

Avoidance & Mitigation

The more successful the avoidance and mitigation of vulnerability root-causes, the fewer and less severe will be the consequences of any attempted exploitation of vulnerabilities in the application's source code.

Mitigate existing vulnerability

Avoid creating vulnerability through security-aware source coding

Categories of Problem Types

104 problem types are divided into five high-level categories:

- Range and Type Errors
- Environmental Problems
- Synchronization & Timing Errors
- Protocol Errors
- General Logic Errors

A & M Period

Periods when vulnerabilities can be avoided and mitigated through improved source coding — organized by phases of SDLC.

Exposure Period

Periods when vulnerabilities can be inadvertently introduced into source code by developer — organized by phases of SDLC.

Goal: Make security services invulnerable to attack

Consequences of Exploitable Vulnerabilities

can be failures in these basic security services:

- Authorization (resource access control)
- Confidentiality (of data or other resources)
- Authentication (identity establishment and integrity)
- Availability (denial of service)
- Accountability
- Non-repudiation

Exploiter

Attacks vulnerabilities in code, which are failures in basic security services.

Severity of Exploit

Indicates criticality of an exploited vulnerability in application's source code.

Probability of Exploit

Likelihood that a vulnerability will result in a successful exploitation of application's source code.

Required Resources

Prerequisites for exploiter to attack vulnerabilities in application's source code.

Risk Assessment

Overview of Use Cases

The following CLASP Vulnerability Use Cases are described:

- Monolithic UNIX
- Monolithic mainframe
- Distributed architecture (HTTP[S] & TCP/IP)

Each CLASP Use Case is organized into the following sections:

Section	Description
Business scenario	Provides an overview of the business context of the CLASP Use Case.
IT environment	Describes the operating system(s) and programming languages used in the IT environment and other useful information.
Diagram of component architecture	Provides an overview of the major components in the architectural environment.
Table of component architecture	This table describes the diagram of component architecture.
Diagram of security-related process flow	Provides an overview of the security-related process flow of each Use Case.
Table of security-related process flow	This table accompanies the diagram of security-related process flow and provides a stepped description of the process flow.
Diagram of vulnerabilities and security services	The diagram of vulnerabilities depicts the points in the security-related process flow where vulnerabilities can occur.

Section	Description
Table of vulnerability-related information	This table accompanies the diagram of vulnerabilities and describes this related information: • Specific security services affected by the vulnerabilities. • The specific CLASP problem types which can lead to vulnerabilities. • The category of each of the problem types — i.e.: • Range and type errors; • Environmental problems; • Synchronization and timing errors; • Protocol errors; • General logic errors. • The possible consequences of unresolved security vulnerabilities.

Notes on Operating Systems

The CLASP process is application-centric, and the CLASP Use Cases clearly document this orientation. In order to emphasize the central role of applications in the Use Cases, the respective operating system is not described in detail in the diagrams and tables describing the security-related process flows and component architecture. Neither the UNIX, nor mainframe, nor distributed architecture Use Case indicates what the application user is logged on to, other than the machine itself.

Vulnerability Types by Operating System

In general, UNIX and Windows applications are more prone than mainframe applications to vulnerabilities based on security-unaware programming. Vulnerabilities in mainframe applications are more likely to be the consequence of improper administrative care and incorrect system configuration.

It is quite possible for a programmer to introduce vulnerabilities, for example, into CICS — the application environment used in the monolithic mainframe CLASP Use Case — that relate to failure to validate input. However, customers using modern-day CICS have many safeguards (e.g., storage isolation) to prevent transactions from accessing things in storage for which they have no authorization. In addition, COBOL does not provide the programmer with such vulnerable devices as memory pointers, which are available in C, Java, PL/1, and assembler languages.

Operating Systems & Security Services

UNIX and mainframe systems differ in how they perform key security services: Examples:

- In the UNIX Use Case, the user logs onto an interactive session with the operating system, and the security checks for authentication (user ID and password) and authorization (application and data-related) are all performed by the operating system. More correctly, a “shell” — i.e., user interface — within the operating system performs the security checks.
- In the mainframe Use Case, the user logs on to CICS — which is the most common application environment — and not the operating system. Therefore CICS (or rather the proper configuration/administration of CICS) ensures that the TP monitor performs the required authentication. Similarly, it is CICS (i.e., RACF in the form of a call-in) which performs the authorization check before permitting the user to execute the transaction — again requiring proper configuration/administration of the application environment. In contrast, authorization checks for accessing the data (located in the VSAM file cluster) are performed at the operating system-level before allowing any programs to read or update the data.

Case 1: Monolithic UNIX

Business Scenario

A leading investment corporation has applications which process specialized orders for select personal customers. Application users are customer service representatives who either update or create custom account information. The incoming orders are in the form of telephone communication. All incoming orders are processed by the applications on a single UNIX machine.

IT Environment

The IT organization of the investment corporation develops its own applications in order to gain an advantage in a highly competitive and quickly changing business environment. The applications in question are developed in C/C++ on UNIX in order to reduce time-to-market. The users utilize VT terminals in order to exclude the potential security vulnerabilities in an IP-based network.

Diagram: Component Architecture

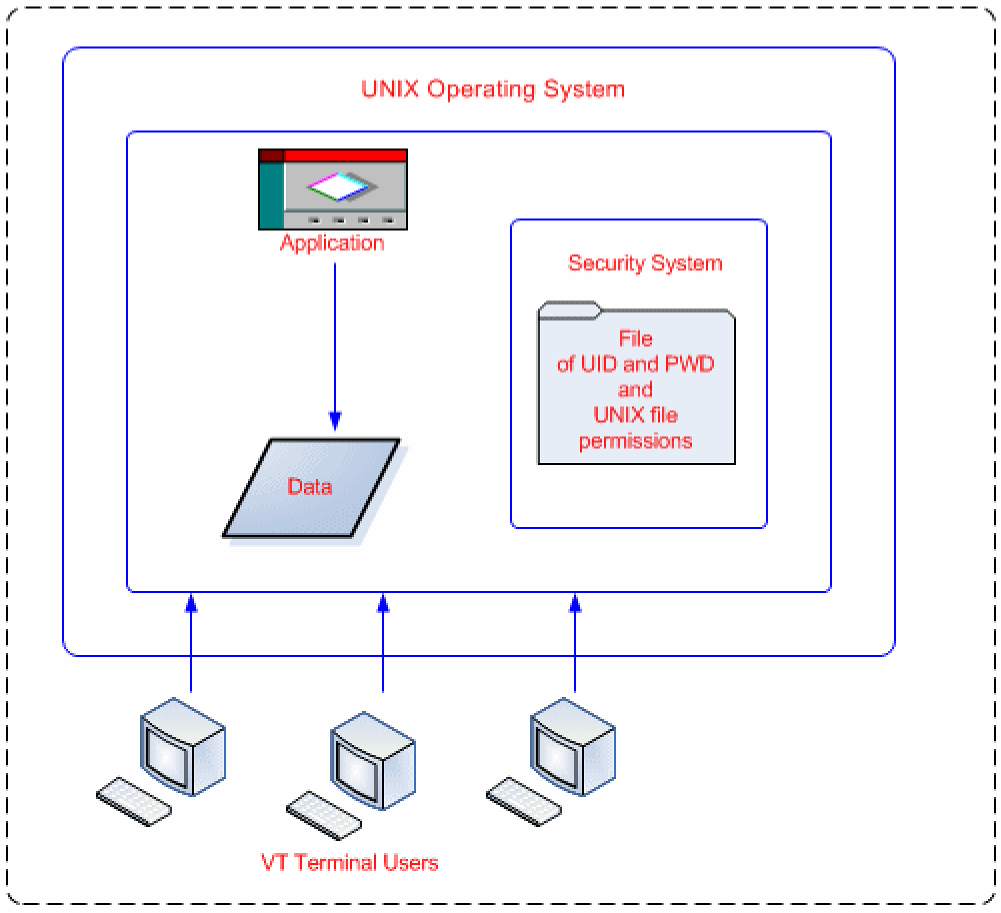


Table: Component Architecture

Component	Description of Component
• User	The application users are customer service representatives of the investment corporation who log on to the single UNIX machine. The users are located within a facility of the organization — i.e., no remote access is required.
• Application	This is a custom application developed within the organization — i.e., it is not a package application.
• Data	The application data in question is UNIX file-based.
• Security System	The elements of the security system are: • Local password file; • UNIX file permissions.

Diagram: Security-Related Process Flow

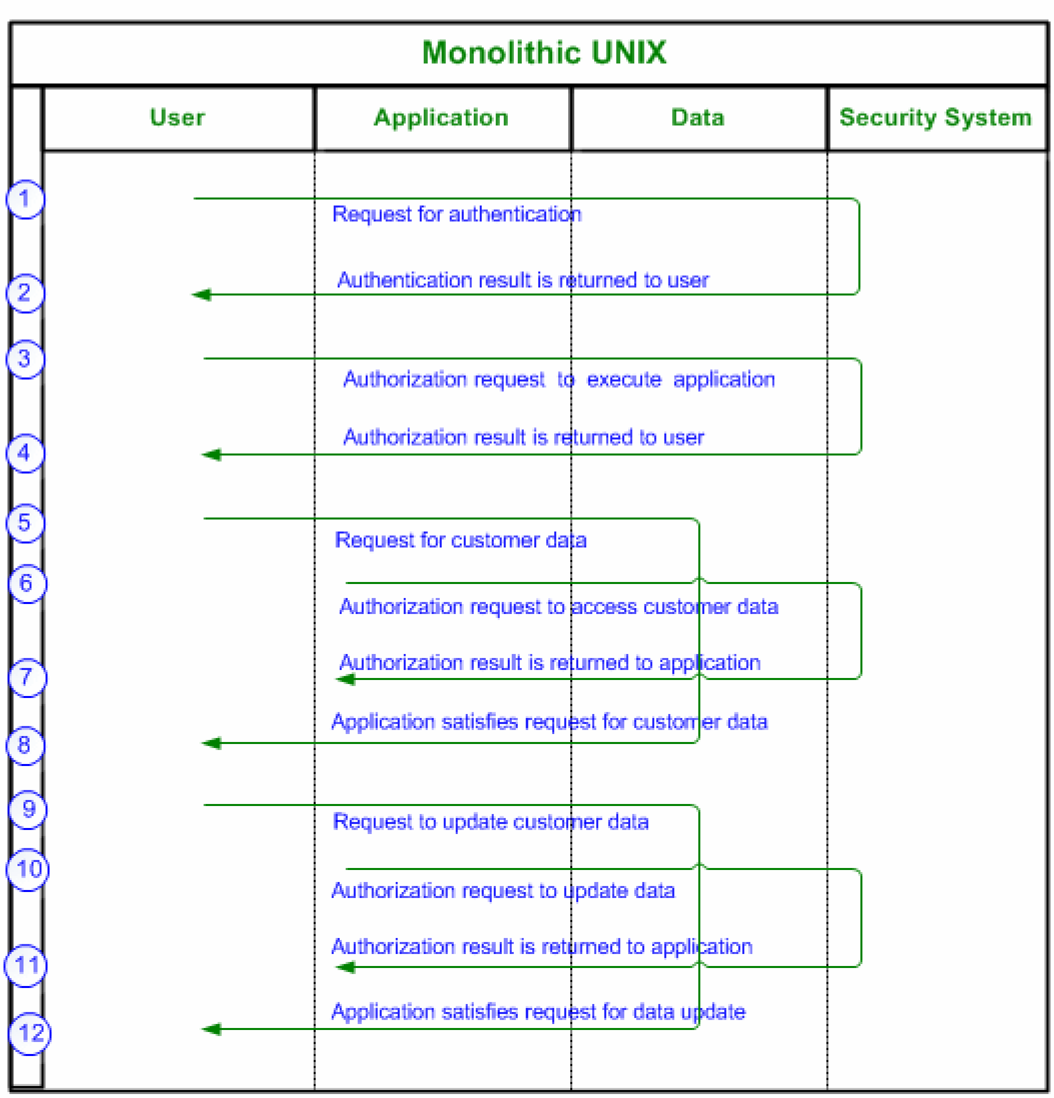

Monolithic UNIX
User
Application
Data
Security System
1
Request for authentication
2
Authentication result is returned to user
3
Authorization request to execute application
4
Authorization result is returned to user
5
Request for customer data
6
Authorization request to access customer data
7
Authorization result is returned to application
8
Application satisfies request for customer data
9
Request to update customer data
10
Authorization request to update data
11
Authorization result is returned to application
12
Application satisfies request for data update

Table: Security-Related Process Flow

The following table provides a stepped description of the security-related process flow depicted in the figure above.

Step	Description of Step in Security-Related Process Flow
1	The user performs an initial logon and is prompted for user ID and password.
2	If the user ID and password are successfully entered, a session with the UNIX machine is initiated.
3	The application user invokes the desired application.
4	To obtain authorization to run the application, the UNIX operating system checks file permissions for the invoking user.
5	The application user (i.e., customer service representative) requests account information for an existing customer.
6	The application reads account record(s) for the specified customer. In the process of reading the data, the operating system checks UNIX file permissions for reading the data.
7	If the UNIX file permissions allow it, the record is returned to the application.
8	The customer data is returned to the application user and is displayed on the VT terminal.
9	The application user enters required account updates (as instructed by the customer, specifically buying or selling investments).
10	The application updates account record(s) for the specified customer. In the process of updating the data, the operating system checks UNIX file permissions for reading the data.
11	If the UNIX file permissions allow it, the record is returned to the application.
12	The application satisfies the request for updating the account data for the specified customer and displays confirmation of the update on the VT terminal.

Diagram: Vulnerabilities & Security Services

The following security-related process flow shows a selection of CLASP vulnerabilities that are possible for this process flow.

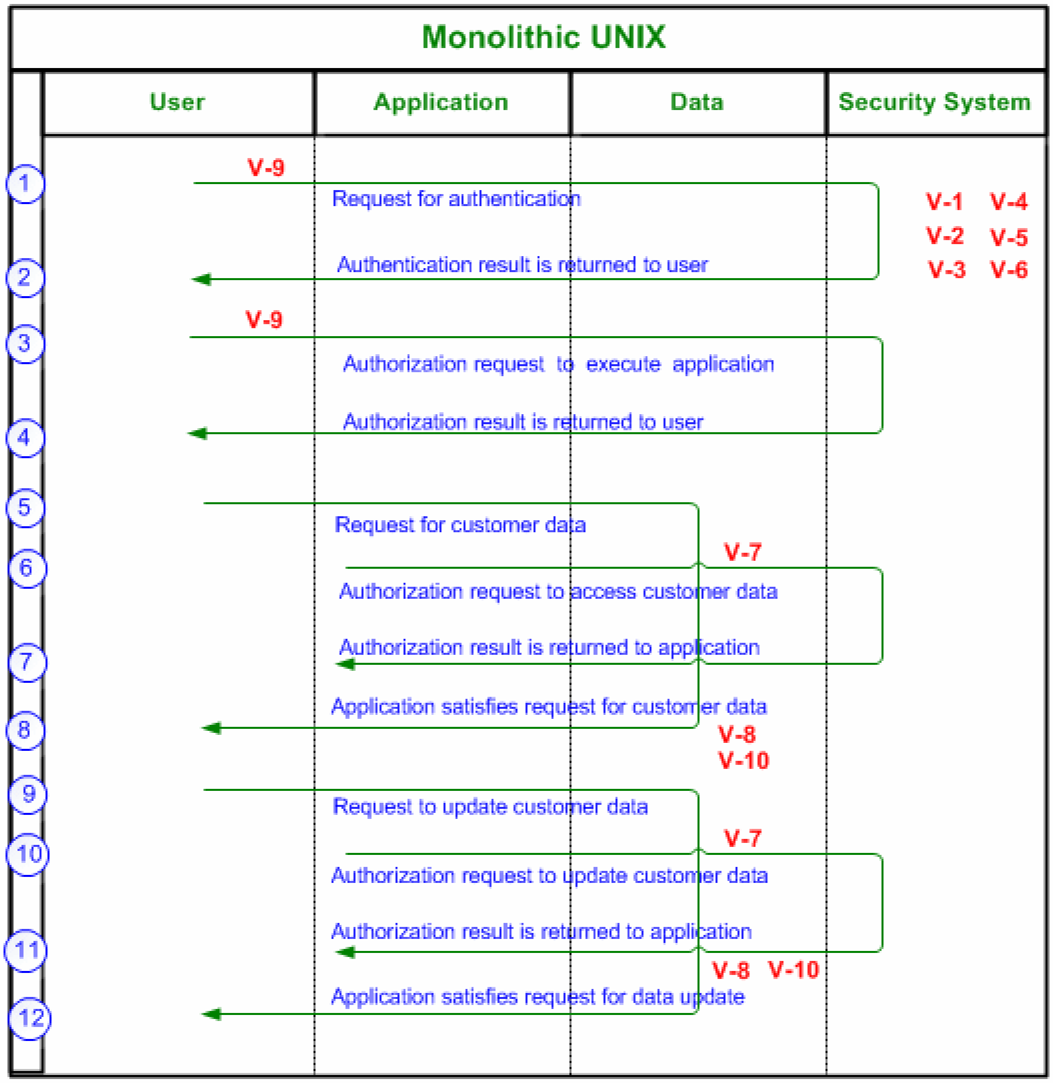

Monolithic UNIX
User
Application
Data
Security System
1
V-9
Request for authentication
V-1 V-4
V-2 V-5
V-3 V-6
2
Authentication result is returned to user
3
V-9
Authorization request to execute application
4
Authorization result is returned to user
5
Request for customer data
6
V-7
Authorization request to access customer data
7
Authorization result is returned to application
Application satisfies request for customer data
8
V-8
V-10
9
Request to update customer data
10
V-7
Authorization request to update customer data
11
Authorization result is returned to application
V-8 V-10
12
Application satisfies request for data update

Table: Vulnerability-Related Information

The following table provides a stepped description of the security-related process flow depicted in the figure above:

Tag	Vulnerabilities & Security Services
V-1	• Affected Security Service(s): Authentication • **CLASP Problem Type:** Use of hard-coded password • **Category of Problem Type:** Protocol errors • **Consequence(s) —** Authentication: If hard-coded passwords are used, it is almost certain that malicious users will gain access through the account in question.
V-2	• Affected Security Service(s): Authentication • **CLASP Problem Type:** Using password systems • **Category of Vulnerability:** Protocol errors • **Consequence(s) —** Authentication: The failure of a password authentication mechanism will almost always result in attackers being authorized as valid users.
V-3	• Affected Security Service(s): Authentication • **CLASP Problem Type:** Allowing password aging • **Category of Problem Type:** Protocol errors • **Consequence(s) —** Authentication: As passwords age, the probability that they are compromised grows.
V-4	• **Affected Security Service(s):** Authentication • **CLASP Problem Type:** Not allowing password aging • **Category of Problem Type:** Protocol errors • **Consequence(s) —** Authentication: As passwords age, the probability that they are compromised grows.

Tag	Vulnerabilities & Security Services
V-5	• **Affected Security Service(s):** Confidentiality; Authentication • **CLASP Problem Type:** Storing passwords in a recoverable format • **Category of Problem Type:** Protocol errors • Consequence(s): • Confidentiality: User's passwords may be revealed. • Authentication: Revealed passwords may be reused elsewhere to impersonate the users in question.
V-6	• **Affected Security Service(s):** Authentication • **CLASP Problem Type:** Using single-factor authentication • **Category of Problem Type:** Protocol errors • **Consequence(s) —** Authentication: If the secret in a single-factor authentication scheme gets compromised, full authentication is possible.
V-7	• **Affected Security Service(s):** Integrity • **Vulnerability:** Failure to protect stored data from modification • **Category of Problem Type:** Protocol errors • **Consequence(s) —** Integrity: The object could be tampered with.
V-8	• Affected Security Service(s): Availability; Access control (instruction processing); Other • **CLASP Problem Type:** Buffer overflow • **Category of Problem Type:** Range and type errors • Consequence(s): • Availability: Buffer overflows generally lead to crashes. Other attacks leading to lack of availability are possible, including putting the program into an infinite loop. • Access control (instruction processing): Buffer overflows often can be used to execute arbitrary code, which is usually outside the scope of a program's implicit security policy. • Other: When the consequence is arbitrary code execution, this can often be used to subvert any other security service.

Tag	Vulnerabilities & Security Services
V-9	• **Affected Security Service(s):** Authorization; Authentication • **CLASP Problem Type:** Failure to check whether privileges were dropped successfully • **Category of Problem Type:** General logic errors • Consequence(s): • Authorization: If privileges are not dropped, neither are access rights of the user. Often these rights can be prevented from being dropped. • Authentication: If privileges are not dropped, in some cases the system may record actions as the user which is being impersonated rather than the impersonator.
V-10	• Affected Security Service(s): Confidentiality; Integrity; Accountability • CLASP Problem Type: Failure to encrypt data • **Category of Problem Type:** Protocol Errors • Consequence(s): • Confidentiality: Properly encrypted data channels ensure data confidentiality. • Integrity: Properly encrypted data channels ensure data integrity. • Accountability: Properly encrypted data channels ensure accountability.

Case 2: Monolithic Mainframe

Business Scenario

A leading insurance company has applications which process claims by its customers. Application users are customer service representatives who either create or update claims information based on telephone conversations with their customers. All incoming claims are processed by the applications on a single IBM mainframe machine.

IT Environment

The IT organization of the insurance company develops its own applications in order to gain an advantage in a highly competitive and quickly changing business environment. The applications in question are developed in COBOL to run under CICS on a z/OS IBM mainframe machine. These custom-written applications enable the insurance company to respond rapidly to the time-critical needs of its client. The users utilize 3270 terminals from where they log on directly to CICS.

Diagram: Component Architecture

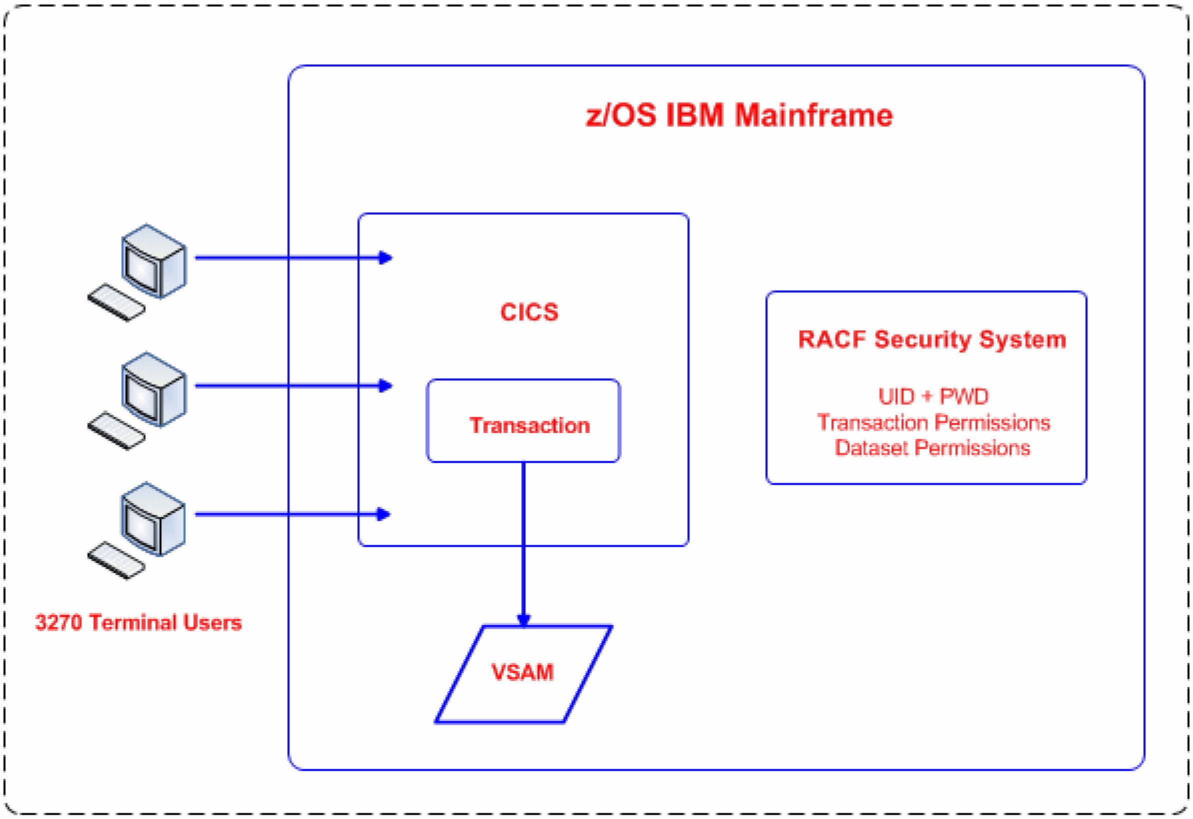


Table: Component Architecture

Component	Description of Component
• User	The application users are customer service representatives of the insurance company who log on to CICS running under z/OS on a single IBM mainframe. The users are located within a facility of the organization — i.e., no remote access is required.
• Application	This is a custom application developed within the organization — i.e., it is not a package application.
• Data	The application data in question is located in VSAM file cluster on the single IBM mainframe machine.
• Security System	The elements of the security system are: • RACF sign-on security; • RACF authorization to execute CICS transactions; • RACF authorization to read/update VSAM file data.

Diagram: Security-Related Process Flow

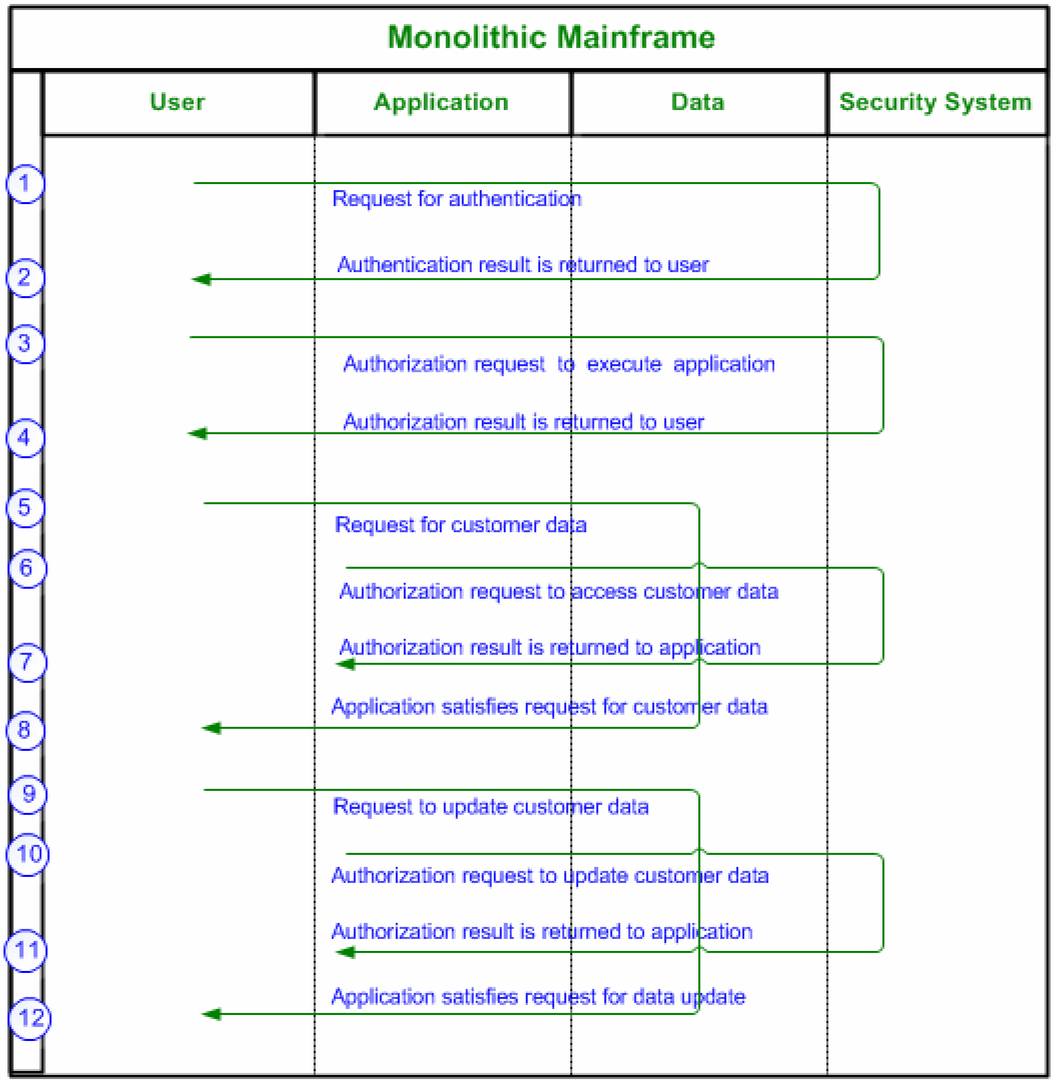

Monolithic Mainframe
User
Application
Data
Security System
1
Request for authentication
2
Authentication result is returned to user
3
Authorization request to execute application
4
Authorization result is returned to user
5
Request for customer data
6
Authorization request to access customer data
7
Authorization result is returned to application
Application satisfies request for customer data
8
9
Request to update customer data
10
Authorization request to update customer data
11
Authorization result is returned to application
Application satisfies request for data update
12

Table: Security-Related Process Flow

The following table provides a stepped description of the security-related process flow depicted in the figure above.

Step	Description of Step in Security-Related Process Flow
1	The user performs an initial logon and is prompted for user ID and password.
2	If the user ID and password are successfully entered, a CICS session with the IBM mainframe is initiated.
3	The application user invokes the desired CICS transaction.
4	To obtain authorization to run the transaction, the CICS determines permission for the invoking user.
5	The application user (i.e., customer service representative) requests account information for an existing customer.
6	The application reads account record(s) for the specified customer. In the process of reading the data, the operating system determines whether the user is permitted to read the relevant VSAM file.
7	If RACF authorizations allow it, the record is returned to the application from VSAM.
8	The customer data is returned to the application user and is displayed on the 3270 terminal.
9	The application user enters required information – either creating a new claim or adding further information (as instructed by the customer, specifically relating to the insurance claim).
10	The application updates account record(s) for the specified customer. In the process of updating the data, the operating system determines whether the user is permitted to update the relevant VSAM file.
11	If RACF authorizations allow it, the record is updated in the VSAM file.
12	The application satisfies the request for updating the customer claim data and displays confirmation of the update on the 3270 terminal.

Diagram: Vulnerabilities & Security Services

The following security-related process flow shows a selection of CLASP vulnerabilities that are possible for this process flow.

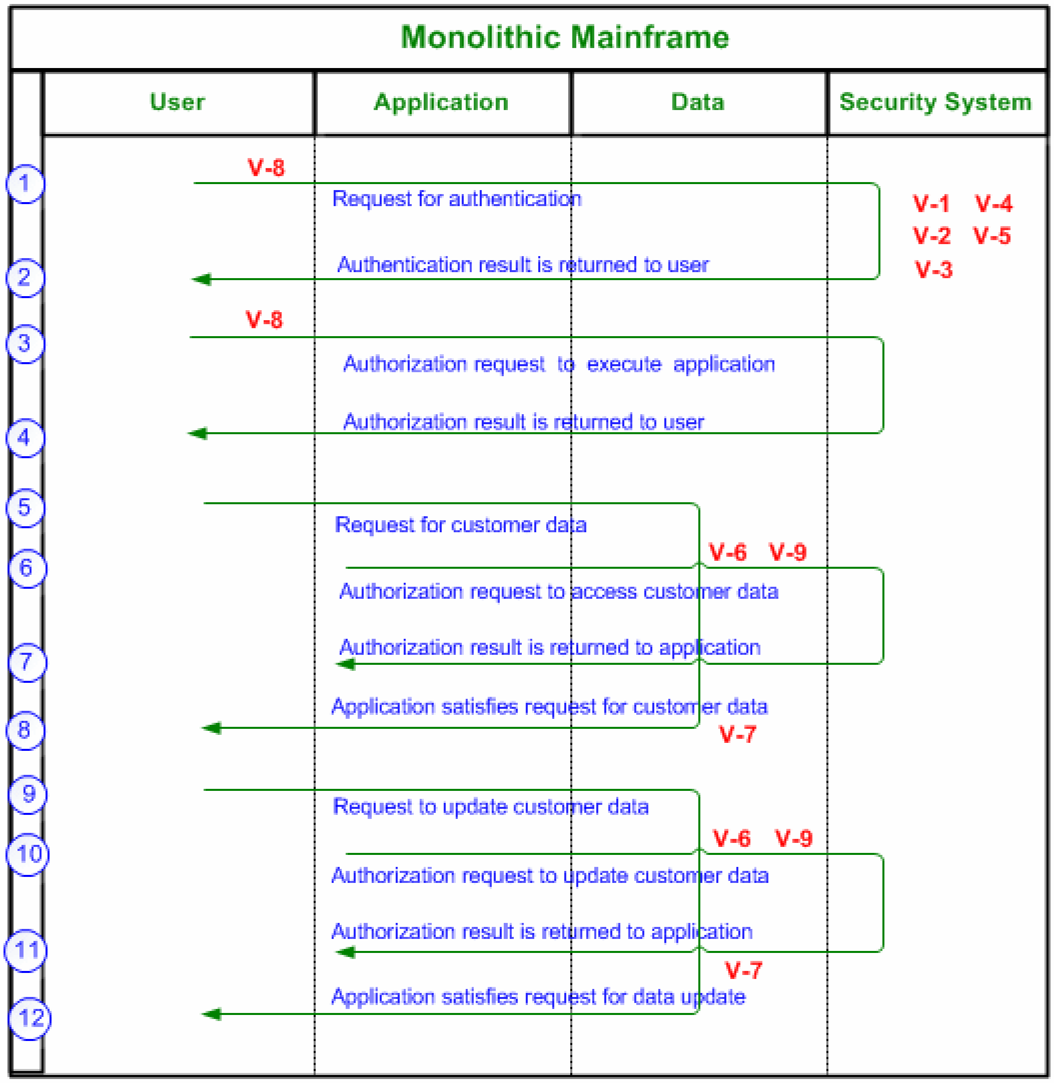

Monolithic Mainframe
User
Application
Data
Security System
1
V-8
Request for authentication
V-1 V-4
V-2 V-5
V-3
2
Authentication result is returned to user
3
V-8
Authorization request to execute application
4
Authorization result is returned to user
5
Request for customer data
6
V-6 V-9
Authorization request to access customer data
7
Authorization result is returned to application
Application satisfies request for customer data
8
V-7
9
Request to update customer data
10
V-6 V-9
Authorization request to update customer data
11
Authorization result is returned to application
V-7
12
Application satisfies request for data update

Table: Vulnerability-Related Information

The following table provides a stepped description of the security-related process flow depicted in the figure above:

Tag	Vulnerabilities & Security Services
V-1	• Affected Security Service(s): Authentication • **CLASP Problem Type:** Using password systems • **Category of Vulnerability:** Protocol errors • **Consequence(s) —** Authentication: The failure of a password authentication mechanism will almost always result in attackers being authorized as valid users. • Note: RACF always fails-safe, which does not allow anyone to log on. However, failure to start CICS with SECURITY=YES would circumvent the RACF authentication check. Therefore, it is necessary to protect the CICS start-up JCL and its parameter files to avoid misuse. In addition, it is also necessary to restrict access to the machine only to CICS users in order to prevent malicious programs being typed-in/compiled by staff. This means that CICS can be considered an “execute” rather than “development” environment for the most part.
V-2	• Affected Security Service(s): Authentication • **CLASP Problem Type:** Allowing password aging • **Category of Problem Type:** Protocol errors • **Consequence(s) —** Authentication: As passwords age, the probability that they are compromised grows. • **Note:** RACF must be correctly configured to obtain correct password aging.
V-3	• **Affected Security Service(s):** Authentication • **CLASP Problem Type:** Not allowing password aging • **Category of Problem Type:** Protocol errors • **Consequence(s) —** Authentication: As passwords age, the probability that they are compromised grows. • **Note:** RACF must be correctly configured to obtain the correct password aging.

Tag	Vulnerabilities & Security Services
V-4	• **Affected Security Service(s):** Confidentiality; Authentication • **CLASP Problem Type:** Storing passwords in a recoverable format • **Category of Problem Type:** Protocol errors • Consequence(s): • Confidentiality: User's passwords may be revealed. • Authentication: Revealed passwords may be reused elsewhere to impersonate the users in question. • **Note:** RACF will not reveal a user's password. However, care must be taken not to store user passwords in order to execute another application component.
V-5	• **Affected Security Service(s):** Authentication • **CLASP Problem Type:** Using single-factor authentication • **Category of Problem Type:** Protocol errors • **Consequence(s) —** Authentication: **I**f the secret in a single-factor authentication scheme gets compromised, full authentication is possible.
V-6	• **Affected Security Service(s):** Integrity • **Vulnerability:** Failure to protect stored data from modification • **Category of Problem Type:** Protocol errors • **Consequence(s) —** Integrity: The object could be tampered with. • **Note:** Tampering could occur if the VSAM file cluster were not protected by RACF, resulting in a failure in data administration.

Tag	Vulnerabilities & Security Services
V-7	• Affected Security Service(s): Availability; Access control (instruction processing); Other • **CLASP Problem Type:** Buffer overflow • **Category of Problem Type:** Range and type errors • Consequence(s): • Availability: Buffer overflows generally lead to crashes. Other attacks leading to lack of availability are possible, including putting the program into an infinite loop. • Access control (instruction processing): Buffer overflows often can be used to execute arbitrary code, which is usually outside the scope of a program's implicit security policy. Other: When the consequence is arbitrary code execution, this can often be used to subvert any other security service. • **Note:** Buffer overflows that maliciously execute code are difficult to cause in CICS / COBOL programming. However, range and type errors can still allow incorrect data validation, leading to incorrect manipulation, display, and updating of data.
V-8	• **Affected Security Service(s):** Authorization; Authentication • **CLASP Problem Type:** Failure to check whether privileges were dropped successfully • **Category of Problem Type:** General logic errors • Consequence(s): • Authorization: If privileges are not dropped, neither are access rights of the user. Often these rights can be prevented from being dropped. • Authentication: If privileges are not dropped, in some cases the system may record actions as the user which is being impersonated rather than the impersonator. • **Note:** In this example, the application is not authorized to drop permissions, etc. However, inadequate data/security administration could leave outdated user IDs and permissions exposed to attack. Refreshing the RACF information after making definition changes is required, especially where the operating system caches such information in protected memory.

Tag	Vulnerabilities & Security Services
V-9	• **Affected Security Service(s):** Authorization • **CLASP Problem Type:** Trust of system event data • **Category of Problem Type:** Environmental Problems • **Consequence(s) —** Authorization: If one trusts the system-event information and executes commands based on it, one could potentially take actions based on a spoofed identity.

Case 3: Distributed Architecture

Business Scenario

The securities department of a major international investment corporation employs a distributed computing system for performing automated operations **—** updated in real-time **—** which support many types of stock market activity and securities, including: shares; bonds; investment certificates; deposit certificates; stock options; state securities; etc.

The main office of the securities department, which performs solely controlling and management functions, has a central database server which collects data from branch offices through TCP/IP in order to prepare consolidated reports. Each branch office has a local database server containing data on its own group of customers and transmits selected data to the central office. Customers communicate with branch offices through HTTP[S] connections.

IT Environment (HTTP[S] & TCP/IP)

- **Customers:** Customers are able to browse their account information through their browsers via HTTP[S].; this executes a sub-set of the available COM objects located in the branch office, which services the customers' accounts, in order to access DBMS-stored data located in the branch offices.
- **Main Office:** The IT organization of the securities company develops its own applications in order to gain competitive advantage in this quickly changing business environment. The applications in the main office are Windows-based, COM-written applications, accessing the DBMS server in order to obtain a centralized view of the company's transactions performed in branch offices. The main office and branch offices communicate via TCP/IP connection.
- **Branch Offices:** The branch offices also have Windows-based, COM-written applications, accessing the local DBMS servers for the purpose of executing transactions which involve the retrieval and writing of data to the local DBMS servers. The branch offices and main office communicate via TCP/IP connection.

Diagram: Component Architecture

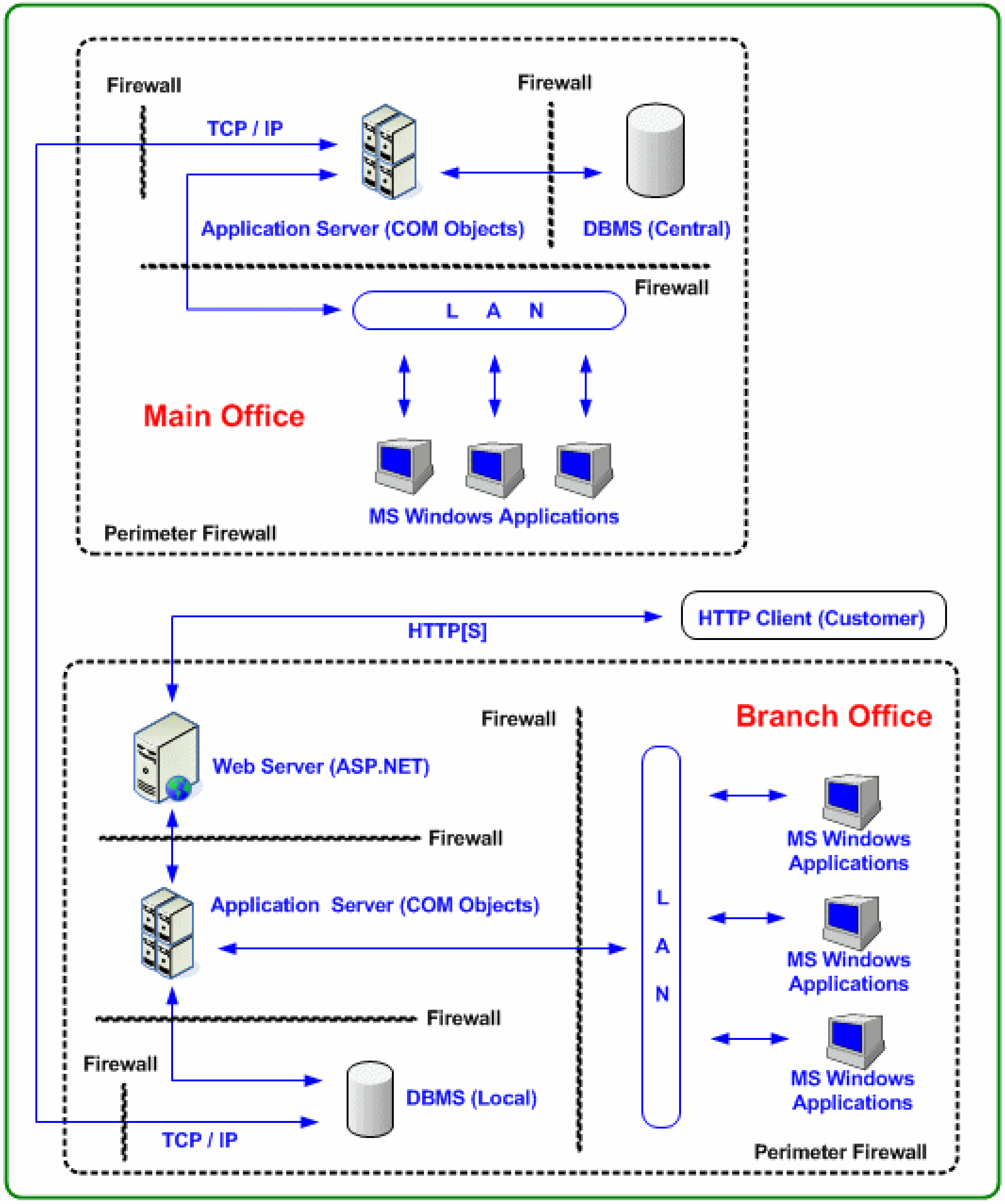


Table: Component Architecture

Component	Description of Component
• User	Three types of users are represented in this Use Case: • Employee, located in the main office who logs onto the Local Area Network in the office and executes applications on the application server of the main office. • Employee, located in the branch office who logs onto the Local Area Network in the office and executes applications on the application server of the branch office. • Customer, who accesses his locally held account data by web browser through HTTP[S].
• Application	This is a custom application with the following components: • Windows client application(s) within the main office and the branch office. These invoke Windows COM objects in order to perform business logic and read/update customer data. • Windows COM objects located either in the main office or the branch offices. • ASP.NET server pages which execute secure HTTP transactions via the browser of the customer and, in turn, invoke a subset of COM objects located in the application server of the branch office.
• Data	The application data in question is located in: • The main office DBMS server. • The branch office DBMS server. **Note:** The main office DBMS server represents a consolidated view of all the branch offices.

Component	Description of Component
• Security System	**The elements of the security system are:** • Windows Domain controller for the main office and each of the branch offices; • Various fire walls located in each office (main and branch); • Digital external certificates which are distributed to each customer in order to gain access to the web server in the branch office and are validated locally through a private signing key to sign the message digitally. The associated public signing key in the certificate is then used to verify this digital signature. • Configuration of the respective DBMS servers in order to comply with the locally held Domain server permission.

Customer's Participation

This section describes the customer's participation in the security-related process flow. The customer's participation consists of three high-level, security-related activities: authentication; authorization to execute; and authorization to access data.

Diagram: Component Architecture

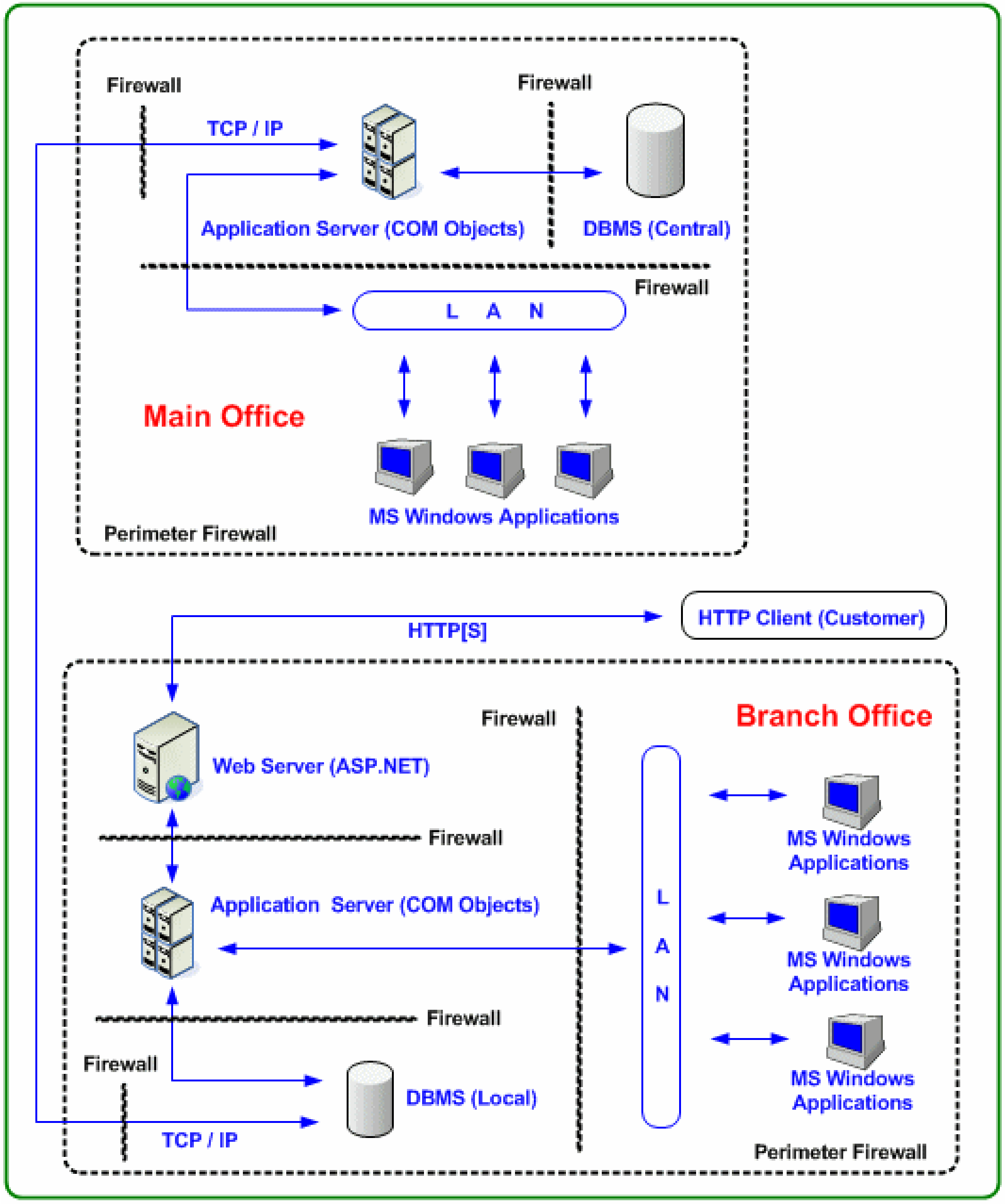
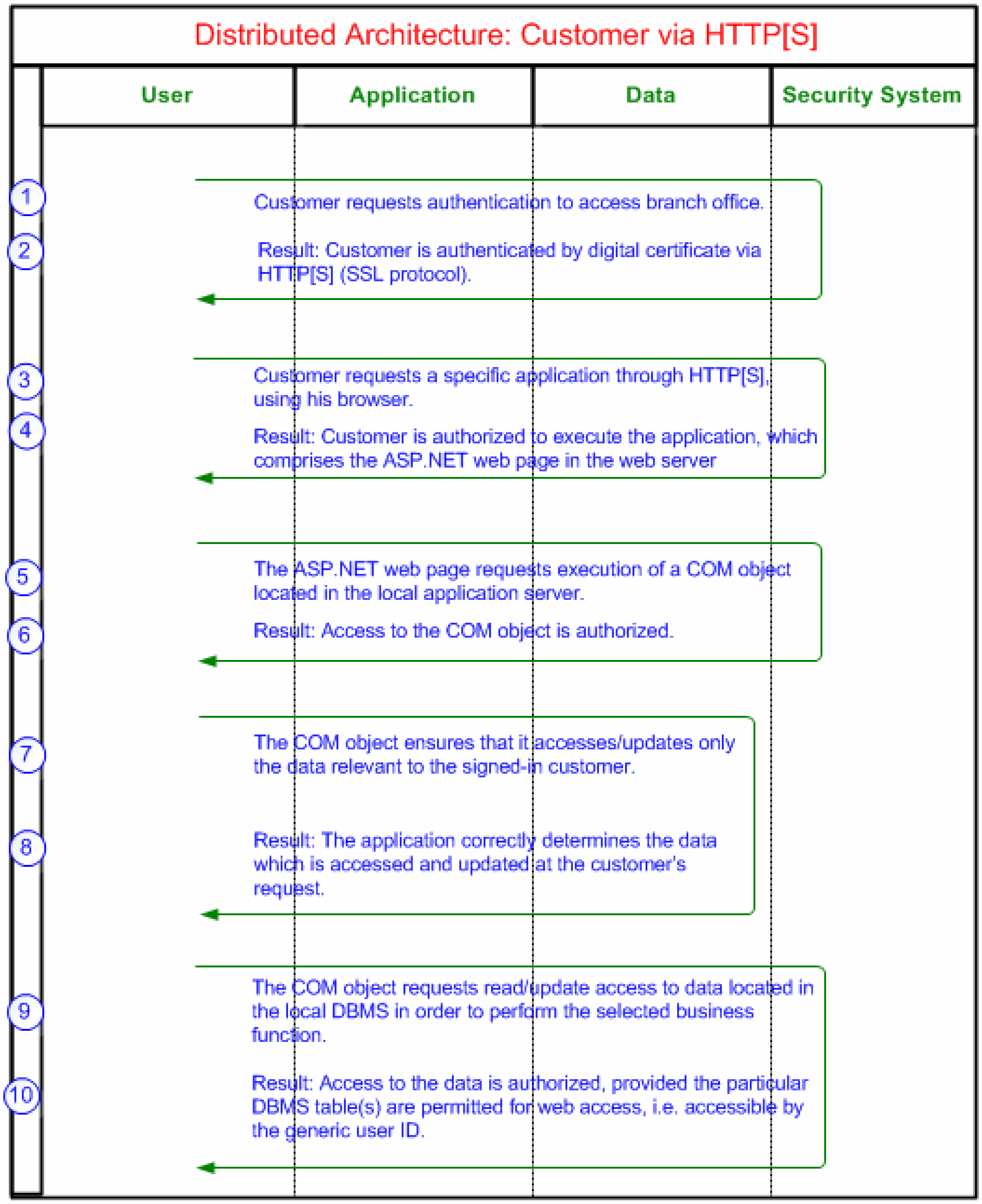


Diagram: Security-Related Process Flow



Customer requests authentication to access branch office.

Result: Customer is authenticated by digital certificate via HTTP[S] (SSL protocol).

Customer requests a specific application through HTTP[S], using his browser.

Result: Customer is authorized to execute the application, which comprises the ASP.NET web page in the web server

The ASP.NET web page requests execution of a COM object located in the local application server.

Result: Access to the COM object is authorized.

The COM object ensures that it accesses/updates only the data relevant to the signed-in customer.

Result: The application correctly determines the data which is accessed and updated at the customer's request.

The COM object requests read/update access to data located in the local DBMS in order to perform the selected business function.

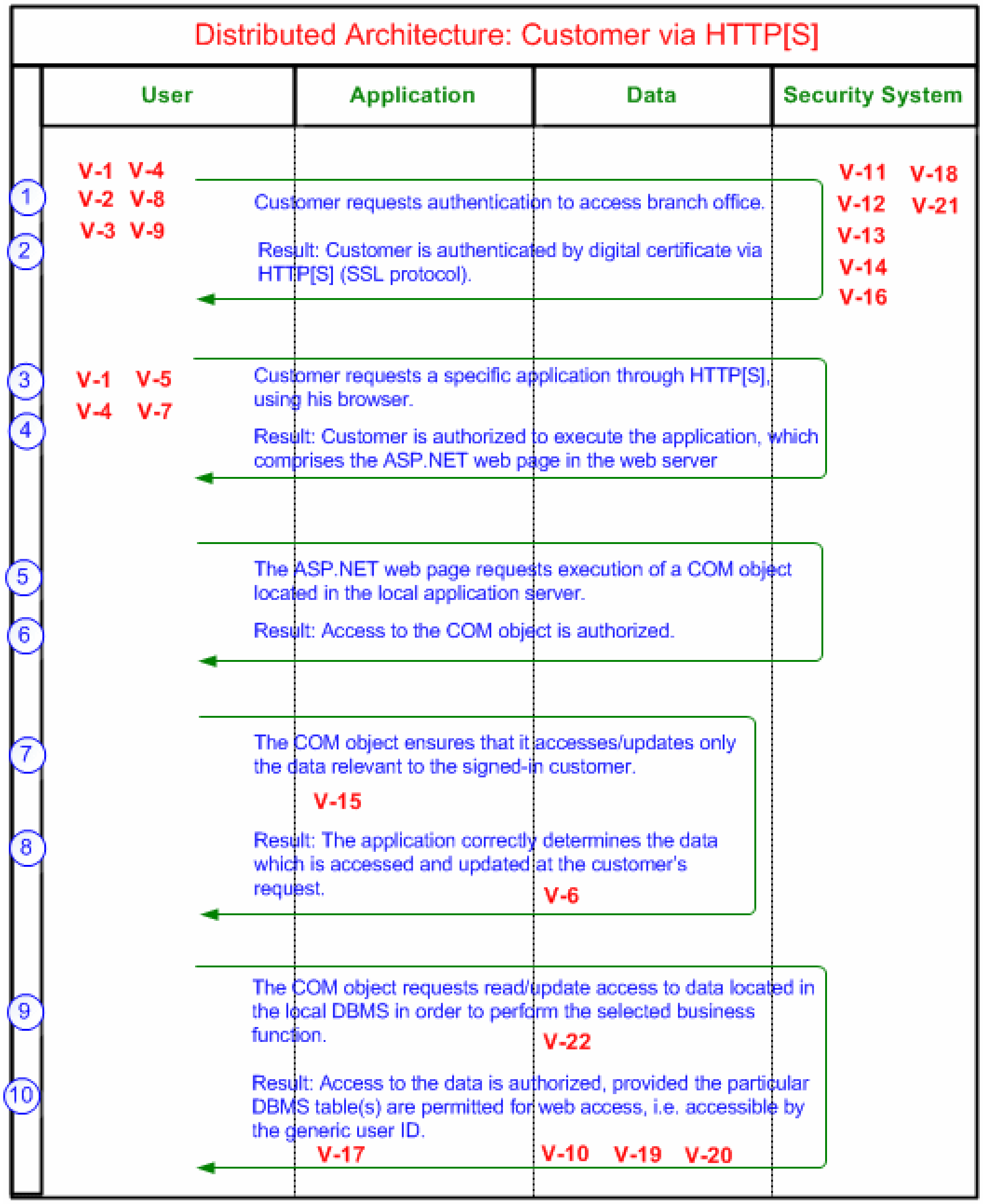
Result: Access to the data is authorized, provided the particular DBMS table(s) are permitted for web access, i.e. accessible by the generic user ID.

Table: Security-Related Process Flow

Step	Description of Step in Security-Related Process Flow
1	The customer accessing the branch office requests authentication. He uses his user name and password in conjunction with his personal digit certificate as part of the sign-in process.
2	The customer accessing the branch office is granted authentication **—** provided the user name password are correct and the digital certificate is valid and not expired. This process is performed over HTTP[S] (SSL protocol) in accordance with a root (CA certificate) residing in the branch office.
3	After sign-in, the customer requests a specific application through HTTP[S], using his browser.
4	The customer is authorized to execute the application, which comprises the ASP.NET web page in the web server **—** provided his authorization extends to the selected web page which he has selected.
5	The ASP.NET web page requests execution of a COM object located in the local application server.
6	Access to the COM objects is authorized **—** provided the request is valid for this particular use (i.e., web access). **Note:** The sub-set of COM objects accessible to the customer differs from that accessible to the employees of the branch office. The customer will be allowed to access the ASP.NET pages; the employees of the branch office do not have this authorization.
7	The COM object must ensure it accesses/updates only the data relevant to the signed-in customer. This is under the control of the application logic built into the COM object.
8	The application correctly determines the data which is accessed and updated at the customer’s request. **Note:** Since the customer does not have a real (i.e., specific) Windows user ID, application-level security is required to ensure that the customer reads only authorized data. The DBMS of the branch office is not able to distinguish between different customers if they all are granted access to the branch office LAN, using a generic user ID.
9	The COM object then requests read/update access to data located in the local DBMS in order to perform the selected business function.
10	Access to the data is authorized provided that the particular DBMS table(s) are permitted for web access **—** i.e. accessible by the generic user ID.

Diagram: Vulnerabilities & Security Services

The following security-related process flow shows a selection of CLASP vulnerabilities that are possible for this process flow.



Customer requests authentication to access branch office.

Result: Customer is authenticated by digital certificate via HTTP[S] (SSL protocol).

Customer requests a specific application through HTTP[S], using his browser.

Result: Customer is authorized to execute the application, which comprises the ASP.NET web page in the web server

The ASP.NET web page requests execution of a COM object located in the local application server.

Result: Access to the COM object is authorized.

The COM object ensures that it accesses/updates only the data relevant to the signed-in customer.

Result: The application correctly determines the data which is accessed and updated at the customer's request.

The COM object requests read/update access to data located in the local DBMS in order to perform the selected business function.

Result: Access to the data is authorized, provided the particular DBMS table(s) are permitted for web access, i.e. accessible by the generic user ID.

Table: Vulnerability-Related Information

The following table provides a stepped description of the security-related process flow depicted in the figure above:

Tag	Vulnerabilities & Security Services
V-1	• Affected Security Service(s): Confidentiality; Authentication; Access Control; Integrity; Accountability • CLASP Problem Type: Injection problem ('"data" used as something else) • Category of Problem Type: Range and Type Errors • Consequence(s): • Confidentiality: Many injection attacks involve the disclosure of important information — in terms of both data sensitivity and usefulness in further exploitation • Authentication: In some cases injectable code controls authentication; this may lead to remote vulnerability • Access Control: Injection attacks are characterized by the ability to significantly change the flow of a given process, and in some cases, to the execution of arbitrary code. • Integrity: Data injection attacks lead to loss of data integrity in nearly all cases as the control-plane data injected is always incidental to data recall or writing. • Accountability: Often the actions performed by injected control code are unlogged.

Tag	Vulnerabilities & Security Services
V-2	• Affected Security Service(s): Confidentiality; Authentication; Authorization; Integrity • CLASP Problem Type: SQL injection • **Category of Problem Type:** Range and Type Errors • Consequence(s): • Confidentiality: Since SQL databases generally hold sensitive data, loss of confidentiality is a frequent problem with SQL injection vulnerabilities. • Authentication: If poor SQL commands are used to check user names and passwords, it may be possible to connect to a system as another user with no previous knowledge of the password. • Authorization: If authorization information is held in an SQL database, it may be possible to change this information through the successful exploitation of an SQL injection vulnerability. • Integrity: Just as it may be possible to read sensitive information, it is also possible to make changes or even delete this information with an SQL injection attack.
V-3	• Affected Security Service(s): Access control • CLASP Problem Type: Command injection • **Category of Problem Type:** Range and Type Errors • **Consequence(s) —** Access control: Command injection allows for the execution of arbitrary commands and code by the attacker.
V-4	• Affected Security Service(s): Confidentiality; Access control • CLASP Problem Type: Cross-site scripting • Category of Problem Type: Range and Type Errors • Consequence(s): • Confidentiality: The most common attack performed with cross-site scripting involves the disclosure of information stored in user cookies. • Access control: In some circumstances it may be possible to run arbitrary code on a victim's computer when cross-site scripting is combined with other flaws

Tag	Vulnerabilities & Security Services
V-5	• Affected Security Service(s): Authorization; Authentication • CLASP Problem Type: Failure to check whether privileges were dropped successfully • **Category of Problem Type:** General Logic Errors • Consequence(s): • Authorization: If privileges are not dropped, neither are access rights of the user. Often these rights can be prevented from being dropped. • Authentication: If privileges are not dropped, in some cases the system may record actions as the user which is being impersonated rather than the impersonator.
V-6	• Affected Security Service(s): Confidentiality • CLASP Problem Type: Accidental leaking of sensitive information through sent data • Category of Problem Type: Synchronization and Timing Errors • **Consequence(s)** — Confidentiality: Data leakage results in the compromise of data confidentiality
V-7	• Affected Security Service(s): Authorization • CLASP Problem Type: Capture-replay • Category of Problem Type: Synchronization and Timing Errors • **Consequence(s)** — Authorization: Messages sent with a capture-relay attack allow access to resources which are not otherwise accessible without proper authentication.
V-8	• Affected Security Service(s): Integrity; Authentication: • CLASP Problem Type: Failure to validate host-specific certificate data • **Category of Problem Type:** Protocol Errors • Consequence(s): • Integrity: The data read from the system vouched for by the certificate may not be from the expected system. • Authentication: Trust afforded to the system in question — based on the expired certificate — may allow for spoofing or redirection attacks.

Tag	Vulnerabilities & Security Services
V-9	• Affected Security Service(s): Authentication; Integrity; Confidentiality: • CLASP Problem Type: Failure to check for certificate revocation • **Category of Problem Type:** Protocol Errors • Consequence(s): • Authentication: Trust may be assigned to an entity who is not who it claims to be. • Integrity: Data from an untrusted (and possibly malicious) source may be integrated. • Confidentiality: Date may be disclosed to an entity impersonating a trusted entity, resulting in information disclosure.
V-10	• Affected Security Service(s): Confidentiality; Integrity; Accountability • CLASP Problem Type: Failure to encrypt data • **Category of Problem Type:** Protocol Errors • Consequence(s): • Confidentiality: Properly encrypted data channels ensure data confidentiality. • Integrity: Properly encrypted data channels ensure data integrity. • Accountability: Properly encrypted data channels ensure accountability.
V-11	• Affected Security Service(s): Authentication • CLASP Problem Type: Use of hard-coded cryptographic key • **Category of Problem Type:** Protocol Errors • **Consequence(s)** — Authentication: If hard-coded cryptographic keys are used, it is almost certain that malicious users will gain access through the account in question.
V-12	• Affected Security Service(s): Authentication • CLASP Problem Type: Trusting self-reported IP address • **Category of Problem Type:** Protocol Errors • **Consequence(s)** — Authentication: Malicious users can fake authentication information, impersonating any IP address.

Tag	Vulnerabilities & Security Services
V-13	• Affected Security Service(s): Authentication • CLASP Problem Type: Trusting self-reported DNS name • **Category of Problem Type:** Protocol Errors • Consequence(s) — Authentication: Malicious users can fake authentication information by providing false DNS information.
V-14	• Affected Security Service(s): Authentication • CLASP Problem Type: Using a key past its expiration date • **Category of Problem Type:** Protocol Errors • Consequence(s) — Authentication: The cryptographic key in question may be compromised, providing a malicious user with a method for authenticating as the victim.
V-15	• Affected Security Service(s): Availability; Authentication: • CLASP Problem Type: Insufficient entropy in PRNG • **Category of Problem Type:** Environmental Problems • Consequence(s): • Availability: If a pseudo-random number generator is using a limited entropy source which runs out (if the generator fails closed), the program may pause or crash. • Authentication: If a PRNG is using a limited entropy source which runs out, and the generator fails open, the generator could produce predictable random numbers. Potentially a weak source of random numbers could weaken the encryption method used for authentication of users. In this case, potentially a password could be discovered.
V-16	• Affected Security Service(s): Authentication; Integrity; Confidentiality • CLASP Problem Type: Race condition in checking for certificate revocation • **Category of Problem Type:** Synchronization and Timing Errors • Consequence(s) • Authentication: Trust may be assigned to an entity who is not who it claims to be. • Integrity: Data from an untrusted (and possibly malicious) source may be integrated. • Confidentiality: Date may be disclosed to an entity impersonating a trusted entity, resulting in information disclosure.

Tag	Vulnerabilities & Security Services
V-17	• **Affected Security Service(s):** Confidentiality • **CLASP Problem Type:** Covert storage channel • **Category of Problem Type:** Range and Type Errors • **Consequence(s)** — Confidentiality: Covert storage channels may provide attackers with important information about the system in question.
V-18	• Affected Security Service(s): Authentication; Accountability • CLASP Problem Type: Failure to follow chain of trust in certificate validation • **Category of Problem Type:** Synchronization and Timing Errors • Consequence(s): • Authentication: Exploitation of this flaw can lead to the trust of data that may have originated with a spoofed source. • Accountability: Data, requests, or actions taken by the attacking entity can be carried out as a spoofed benign entity.
V-19	• Affected Security Service(s): Confidentiality • CLASP Problem Type: Accidental leaking of sensitive information through data queries • **Category of Problem Type:** Synchronization and Timing Errors • **Consequence(s)** — Confidentiality: Sensitive information may possibly be through data queries accidentally.
V-20	• Affected Security Service(s): Integrity; Availability; Access Control • CLASP Problem Type: Using freed memory • **Category of Problem Type:** Range and Type Errors • Consequence(s): • Integrity: The use of previously freed memory may corrupt valid data, if the memory area in question has been allocated and used properly elsewhere. • Availability: If chunk consolidation occur after the use of previously freed data, the process may crash when invalid data is used as chunk information. • Access Control (instruction processing): If malicious data is entered before chunk consolidation can take place, it may be possible to take advantage of a write-what-where primitive to execute arbitrary code.

Tag	Vulnerabilities & Security Services
V-21	• Affected Security Service(s): Authentication; Confidentiality • **CLASP Problem Type:** Key exchange without entity authentication • **Category of Problem Type:** Protocol Errors • Consequence(s): • Authentication: No authentication takes place in this process, bypassing an assumed protection of encryption • Confidentiality: The encrypted communication between a user and a trusted host may be subject to a “man-in-the-middle” sniffing attack
V-22	• Affected Security Service(s): Availability; Access control (instruction processing); Other • **CLASP Problem Type:** Buffer overflow • **Category of Problem Type:** Range and type errors • Consequence(s): • Availability: Buffer overflows generally lead to crashes. Other attacks leading to lack of availability are possible, including putting the program into an infinite loop. • Access control (instruction processing): Buffer overflows often can be used to execute arbitrary code, which is usually outside the scope of a program’s implicit security policy. • Other: When the consequence is arbitrary code execution, this can often be used to subvert any other security service.

Branch Office's Participation

This section describes the branch office's participation in the security-related process flow. The branch office's participation consists of three high-level, security-related activities: authentication; authorization to execute; and authorization to access data.

Diagram: Component Architecture

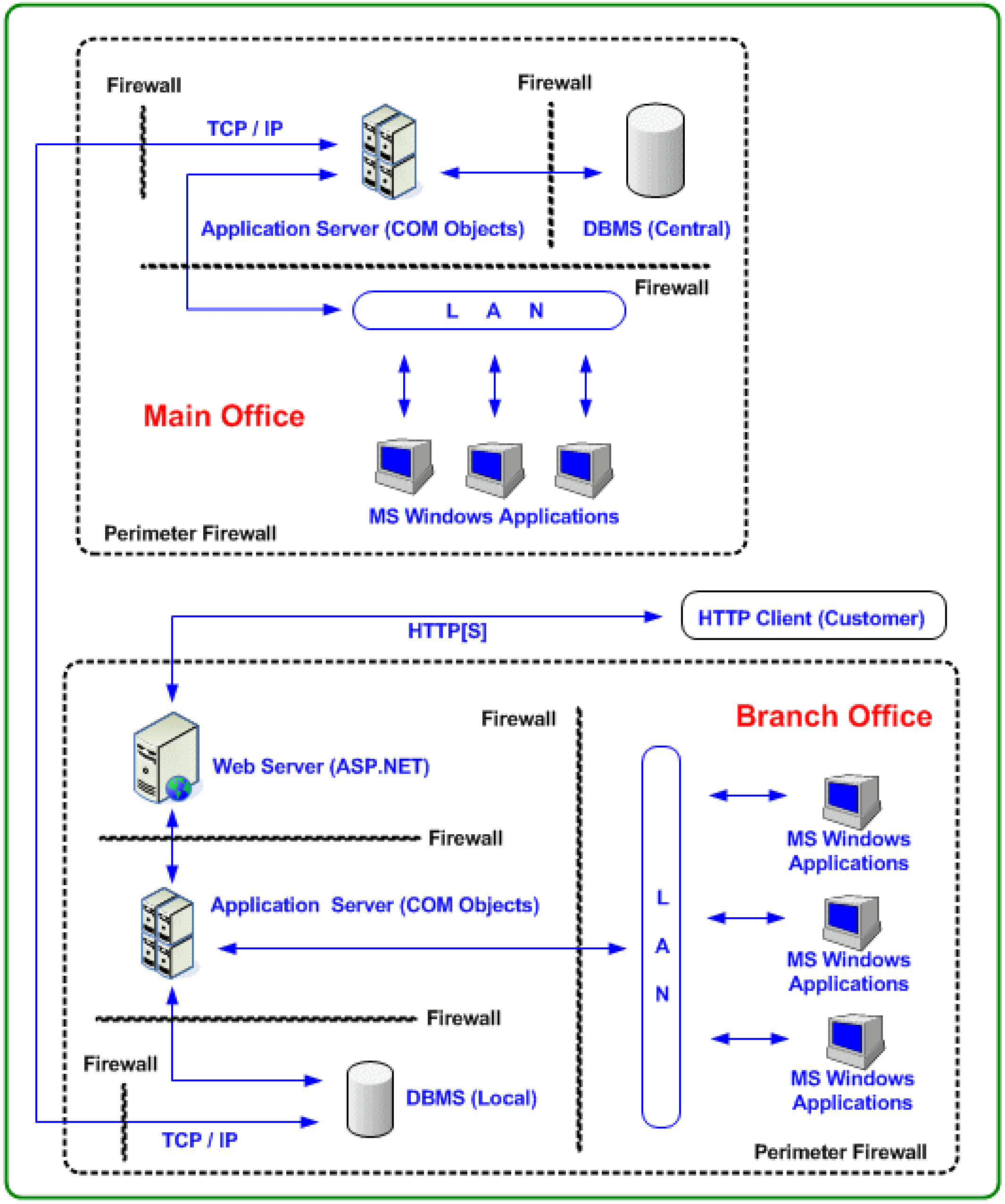
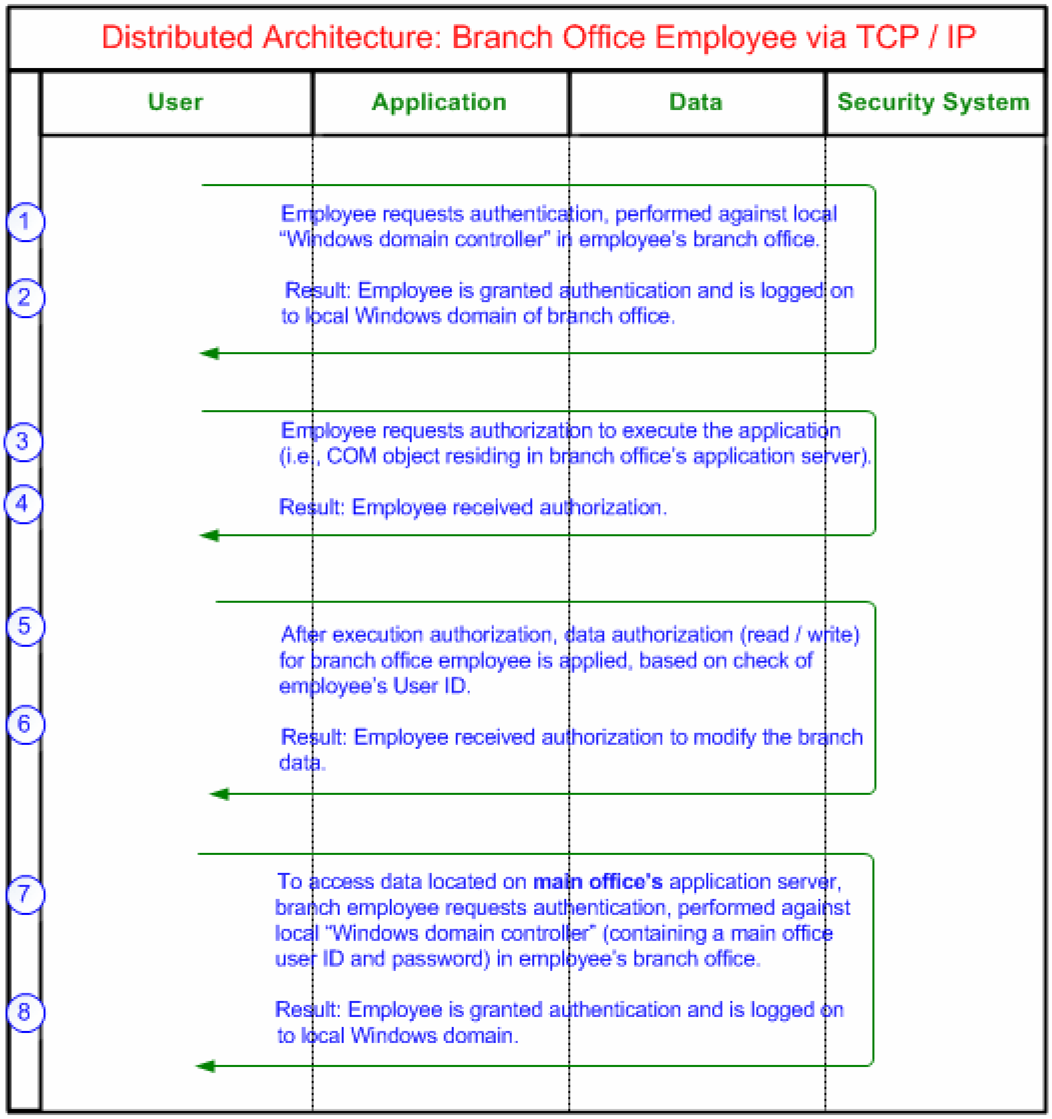


Diagram: Security-Related Process Flow


Distributed Architecture: Branch Office Employee via TCP / IP
User
Application
Data
Security System
1
2
3
4
5
6
7
8

Employee requests authentication, performed against local "Windows domain controller" in employee's branch office.

Result: Employee is granted authentication and is logged on to local Windows domain of branch office.

Employee requests authorization to execute the application (i.e., COM object residing in branch office's application server).

Result: Employee received authorization.

After execution authorization, data authorization (read / write) for branch office employee is applied, based on check of employee's User ID.

Result: Employee received authorization to modify the branch data.

To access data located on **main office's** application server, branch employee requests authentication, performed against local "Windows domain controller" (containing a main office user ID and password) in employee's branch office.

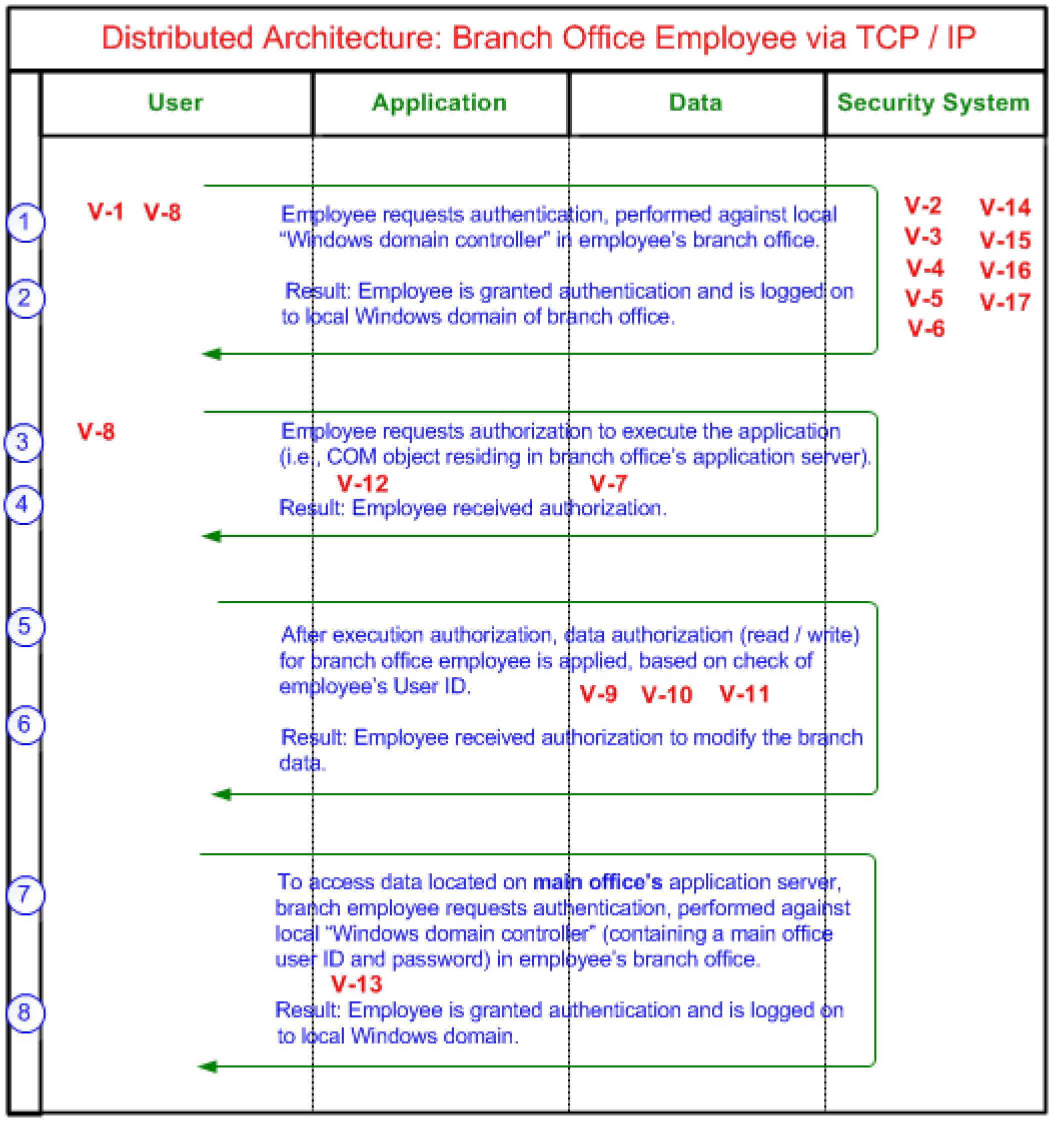
Result: Employee is granted authentication and is logged on to local Windows domain.

Table: Security-Related Process Flow

Step	Description of Step in Security-Related Process Flow
1	The employee in a branch office requests authentication.
2	Result: The employee is logged on to the local Windows domain. **Note:** Authentication is performed against the "local Windows domain controller" in the branch office of the employee, which holds user ID's/password for each physical site.
3	The branch office employee requests authorization to execute the application — i.e., actually the COM object residing in the application server of the branch office. **Note:** Not all COM objects are accessible to all users, based on the logged on Windows Domain user ID.
4	Result: The branch office employee is authorized to execute the application.
5	After execution authorization is performed, data authorization (read/update) for the branch office employee is applied. The authorization is based on the specific user ID in use — i.e., that of the main office user, branch office user, or customer. This authorization check is performed by the security component of the DBMS working in conjunction with the Windows user ID.
6	Result: The branch office employee is authorized to modify branch office data.
7	To access data located on **main office's** DBMS via COM objects located within the main office's application server, the branch employee requests authentication, performed against local “Windows domain controller” (containing a main office user ID and password) in employee's branch office.
8	Result: Employee of branch office is granted authentication and is logged on to local Windows domain for the purpose of accessing data on the application server of main office.

Diagram: Vulnerabilities & Security Services

The following security-related process flow shows a selection of CLASP vulnerabilities that are possible for this process flow.



Distributed Architecture: Branch Office Employee via TCP / IP

Employee requests authentication, performed against local "Windows domain controller" in employee's branch office.

Result: Employee is granted authentication and is logged on to local Windows domain of branch office.

Employee requests authorization to execute the application (i.e., COM object residing in branch office's application server).

Result: Employee received authorization.

After execution authorization, data authorization (read / write) for branch office employee is applied, based on check of employee's User ID.

Result: Employee received authorization to modify the branch data.

To access data located on **main office's** application server, branch employee requests authentication, performed against local "Windows domain controller" (containing a main office user ID and password) in employee's branch office.

Result: Employee is granted authentication and is logged on to local Windows domain.

Table: Vulnerability-Related Information

The following table provides a stepped description of the security-related process flow depicted in the figure above:

Tag	Vulnerabilities & Security Services
V-1	• Affected Security Service(s): Authentication • **CLASP Problem Type:** Use of hard-coded password • **Category of Problem Type:** Protocol errors • **Consequence(s) —** Authentication: If hard-coded passwords are used, it is almost certain that malicious users will gain access through the account in question.
V-2	• Affected Security Service(s): Authentication • **CLASP Problem Type:** Using password systems • **Category of Vulnerability:** Protocol errors • **Consequence(s) —** Authentication: The failure of a password authentication mechanism will almost always result in attackers being authorized as valid users.
V-3	• Affected Security Service(s): Authentication • **CLASP Problem Type:** Allowing password aging • **Category of Problem Type:** Protocol errors • **Consequence(s) —** Authentication: As passwords age, the probability that they are compromised grows.
V-4	• **Affected Security Service(s):** Authentication • **CLASP Problem Type:** Not allowing password aging • **Category of Problem Type:** Protocol errors • **Consequence(s) —** Authentication: As passwords age, the probability that they are compromised grows.

Tag	Vulnerabilities & Security Services
V-5	• **Affected Security Service(s):** Confidentiality; Authentication • **CLASP Problem Type:** Storing passwords in a recoverable format • **Category of Problem Type:** Protocol errors • Consequence(s): • Confidentiality: User's passwords may be revealed. • Authentication: Revealed passwords may be reused elsewhere to impersonate the users in question.
V-6	• **Affected Security Service(s):** Authentication • **CLASP Problem Type:** Using single-factor authentication • **Category of Problem Type:** Protocol errors • **Consequence(s) —** Authentication: If the secret in a single-factor authentication scheme gets compromised, full authentication is possible.
V-7	• **Affected Security Service(s):** Integrity • **Vulnerability:** Failure to protect stored data from modification • **Category of Problem Type:** Protocol errors • **Consequence(s) —** Integrity: The object could be tampered with.
V-8	• **Affected Security Service(s):** Authorization; Authentication • **CLASP Problem Type:** Failure to check whether privileges were dropped successfully • **Category of Problem Type:** General logic errors • Consequence(s): • Authorization: If privileges are not dropped, neither are access rights of the user. Often these rights can be prevented from being dropped. • Authentication: If privileges are not dropped, in some cases the system may record actions as the user which is being impersonated rather than the impersonator.

Tag	Vulnerabilities & Security Services
V-9	• Affected Security Service(s): Availability; Access control (instruction processing); Other • **CLASP Problem Type:** Buffer overflow • **Category of Problem Type:** Range and type errors • Consequence(s): • Availability: Buffer overflows generally lead to crashes. Other attacks leading to lack of availability are possible, including putting the program into an infinite loop. • Access control (instruction processing): Buffer overflows often can be used to execute arbitrary code, which is usually outside the scope of a program's implicit security policy. • Other: When the consequence is arbitrary code execution, this can often be used to subvert any other security service.
V-10	• Affected Security Service(s): Confidentiality • CLASP Problem Type: Accidental leaking of sensitive information through sent data • Category of Problem Type: Synchronization and Timing Errors • **Consequence(s)** — Confidentiality: Data leakage results in the compromise of data confidentiality
V-11	• Affected Security Service(s): Confidentiality; Integrity; Accountability • CLASP Problem Type: Failure to encrypt data • **Category of Problem Type:** Protocol Errors • Consequence(s): • Confidentiality: Properly encrypted data channels ensure data confidentiality. • Integrity: Properly encrypted data channels ensure data integrity. • Accountability: Properly encrypted data channels ensure accountability.
V-12	• Affected Security Service(s): Authorization • CLASP Problem Type: Comparing classes by name • **Category of Problem Type:** Synchronization and Timing Errors • **Consequence(s)** — Authorization: If a program trusts, based on the name of the object, to assume that it is the correct object, it may execute the wrong

Tag	Vulnerabilities & Security Services
	program.
V-13	• **Affected Security Service(s):** Confidentiality • **CLASP Problem Type:** Covert storage channel • **Category of Problem Type:** Range and Type Errors • **Consequence(s)** — Confidentiality: Covert storage channels may provide attackers with important information about the system in question.
V-14	• Affected Security Service(s): Authentication • CLASP Problem Type: Use of hard-coded cryptographic key • **Category of Problem Type:** Protocol Errors • **Consequence(s)** — Authentication: If hard-coded cryptographic keys are used, it is almost certain that malicious users will gain access through the account in question.
V-15	• Affected Security Service(s): Authentication • CLASP Problem Type: Trusting self-reported IP address • **Category of Problem Type:** Protocol Errors • **Consequence(s)** — Authentication: Malicious users can fake authentication information, impersonating any IP address.
V-16	• Affected Security Service(s): Authentication • CLASP Problem Type: Trusting self-reported DNS name • **Category of Problem Type:** Protocol Errors • Consequence(s) — Authentication: Malicious users can fake authentication information by providing false DNS information.

Tag	Vulnerabilities & Security Services
V-17	• Affected Security Service(s): Authentication • CLASP Problem Type: Using a key past its expiration date • **Category of Problem Type:** Protocol Errors • Consequence(s) — Authentication: The cryptographic key in question may be compromised, providing a malicious user with a method for authenticating as the victim.

Main Office's Participation

This section describes the main office's participation in the security-related process flow. The main office's participation consists of three high-level, security-related activities: authentication; authorization to execute; and authorization to access data.

Diagram: Component Architecture

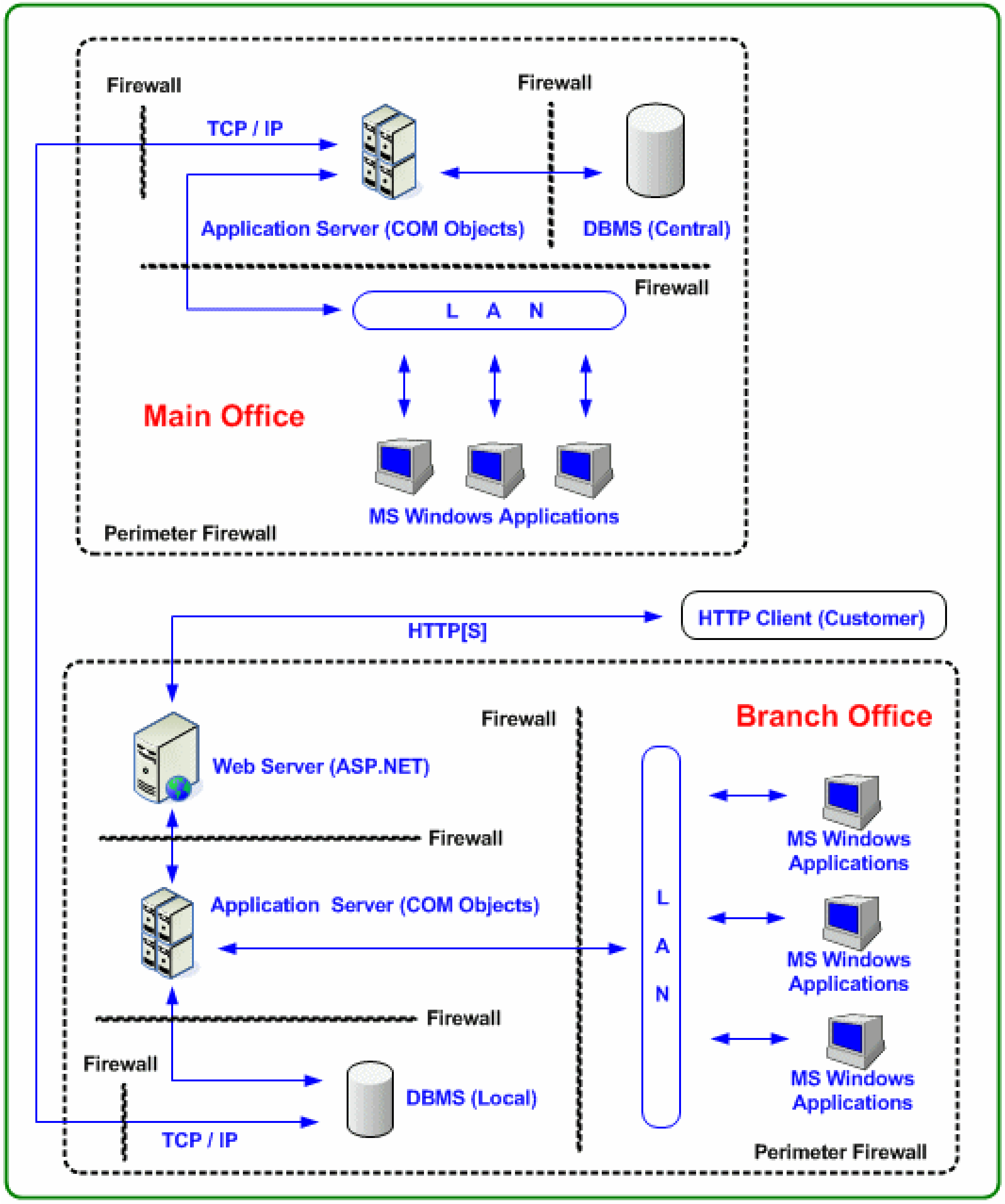
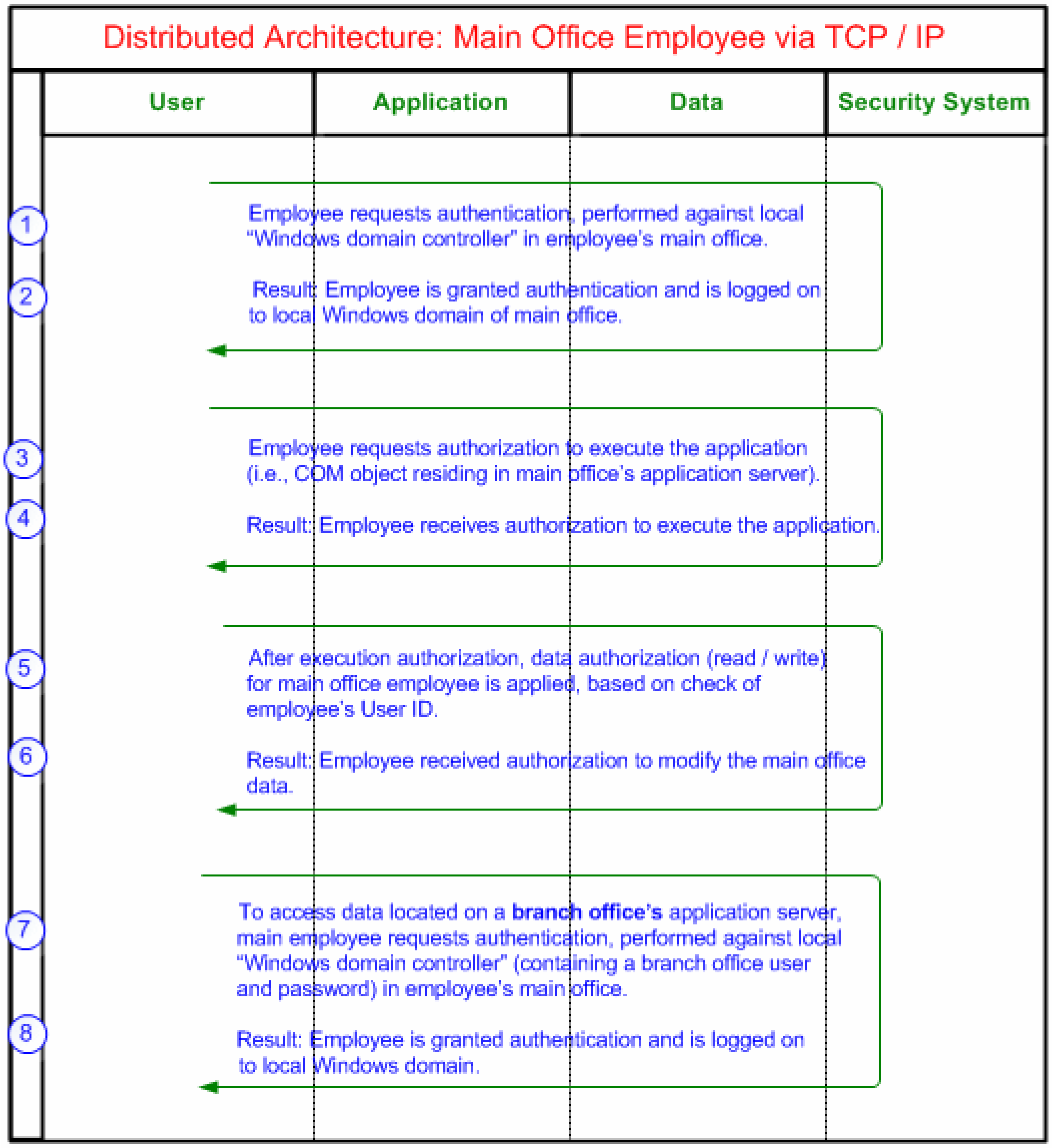


Diagram: Security-Related Process Flow



Employee requests authentication, performed against local "Windows domain controller" in employee's main office.

Result: Employee is granted authentication and is logged on to local Windows domain of main office.

Employee requests authorization to execute the application (i.e., COM object residing in main office's application server).

Result: Employee receives authorization to execute the application.

After execution authorization, data authorization (read / write) for main office employee is applied, based on check of employee's User ID.

Result: Employee received authorization to modify the main office data.

To access data located on a **branch office's** application server, main employee requests authentication, performed against local "Windows domain controller" (containing a branch office user and password) in employee's main office.

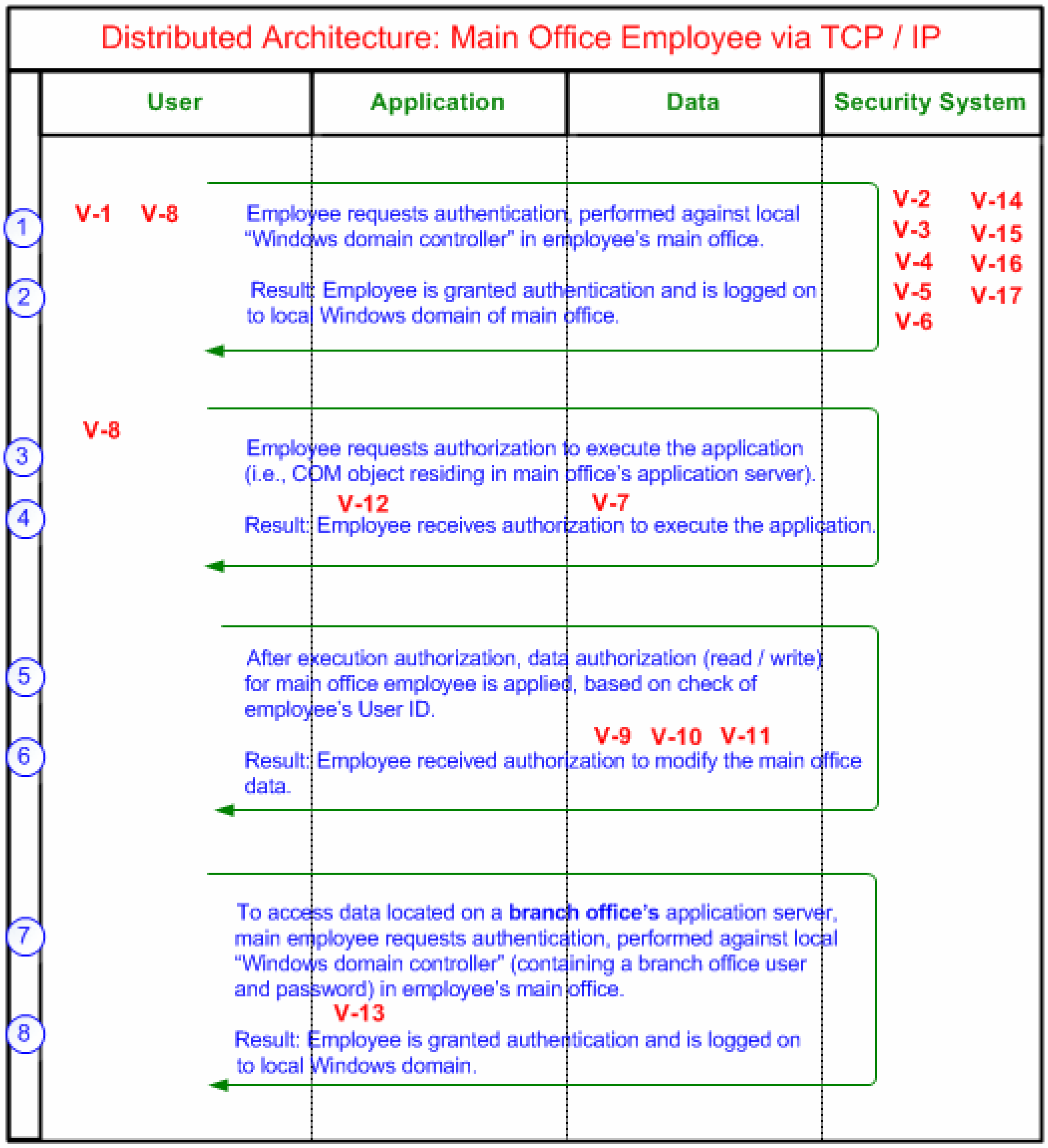
Result: Employee is granted authentication and is logged on to local Windows domain.

Table: Security-Related Process Flow

Step	Description of Step in Security-Related Process Flow
1	The employee in the main office requests authentication.
2	Result: The employee is logged on to the local Windows domain of the main office. **Note:** Authentication is performed against the "local Windows domain controller" of the main office, which holds user ID's/password for each physical site.
3	The main office employee requests authorization to execute the application — i.e., actually the COM object residing in the application server of the main office. **Note:** Not all COM objects are accessible to all users, based on the logged on Windows Domain user ID.
4	Result: The main office employee is authorized to execute the application.
5	After execution authorization is performed, data authorization (read/update) for the main office employee is applied. The authorization is based on the specific user ID in use — i.e., that of the main office user, branch office user, or customer. This authorization check is performed by the security component of the DBMS working in conjunction with the Windows user ID.
6	Result: The main office employee is authorized to modify data residing on the main office's application server.
7	To access data located on a **local office's** DBMS via COM objects located within their own application server, the main employee requests authentication, performed against local "Windows domain controller" (containing a local office user ID and password) in employee's branch office. **Note:** The branch office Domain Controller needs to hold a "main office" user ID.
8	Result: The Employee of the main office is granted authentication and is logged on to local Windows domain for the purpose of accessing data on the application server of a local office.

Diagram: Vulnerabilities & Security Services

The following security-related process flow shows a selection of CLASP vulnerabilities that are possible for this process flow.



Distributed Architecture: Main Office Employee via TCP / IP

Employee requests authentication, performed against local "Windows domain controller" in employee's main office.

Result: Employee is granted authentication and is logged on to local Windows domain of main office.

Employee requests authorization to execute the application (i.e., COM object residing in main office's application server).

Result: Employee receives authorization to execute the application.

After execution authorization, data authorization (read / write) for main office employee is applied, based on check of employee's User ID.

Result: Employee received authorization to modify the main office data.

To access data located on a **branch office's** application server, main employee requests authentication, performed against local "Windows domain controller" (containing a branch office user and password) in employee's main office.

Result: Employee is granted authentication and is logged on to local Windows domain.

Table: Vulnerability-Related Information

The following table provides a stepped description of the security-related process flow depicted in the figure above:

Tag	Vulnerabilities & Security Services
V-1	• Affected Security Service(s): Authentication • **CLASP Problem Type:** Use of hard-coded password • **Category of Problem Type:** Protocol errors • **Consequence(s) —** Authentication: If hard-coded passwords are used, it is almost certain that malicious users will gain access through the account in question.
V-2	• Affected Security Service(s): Authentication • **CLASP Problem Type:** Using password systems • **Category of Vulnerability:** Protocol errors • **Consequence(s) —** Authentication: The failure of a password authentication mechanism will almost always result in attackers being authorized as valid users.
V-3	• Affected Security Service(s): Authentication • **CLASP Problem Type:** Allowing password aging • **Category of Problem Type:** Protocol errors • **Consequence(s) —** Authentication: As passwords age, the probability that they are compromised grows.
V-4	• **Affected Security Service(s):** Authentication • **CLASP Problem Type:** Not allowing password aging • **Category of Problem Type:** Protocol errors • **Consequence(s) —** Authentication: As passwords age, the probability that they are compromised grows.

Tag	Vulnerabilities & Security Services
V-5	• **Affected Security Service(s):** Confidentiality; Authentication • **CLASP Problem Type:** Storing passwords in a recoverable format • **Category of Problem Type:** Protocol errors • Consequence(s): • Confidentiality: User’s passwords may be revealed. • Authentication: Revealed passwords may be reused elsewhere to impersonate the users in question.
V-6	• **Affected Security Service(s):** Authentication • **CLASP Problem Type:** Using single-factor authentication • **Category of Problem Type:** Protocol errors • **Consequence(s) —** Authentication: If the secret in a single-factor authentication scheme gets compromised, full authentication is possible.
V-7	• **Affected Security Service(s):** Integrity • **Vulnerability:** Failure to protect stored data from modification • **Category of Problem Type:** Protocol errors • **Consequence(s) —** Integrity: The object could be tampered with.
V-8	• **Affected Security Service(s):** Authorization; Authentication • **CLASP Problem Type:** Failure to check whether privileges were dropped successfully • **Category of Problem Type:** General logic errors • Consequence(s): • Authorization: If privileges are not dropped, neither are access rights of the user. Often these rights can be prevented from being dropped. • Authentication: If privileges are not dropped, in some cases the system may record actions as the user which is being impersonated rather than the impersonator.

Tag	Vulnerabilities & Security Services
V-9	• Affected Security Service(s): Availability; Access control (instruction processing); Other • **CLASP Problem Type:** Buffer overflow • **Category of Problem Type:** Range and type errors • Consequence(s): • Availability: Buffer overflows generally lead to crashes. Other attacks leading to lack of availability are possible, including putting the program into an infinite loop. • Access control (instruction processing): Buffer overflows often can be used to execute arbitrary code, which is usually outside the scope of a program's implicit security policy. • Other: When the consequence is arbitrary code execution, this can often be used to subvert any other security service.
V-10	• Affected Security Service(s): Confidentiality • CLASP Problem Type: Accidental leaking of sensitive information through sent data • Category of Problem Type: Synchronization and Timing Errors • **Consequence(s)** — Confidentiality: Data leakage results in the compromise of data confidentiality
V-11	• Affected Security Service(s): Confidentiality; Integrity; Accountability • CLASP Problem Type: Failure to encrypt data • **Category of Problem Type:** Protocol Errors • Consequence(s): • Confidentiality: Properly encrypted data channels ensure data confidentiality. • Integrity: Properly encrypted data channels ensure data integrity. • Accountability: Properly encrypted data channels ensure accountability.
V-12	• Affected Security Service(s): Authorization • CLASP Problem Type: Comparing classes by name • **Category of Problem Type:** Synchronization and Timing Errors • **Consequence(s)** — Authorization: If a program trusts, based on the name of the object, to assume that it is the correct object, it may execute the wrong

Tag	Vulnerabilities & Security Services
	program.
V-13	• **Affected Security Service(s):** Confidentiality • **CLASP Problem Type:** Covert storage channel • **Category of Problem Type:** Range and Type Errors • **Consequence(s)** — Confidentiality: Covert storage channels may provide attackers with important information about the system in question.
V-14	• Affected Security Service(s): Authentication • CLASP Problem Type: Use of hard-coded cryptographic key • **Category of Problem Type:** Protocol Errors • **Consequence(s)** — Authentication: If hard-coded cryptographic keys are used, it is almost certain that malicious users will gain access through the account in question.
V-15	• Affected Security Service(s): Authentication • CLASP Problem Type: Trusting self-reported IP address • **Category of Problem Type:** Protocol Errors • **Consequence(s)** — Authentication: Malicious users can fake authentication information, impersonating any IP address.
V-16	• Affected Security Service(s): Authentication • CLASP Problem Type: Trusting self-reported DNS name • **Category of Problem Type:** Protocol Errors • Consequence(s) — Authentication: Malicious users can fake authentication information by providing false DNS information.

Tag	Vulnerabilities & Security Services
V-17	• Affected Security Service(s): Authentication • CLASP Problem Type: Using a key past its expiration date • **Category of Problem Type:** Protocol Errors • Consequence(s) — Authentication: The cryptographic key in question may be compromised, providing a malicious user with a method for authenticating as the victim.