AHCCCS

Arizona Medical Information Exchange

Security & Privacy
Procedures and Standards Manual

September 29, 2008

Table of Contents

Chapter 100 – Definitions

101: Definitions

The definitions and terms below are used by Arizona Medical Information Exchange and found in this procedure manual:

Action Plan means a set of steps, dates and responsibly party to complete a process to mitigate a risk.

AHCCCS HIE (or Utility) means the AHCCCS Health Information Exchange Utility, which is the project that is implementing the Arizona Medical Information Exchange application.

AMIE (or "Exchange") means the Arizona Medical Information Exchange application that uses a Record Locator Service to provide access to electronic health Data by use of the Viewer web application found at www.azhealthyrecord.gov

Authorized User (or User) means an individual authorized by AHCCCS HIE under a Participation Agreement to use the Exchange to access Data for a Permitted Use.

Breach, **"**Breach of the security of the system", "Breach of the security system" or "security Breach" means an unauthorized acquisition of and access to unencrypted or unredacted computerized Data that materially compromises the security or confidentiality of personal information maintained by a person as part of a database of personal information regarding multiple individuals and that causes or is reasonably likely to cause substantial economic loss to an individual. Good faith acquisition of personal information by an employee or agent of the person for the purposes of the person is not a breach of the security system if the personal information is not used for a purpose unrelated to the person or subject to further willful unauthorized disclosure. (Arizona Revised Statutes, ARS §44-7501).

Computer Incident Response Team (CIRT) is a team of individuals trained in information security incident handling and appointed by AHCCCS management to investigate computer incidents and potential Data Breaches.

Data means protected health information provided to the Exchange by Data Suppliers. Protected Health Information is defined by the Health Insurance Portability and Accountability Act (HIPAA) Standards for Privacy of Individually Identifiable Health Information, 45 C.F.R. Part 160 and Part 164, Subpart E, and the HIPAA Security Standards, 45 C.F.R. Part 160 and Part 164, Subpart C, both as amended from time to time.

Data Exchange means electronically providing or accessing Data through the AMIE.
Data Supplier means an organization, such as a hospital, clinical laboratory, pharmacy claims aggregation company, or otherwise that makes Data available for

access through the AMIE and has entered into a Participation Agreement. A Data Supplier also may be a Health Care Provider.

Electronic Protected Health Information (EPHI) means Protected Health Information that is created, received, maintained or transmitted in electronic format, as defined in the HIPAA Security rule, 45 C.F.R. Part 160 and Part 164.

Health Care Provider means a physician, group practice, hospital or health system, or other health care organization or professional that provides Treatment to Patients, has been assigned an AHCCCS provider number, and has entered into a Participation Agreement with the AHCCCS HIE. A Health Care Provider also may be a Data Supplier as well as an Authorized User.

Health Information Exchange (HIE) means a multi-stakeholder entity that enables the movement of health-related Data within state, regional, or non-jurisdictional participant groups.

HITSP means Health Information Technology Standards Panel.

Member means an individual who receives, or has received, services from AHCCCS.

Participant means a Health Care Provider (HCP), or an employee of a HCP who is an AHCCCS registered Provider, and/or a Data Supplier that has entered into a Participation Agreement with AHCCCS HIE.

Participation Agreement means the agreement between a Participant and the AHCCCS HIE Utility for the purpose of a HCP's use of the AMIE, or a Data Supplier's transmission of Data through the AMIE and the submission or use of such Data.

Patient means an individual receiving medical Treatment or health care services from a Health Care Provider.

Permitted Use means the reason or reasons Participants and Authorized Users may access Data in the Exchange and use the Data included in the Exchange. The sole "Permitted Use" is a Health Care Provider or Authorized User's access to the Exchange to obtain Data for Treatment of Health Care Provider's Patients. If a Health Care Provider includes Data in its Medical Record, Health Care Provider and Authorized Users may use the Data only for those purposes permitted by law.

Policy means the AMIE Privacy & Security Policy Manual and the AMIE Procedures and Standards Manual.

Protected Health Information (PHI) is defined by the Health Insurance Portability and Accountability Act (HIPAA) Standards for Privacy of Individually Identifiable Health

Information, 45 C.F.R. Part 160 and Part 164, Subpart E, and the HIPAA Security Standards, 45 C.F.R. Part 160 and Part 164, Subpart C, both as amended from time to time.

Record Locator Service (RLS) means an information service that locates patient records across systems given a set of criteria, such as patient demographics or ID numbers.

Role Based Access Principles means providing permission to access particular Data and system functions based upon assigned User job roles.

Sanitization means to alter or mask sensitive Data elements contained in a database.

Scrambled means a process of Microsoft SQL or other similar software to rearrange characters in Data so any Data, such as name, address, phone number, SSN, etc. is no longer the same as it was originally.

Secure Sockets Layer (SSL) means a commonly-used protocol for managing the security of Data transmission on the Internet.

Security incident (or Incident) means the attempted or successful unauthorized access, use, disclosure, modification, or destruction of information or interference with system operations in an information system, as defined by the HIPAA Security rule, 45 C.F.R. § 164.304.

Treatment means the provision, coordination or management of health care services by one or more health care providers, as defined the HIPAA Privacy rule, 45 C.F.R. § 164.501.

Viewer means the AMIE web-browser based application designed to allow Users to search for specific patient related Data, and view a single non-aggregate record through the Exchange.

Chapter 200 – Securing Patient Information

201: Provision of Information to Patients

In keeping with the AHCCCS commitment to maintain the Data and PHI transmitted through the Exchange secure and confidential, and in providing information to Patients, Members and the public about the Arizona Medical Information Exchange (AMIE), AHCCCS will publish information to describe how PHI is used and disclosed for Treatment. The methods of providing information may include any of the following: (1) Member education; (2) public education; and (3) provision of information at the point of service.

202: Privacy and Confidentiality

a. AMIE Development and Architecture Teams will work with Data Suppliers technical teams to approve SSL certificates.
b. AMIE Development and Architecture Teams will work with Participant technical teams to approve message-level security methods.
c. EPHI provided by Data Suppliers required for development and testing (dev/test) purposes by AHCCCS HIE will be encrypted for transmission between Data Supplier and AHCCCS HIE. Data Supplier will export a number of production data records to be sent to AHCCCS HIE and used in dev/test. Data Supplier will encrypt and password protect this file, and a secure web-based email system to make the data available to AHCCCS HIE. An email message with a link to the secure web email server allows AHCCCS HIE staff to securely download the data file. A separate secure email message will include the password to decrypt the file.

203: Authorized Access to Patient Data

Only certain clinicians will be authorized to access Patient Data. In order to view information, a clinician or their organizations must sign a contract guaranteeing that they will only access information if it is needed for Treatment. They may only view information after logging on using their UserID and a secure password. Except for those individuals who will be administrating the Exchange, Patient Data will not be available to anyone other than clinicians. It will not be available to employers, insurance companies, researchers, or any other person.

An Authorized User is an individual authorized by AHCCCS HIE to use the AMIE to access Data for Treatment purposes. All Users will be clinicians (e.g., physicians, nurse clinicians, physician assistants, etc.) or an employee of a HCP who is an AHCCCS registered Provider, and will be authorized by AHCCCS HIE. In all cases, Authorized Users will be given access to the Exchange only after following the procedure set forth Chapter 300 of this document.

There are several levels of security before a User will be granted access to Patient Data through the Exchange. First, the User must be licensed in the State of Arizona and be an AHCCCS provider, or an employee of a HCP who is an AHCCCS registered Provider. Second, the organization that employs the User (or the User, if self-employed) must sign a contract with AHCCCS and promise to follow all of the AMIE privacy and security policies. Third, the User will complete the AMIE training and sign a contract promising to keep the information confidential and only use Patient Data for Treatment. Finally, AHCCCS HIE will assign the clinician a unique user identification and temporary password to allow them to access the Exchange.

Approximately ten AMIE administrative and technical personnel (the AMIE team) involved in the operation of the Exchange may access Patient Data. No other AHCCCS personnel will be able to access the Exchange and the AHCCCS health plans will not be able to access the Exchange.

204: Patient Access to Audit Logs

Upon receiving a formal request from the AHCCCS Privacy Office for a patient audit log, AHCCCS HIE will produce printed copies of the Patient's audit log information reporting all access to the Patient's PHI from the Exchange. AHCCCS HIE will mail the printouts to any address requested by the Patient.

205: Sanitization of EPHI

AHCCCS HIE will sanitize EPHI used for development, testing, quality assurance (QA), and any other non-production use, based upon requirements from the AMIE Development Team.

a. AMIE Security will manually sanitize EPHI by removing all individually identifiable information from all data fields, and certify the process via an email message to the AMIE Director of Development.
b. If only partially sanitized is required, AMIE Security will manually sanitize EPHI by removing at least name, address, and any individually identifiable numbers (SSN, AHCCCS ID, etc.) from data fields, and certify the process via an email message to the AMIE Director of Development.
c. Once received by AHCCCS HIE, the encrypted data file sent from a Data Supplier will be stored in its encrypted format on a file server in the String Test environment. The file containing electronic protected health information (EPHI) remains encrypted on the file server until copied to an MS-SQL 2005 database, also in the String Test environment. Only the AMIE Director of Development, the Lead Developer/Architect, and the Lead DBA will have access to be able to decrypt the data file. At the time of decryption, certain data elements will be modified to scramble personally identifiable information using the Microsoft Visual Studio 2005 Database Edition scrambling tool. The partially scrambled data will be copied into the MS-SQL database. All AMIE developers will have access to the file in the SQL database, and all access is logged in dev/test. The

data may remain in dev/test for about 3-4 months. When AHCCCS HIE no longer needs the data, the data will be deleted from the MS-SQL database and all database logs will be cleared, then the database will be truncated and the space reallocated. This will be documented. From the file server, the encrypted database will also be erased by a tool to securely delete the file.

d. The AMIE Development Team may require and create dummy data to be used in place of EPHI for testing purposes. AMIE Security will review the dummy data and certify the process via an email message to the AMIE Director of Development.

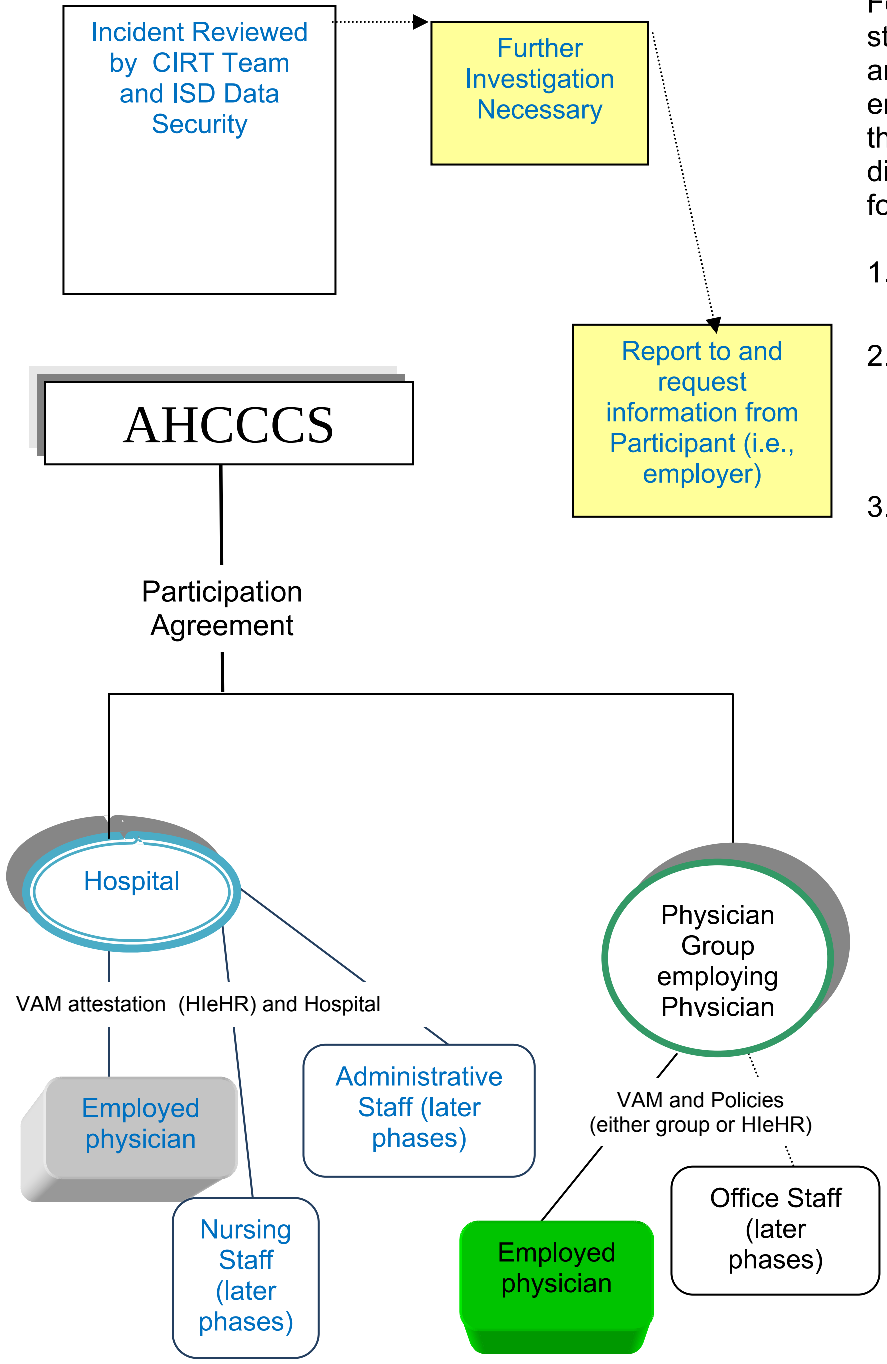
206: Non-Compliance/Sanctions

If AHCCCS HIE receives a report from a Participant regarding any noncompliance with the Participation Agreement or AHCCCS HIE's or Participant's policies for Data access, use or disclosure, AMIE Operations Customer Support team will investigate the report to determine if sanctions or other discipline is required.

The consequences will be dependent upon the type of unauthorized viewing or use. In addition to sanctions that may be imposed by the clinician's employer, potential sanctions include:
(1) permanent termination of user access to the Exchange;
(2) revoking AHCCCS provider status;
(3) reporting to the appropriate Regulatory Board for unprofessional conduct;
(4) reporting to the patient whose PHI was inappropriately viewed, accessed, used or disclosed; or
(5) reporting to the authorities

How will AHCCCS monitor/discipline clinicians who are not employed by a hospital?

These clinicians will be monitored by AHCCCS through the auditing process. If an issue is found, or if there is a report of a potential issue (from a patient, a hospital, another provider, a patient, etc.), AMIE's CIRT Team or Privacy Officer will review the issue as set forth in the auditing policies.


Incident Reviewed by CIRT Team and ISD Data Security
Further Investigation Necessary
Report to and request information from Participant (i.e., employer)
AHCCCS
Participation Agreement
Hospital
VAM attestation (HIeHR) and Hospital
Employed physician
Nursing Staff (later phases)
Administrative Staff (later phases)
Physician Group employing Physician
VAM and Policies (either group or HIeHR)
Employed physician
Office Staff (later phases)

For example, an independent physician who has staff privileges, but is employed by his own P.C. and is the sole owner (the physician is not employed by the hospital). If there is a suspicion that the physician improperly accessed, used or disclosed PHI obtained from the Exchange, the following process would occur:

1. Incident would be reviewed by CIRT Team and ISD Data Security
2. If a determination is made that further investigation is necessary, AHCCCS will report the incident to the Physician Group Employing Physician and request additional information
3. If the employer's response is inadequate or if the employer does not respond, AHCCCS will further investigate and make a determination whether disciplinary action is needed. If so, the discipline can include any of the following:
   a. AHCCCS permanently terminating the physician's access to the Exchange
   b. AHCCCS revoking the physician's provider status
   c. Report the physician to the appropriate licensing board
   d. Report to patients whose records were inappropriately accessed
   e. Report to the authorities
      i. Federal ID theft law (18 USC 1028)
      ii. Federal Computer Fraud law (18 USC 1030
      iii. Arizona ID theft law (ARS 13-2008 et seq
      iv. Arizona Computer tampering law (ARS 13-2316
      v. Arizona unauthorized release of confidential computer information law (ARS 13-2316.02)

Chapter 300 – Participant Management and Authentication

301: AHCCCS HIE Authentication of Participant

AHCCCS HIE will cross-check the Provider ID submitted by a potential Participant in the Exchange with the AHCCCS Provider list. AHCCCS HIE will grant a Participant or its Authorized Users access to the Exchange Viewer upon the Participant signing the Participation Agreement and the Participant's Authorized User signing a Viewer Account Management Form (VAM). (See appendix)

302: User Account Passwords

UserIDs are assigned by the Viewer admin tool based upon the User's name. This UserID is paired with an initial system-generated password for each Authorized User. Passwords require a minimum of eight characters, with three of four complexity items, such as lower and upper case letters, numbers, and special characters. Passwords expire automatically after 60 days and cannot be reused within the following twelve months. After three unsuccessful attempts at login, a User's account will be locked out, requiring the User to phone AMIE Operations Customer Support, identify himself/herself, and request a password reset. Section 306 describes the password reset procedure. All such unsuccessful login attempts are logged. There are no generic, temporary, guest, shared or group UserIDs or passwords allowed to access the Viewer or admin application.

303: Other AMIE Operations Procedures

Technical Support: AMIE Operations Customer Support (CS) will provide on-going technical support such as documentation, a knowledge base, and help desk availability. A CS representative may login to a specific Patient's records to view any errors. Error reports received by CS will be sent to AMIE developers to troubleshoot, then to database admins, and potentially to AHCCCS ISD if required. For troubleshooting purposes, database administrators (DBAs) can find where data may get lost or dumped and correct the issue.

Reports: Two types of reports are available with the Viewer – audit reports and utilization reports. Of these, there are two audit reports and six utilization report available. Each report has multiple configurations to best address any request for audit logs. These reports may be run only by system admins. Utilization reports will be used internally for analysis and may be shared with the Data Provider's executive team by secure email. Audit reports are for internal AMIE use only, unless requested by a Data Provider or Participant. Patient-specific information can only be provided by the Data Provider. Logging is automatically done for all report generation.

Monitoring and Logging: A process is in place to monitor the Viewer and the Viewer Administration tool. Logging occurs at multiple locations, including the gateway and the emulator. AMIE will maintain audit logs documenting Data elements, including 1) When Data was provided to the Exchange; 2) The Participant that provided the Data; 3) When an Authorized User accessed the Data; and 4) The UserID of the Authorized User who accessed the Data and the sponsoring Participant. AMIE will perform manual reviews and verification of audit logs as part of on-going operational monitoring and security practices.

304: Account Lockouts

Inactivity: After a period of fifteen minutes of user inactivity, the Viewer logs out the user, requiring re-authentication to log back in.
Extended Activity: AMIE Operations will log out any User account that remained logged in for 24 hours or more, and will investigate the User's activity during this time to determine if any suspicious activity has taken place.
Inactive Accounts: After a period of 90 days of user inactivity, the AMIE Operations will deactivate a User account.

Chapter 400 – Development

401: Development Procedures

AMIE development and test environment is segregated from the production environment. Development and test systems do not contain copies of non scrubbed production data. Developers do not have on-going access to production systems and data. For special circumstances requiring developer access to production, there are logging and audit trails of this elevated access.

Separation of duties is maintained between system development, database administration, and system administration. Developers write code and create installation scripts. The Configuration team promotes code to other environments, starting from integration to QA, staging, and production. The QA team owns the QA environment. The Operations team owns the production environment.

Development methodology includes an application-level data validation process to ensure the integrity of the data processing. There is no formal, documented methodology for development, testing and quality assurance (QA). Informal procedures exist in the form of an outdated PowerPoint presentation. Documentation of the methodology is in process.

Controls to prevent unauthorized changes to the application source are provided using Microsoft (MS) Team Foundation Server. This source control system is accessible only to authorized developers who chick-out and check-in source code. Full logging is used for source code modification.

Code reviews are performed on the production code using Microsoft .Net Security by architects, not the developers involved in writing the code. Buffer overrun checking and other sanity checks are performed on input data. Secure coding practices, such as those recommended by OWASP, are not being used currently, but are planned for future work

Either Internet Explorer 6.0 or later, or FireFox 2.0 or later are required to be used by the end user. The Viewer application is MS-AJAX enabled. The open-source code MA-SHARE is used. This code is supported by the AMIE Development team, as well as collaboration with the open-source community.

Development staff and database administrators (DBA) rely on AHCCCS ISD to provide relevant and vendor-provided security alerts.

402: Development Methodology

AMIE Development will use Waterfall Methodology and Agile Software Development principles. These software development methodologies are intended to reduce the complexity of software engineering.

Agile Software Development consists of 7 methods:

- Extreme Programming (XP)
- Scrum
- Crystal Orange
- Dynamic Systems Development Method (DSDM)
- Adaptive Software Development (ASD)
- Feature-Driven Development (FDD)
- Pragmatic Programming

The FDD Process features:

- Tiny Building Block for Planning and Tracking
- End to End Implementation lasts about 3 weeks
- FDD facilitates inspection, concurrent development, and tracks progress
- Design and implementation does not effect overall structure of the system built for implementing previous features
- E.g.: RLS, Patient Search, CDX etc

Using these processes, planning is done like a waterfall. Dates are set for the start and completion, with hours to work for each feature set. Planning determines the order of feature sets and completion months for feature sets. Group feature sets are planned for each build.

The steps for an application design by feature:

- First part of short iteration
- Review Requirements and Use Cases
- Identify which Requirements are satisfied with this feature
- Determine which component owners are needed for this feature (E.g.: HIE, Viewer)
- Have the Business Expert tell everything about the feature
- Design the feature (Use UML if needed)
- Design the Inspection (Test)

The steps for an application build by feature:

- Create a Branch in the Source Control for the Feature team to work on.
- Code Feature Set.
- Test Feature Set.
- Code Inspection.
- Promote to build (Integration).
- Deploy

HIeHR Development uses Extended Agile Methodology with Microsoft Solutions Framework and Feature Driven Development. This uses the Agile software development process framework and extends the Agile approach to all phases of the development life cycles. It also brings in the advantages of the FDD Agile methodology, such as MSF 4.1 Artifacts templates.

403: AMIE Software Development Roles

The following roles have been defined for the AMIE Development Team:

- Application Development Manager
- Project Manager
- System Architect
- Security / Infrastructure / Performance Architect
- HIE Application Architect
- HIE Application Developer
- Viewer Application Architect
- Web Designer
- Web Application Developer
- Development Manager
- Sr. Developer
- System Analyst
- Domain Experts
- Data Developer (DBA)
- Component Integration Engineer (Test Engineer)
- Configuration Management, Deployment and Production System Management
- On Boarding System Analyst / PM
- On Boarding Sr. Developer
- On Boarding DBA

The Application Development Manager (ADM) is responsible for leading the day-to-day development activities including the resolution of resource conflicts, and development methodology. The ADM works with the Architects in finalizing the feature sets, the Architects and the Project Managers to prioritize and plans the features for each build, and with Project Manager in the resource allocation and creating work break down structures. The ADM is also responsible for the overall timely release of the each build software as per the plan, for the overall timely release of the artifacts associated with each build, for the hardware, software and tools for the Application Development and release process, and for organizing the daily scrum meetings and makes sure that the issues are addressed and assigned to the responsible parties. The ADM participates in the change management group for Change Requests and in the QA group for bug fixes.

The Project Manager is the Administrative lead and is responsible for managing budgets, fighting for and managing resources including people, equipment and space.

The System Architect (SA) is responsible for interfacing with user(s), sponsor(s), and other stakeholders to determine needs. The SA generates the highest level of system requirements and ensures these requirements are consistent, complete, correct, and operationally defined. The SA also performs cost-benefit analyses to determine trade offs between manual, software, or hardware functions. The SA develops partitioning to allocate requirements into discrete partitions, partitions large systems into (successive layers of) subsystems and components each of which can be handled by a single engineer or team of engineers or subordinate architect. The SA ensures a maximally robust architecture and generates acceptance test requirements with designers, test engineers, and users, as well as generating sketches, models, early user guide, and prototypes to keep the user and the engineers up to date and in agreement on the system to be provided.

The Security / Infrastructure / Performance Architect designs and specifies requirements and is responsible for implementing the infrastructure of the system, includes data partner interfaces. The Security Architect is the chief security designer, specifies requirements for, and is responsible for creating the security model of the system, including data partner security topics. Security must comply with all pertinent health information security policies including HIPAA regulations. The Security Architect is responsible for designing the solution to support the required volume of transactions meeting response requirements, and specifies criteria for performance measures.

The HIE Application Architect is responsible for overall HIE design, for running HIE design sessions, working with the users and make sure that the design is in synch with the requirements, and defines, develops and documents technical specifications supporting requirements from users.

The HIE Application Developer perform development tasks, programming per the HIE design specifications, and development of HIE gateways and frameworks. The HIE Application Developer also participates in high-level requirements analysis and design, aids the team in low-level analysis and design, participates in development of the features, creates the Unit Test scripts and provides the Unit test results, and creates the release documentation for the feature releases. The HIE Application Developer works with development manger in prioritizing the change management and bug fixes, and creates Deployment/Setup Projects.

The Viewer Application Architect is responsible for overall Viewer design, for running Viewer design sessions, working with the users and make sure that the design is in synch with the requirements, defining, developing and documenting technical specifications supporting requirements from users.

The Web Designer defines and maintains web User Interface (UI) standards, designs the look and feel of web interface, defines the screen flow and organization of the site/information architecture.

The Web Application Developer performs development tasks, and programming per the design specifications. The Web Application Developer develops the Viewer and HIE Admin tools, participates in high-level requirements analysis and design, aids the team in low-level analysis and design, participates in development of the features, creates the Unit Test scripts and provides the Unit test results, creates the release documentation for the feature releases, works with development manger in prioritizing the change management and bug fixes, and creates Deployment/Setup Projects.

The System Analyst works with HIE Application Architect and Viewer Application Architect in the feature design, works with Business Analyst and the Application Architect in analyzing the requirements, is responsible for the design documentation, and provides the domain expertise to the development team on the messaging standards.

The Data Developer (DBA) develops stored procedures, creating data views, SQL programming, database design and development and has responsibility for creating the relational data structures. The DBA develops data modeling, data dictionary documentation and maintenance of documentation, and develops data maintenance processes.

The Component Integration Engineer (Test Engineer) develops test and integration scripts, utilities, tools, test data, performs component integration and testing, designs/defines and manages build and release processes. The Test Engineer is responsible for configuration management, deployment and production system management, manages building and release of application versions, and maintains, documents and tracks contents of each release, including enhancements and bug fixes. The Test Engineer also builds setup/installation scripts to deploy software into testing, staging and production environments, determines and controls when code is ready to be promoted, based on input from user, development and QA representatives, performs production system upgrades, and monitors system performance and service quality.

The On-Boarding System Analyst / PM manage the technical on-boarding process, including coordination with data partners, and data partner interface testing. This PM analyzes, defines, implements and documents data translation, data mapping, technical interface, web service and security requirements.

The On-Boarding Senior Developer performs development tasks for on-boarding process, programming per the design specifications, and does unit testing.

The On-Boarding DBA develops stored procedures, creates data views, does SQL programming, database design and development and has responsibility for creating the relational data structures. The On-Boarding DBA also develops data modeling, data dictionary documentation and maintenance of documentation, and the data maintenance processes.

404: OWASP compliant coding practices

Open Web Application Security Project (OWASP) is a worldwide free and open community focused on improving the security of web application software. OWASP has identified ten critical web application security flaws and urge all web developers to adopt strategies to ensure their web applications do not contain these flaws. This section will briefly explain the top ten security flaws identified by OWASP and how the AMIE application is coded to avoid the flaws.

#1 - Cross Site Scripting

Cross site scripting, better known as XSS, is in fact a subset of HTML injection. XSS is the most prevalent and pernicious web application security issue. XSS flaws occur whenever an application takes data that originated from a user and sends it to a web browser without first validating or encoding that content.

XSS allows attackers to execute script in the victim's browser, which can hijack user sessions, deface web sites, insert hostile content, conduct phishing attacks, and take over the user's browser using scripting malware. The malicious script is usually JavaScript, but any scripting language supported by the victim's browser is a potential target for this attack.

Protection

The best protection for XSS is a combination of "white list" validation of all incoming data and appropriate encoding of all output data. Validation allows the detection of attacks, and encoding prevents any successful script injection from running in the browser.

Preventing XSS across an entire application requires a consistent architectural approach:

- Input Validation

  The AMIE validates all incoming fields.
- Output Encoding

  All fields that are user originated data that needs to be displayed out to the screen are output encoded.
- Specify Output Encoding

  The AMIE has a default output encoding specified as UTF-8.
- Do not use "blacklist" validation

  The AMIE does not use "blacklist" validation.
- Watch out for canonicalization

  The AMIE does not canonicalize any use input.

Additional Actions

The AMIE system will implement the Microsoft Anti-XSS library for ASP.Net.

#2 - Injection Flaws

Injection flaws, particularly SQL injection, are common in web applications. There are many types of injections: SQL, LDAP, XPath, XSLT, HTML, XML, OS command injection and many more.

Injection occurs when user-supplied data is sent to an interpreter as part of a command or query. Attackers trick the interpreter into executing unintended commands via supplying specially crafted data. Injection flaws allow attackers to create, read, update, or delete any arbitrary data available to the application. In the worst case scenario, these flaws allow an attacker to completely compromise the application and the underlying systems, even bypassing deeply nested fire-walled environments.

Protection

Avoid the use of interpreters when possible. If you must invoke an interpreter, the key method to avoid injections is the use of safe APIs, such as strongly typed parameterized queries and object relational mapping (ORM) libraries. These interfaces handle all data escaping, or do not require escaping. Note that while safe interfaces solve the problem, validation is still recommended in order to detect attacks.

Using interpreters is dangerous, so it's worth it to take extra care, such as the following:

- Input Validation

  The AMIE validates all incoming fields.
- Use strongly typed parameterized query APIs

  All database communication is accomplished through the use of strongly typed stored procedure parameters
- Enforce least privilege

  System user is restricted to execute permissions on stored procedures only.
- Avoid detailed error messages

  Production systems will have a custom error page that will show only that an error occurred and no details of the error will be presented to the browser. The error will be logged by the system for later troubleshooting.
- Avoid stored procedure injection

  All stored procedures will be written to prevent injection.
- Do not use dynamic query interfaces

  The AMIE does not use any dynamic query interfaces.
- Do not use simple escaping functions

  Only the approved javascript and C#.Net escaping functions are used.

Additional Actions

Custom error pages will be designed and a code reviews will be performed to ensure existing code is injection proof.

#3 - Malicious File Execution
Malicious file execution vulnerabilities are found in many applications. Developers will often directly use or concatenate potentially hostile input with file or stream functions, or improperly trust input files. On many platforms, frameworks allow the use of external object references, such as URLs or file system references. When the data is insufficiently checked, this can lead to arbitrary remote and hostile content being included, processed or invoked by the web server.
This allows attackers to perform:

- Remote code execution
- Remote root kit installation and complete system compromise
- On Windows, internal system compromise may be possible through the use of PHP's SMB file wrappers

This attack is particularly prevalent on PHP, and extreme care must be taken with any stream or file function to ensure that user supplied input does not influence file names.

Protection
Preventing remote file include flaws takes some careful planning at the architectural and design phases, through to thorough testing. In general, a well-written application will not use user-supplied input in any filename for any server-based resource (such as images, XML and XSL transform documents, or script inclusions), and will have firewall rules in place preventing new outbound connections to the Internet or internally back to any other server. However, many legacy applications will continue to have a need to accept user supplied input.

Among the most important considerations are:

- Use an indirect object reference map

  The AMIE does not expose any private object references.
- Use explicit taint checking mechanisms

  The AMIE does not use user-supplied input in any filename.
- Strongly validate user input

  The AMIE validates all incoming fields.
- Check any user supplied files or filenames

  The AMIE only accepts user supplied files in one location and the file is strongly checked for correct content type and content. The file contents are encoded and validated before present back to the user.

Additional Actions
No additional actions required to secure against malicious file execution.

#4 - Insecure Direct Object Reference
A direct object reference occurs when a developer exposes a reference to an internal implementation object, such as a file, directory, database record, or key, as a URL or form parameter. An attacker can manipulate direct object references to access other objects without authorization, unless an access control check is in place.

Protection

The best protection is to avoid exposing direct object references to users by using an index, indirect reference map, or other indirect method that is easy to validate. If a direct object reference must be used, ensure that the user is authorized before using it.

Establishing a standard way of referring to application objects is important:

- Avoid exposing your private object references to users

  The AMIE does not expose any private object references.
- Validate any private object references

  The AMIE does not expose any private object references.
- Verify authorization to all referenced objects

  The AMIE does not expose any private object references.

Additional Actions

No additional actions required to secure against direct object references.

#5 - Cross Site Request Forgery
Cross site request forgery is not a new attack, but is simple and devastating. A CSRF attack forces a logged-on victim's browser to send a request to a vulnerable web application, which then performs the chosen action on behalf of the victim.

This vulnerability is extremely widespread, as any web application that

- Has no authorization checks for vulnerable actions
- Will process an action if a default login is able to be given in the request (e.g. http://www.example.com/admin/doSomething.ctl?username=admin&passwd=admin)
- Authorizes requests based only on credentials that are automatically submitted such as the session cookie if currently logged into the application, or "Remember me" functionality if not logged into the application, or a Kerberos token if part of an Intranet participating in integrated logon with Active Directory is at risk. Unfortunately, today, most web applications rely solely on automatically submitted credentials such as session cookies, basic authentication credentials, source IP addresses, SSL certificates, or Windows domain credentials.

Protection

- For sensitive data or value transactions, re-authenticate or use transaction signing

  The AMIE system will use Phone factor to ensure genuine authentication.
- Do not use GET requests (URLs) for sensitive data or to perform value transactions

  The AMIE does not use GET requests for sensitive data.
- POST alone is insufficient a protection

  ASP.NET has built in page validation keys to ensure each post is valid.
- Set a ViewStateUserKey.

  The AMIE implements the ViewStateUserKey to ensure that the ViewState is encrypted differently for each user session.

Additional Action

No additional actions required to secure against cross site request forgery.

#6 - Information Leakage and Improper Error Handling

Applications can unintentionally leak information about their configuration, internal workings, or violate privacy through a variety of application problems. Applications can also leak internal state via how long they take to process certain operations or via different responses to differing inputs, such as displaying the same error text with different error numbers. Web applications will often leak information about their internal state through detailed or debug error messages. Often, this information can be leveraged to launch or even automate more powerful attacks.

Protection

Developers should try to make their application generate errors. Applications that have not been tested in this way will almost certainly generate unexpected error output. Applications should also include a standard exception handling architecture to prevent unwanted information from leaking to attackers.

Preventing information leakage requires discipline. The following practices have proven effective:

- Ensure that the entire software development team shares a common approach to exception handling.

  A common approach is share among the development team.
- Disable or limit detailed error handling.

  Production systems will have a custom error page that will show only that an error occurred and no details of the error will be presented to the browser. The error will be logged by the system for later troubleshooting.

- Ensure that secure paths that have multiple outcomes return similar or identical error messages

  Production systems will have a custom error page that will show only that an error occurred and no details of the error will be presented to the browser. The error will be logged by the system for later troubleshooting.
- Create Custom Error Pages

  Production systems will have a custom error page that will show only that an error occurred and no details of the error will be presented to the browser. The error will be logged by the system for later troubleshooting.

Additional Actions

Custom error pages will be designed and a code reviews will be performed to ensure existing code is injection proof.

#7 - Broken Authentication and Session Management

Proper authentication and session management is critical to web application security. Flaws in this area most frequently involve the failure to protect credentials and session tokens through their lifecycle. These flaws can lead to the hijacking of user or administrative accounts, undermine authorization and accountability controls, and cause privacy violations.

Protection

Authentication relies on secure communication and credential storage. First ensure that SSL is the only option for all authenticated parts of the application and that all credentials are stored in hashed or encrypted form.

Preventing authentication flaws takes careful planning. Among the most important considerations are:

- Only use the inbuilt session management mechanism

  The AMIE uses only the Microsoft Database Session management mechanism.
- Do not accept new, preset or invalid session identifiers

  The AMIE creates a new session as soon as a user is authenticated, so no preset or invalid session identifiers can be used.
- Limit or rid your code of custom cookies for authentication or session management

  The AMIE does not use custom cookies or session management routines for authentication.
- Use a single authentication mechanism

  The AMIE uses only Microsoft Forms Authentication.
- Do not allow the login process to start from an unencrypted page

- Generate a new session upon successful authentication

  The AMIE creates a new session as soon as a user is authenticated.
- Ensure that every page has a logout link

  The master page used for all child pages contains a logout link.
- Use a timeout period

  The AMIE has a preset 15 minute timeout period.
- Use only strong ancillary authentication functions

  The AMIE utilizes security question and answer and phone factor as ancillary authentication functions.
- Do not expose any session identifiers or any portion of valid credentials in URLs or logs.

  Session identifiers and credentials are not written to logs or URLs.
- Check the old password when the user changes to a new password

  To change a password, user must supply their previous password.
- Do not rely upon spoofable credentials as the sole form of authentication

  The AMIE does not use any spoofable credentials for authentication.
- Be careful of sending secrets to registered e-mail addresses

  Currently, no secrets are sent via email.

Additional Actions

No additional actions required to secure against broken authentication and session management.

#8 - Insecure Cryptographic Storage

Protecting sensitive data with cryptography has become a key part of most web applications. Simply failing to encrypt sensitive data is very widespread. Applications that do encrypt frequently contain poorly designed cryptography, either using inappropriate ciphers or making serious mistakes using strong ciphers. These flaws can lead to disclosure of sensitive data and compliance violations.

Protection

The most important aspect is to ensure that everything that should be encrypted is actually encrypted. Then you must ensure that the cryptography is implemented properly. As there are so many ways of using cryptography improperly, the following recommendations should be taken as part of your testing regime to help ensure secure cryptographic materials handling:

- Do not create cryptographic algorithms

  The AMIE system uses triple DES encryption for symmetric encryption.
- Do not use weak algorithms

  The AMIE system uses SHA1 for hashing.
- Generate keys offline and store private keys with extreme care

  Private keys are stored in the config file.
- Ensure that infrastructure credentials such as database credentials or MQ queue access details are properly secured.

Credentials are stored in config files.

- Ensure that encrypted data stored on disk is not easy to decrypt

No encrypted data is stored on disk.

Additional Actions

The AMIE system will change hashing encryption algorithm to the stronger SHA512. In addition, alternatives to storing credentials in config files will be explored.

#9 - Insecure Communications

Applications frequently fail to encrypt network traffic when it is necessary to protect sensitive communications. Encryption (usually SSL) must be used for all authenticated connections, especially Internet-accessible web pages, but backend connections as well. Otherwise, the application will expose an authentication or session token. In addition, encryption should be used whenever sensitive data, such as credit card or health information is transmitted. Applications that fall back or can be forced out of an encrypting mode can be abused by attackers.

Protection

The most important protection is to use SSL on any authenticated connection or whenever sensitive data is being transmitted. There are a number of details involved with configuring SSL for web applications properly, so understanding and analyzing your environment is important. For example, IE 7.0 provides a green bar for high trust SSL certificates, but this is not a suitable control to prove safe use of cryptography alone.

- Use SSL for all connections that are authenticated or transmitting sensitive or value data

All public communications will utilize SSL.

- Ensure internal communications are secure

Internal communications will utilize SSL.

Additional Actions

No additional actions required to secure against insecure communications.

#10 - Failure to Restrict URL Access

Frequently, the only protection for a URL is that links to that page are not presented to unauthorized users. However, a motivated, skilled, or just plain lucky attacker may be able to find and access these pages, invoke functions, and view data. Security by obscurity is not sufficient to protect sensitive functions and data in an application. Access control checks must be performed before a request to a sensitive function is granted, which ensures that the user is authorized to access that function.

Protection

Taking the time to plan authorization by creating a matrix to map the roles and functions of the application is a key step in achieving protection against unrestricted URL access. Web applications must enforce access control on every URL and business function. It is not sufficient to put access control into the presentation layer and leave the business logic unprotected. It is also not sufficient to check once during the process to ensure the user is authorized, and then not check again on subsequent steps. Otherwise, an attacker can simply skip the step where authorization is checked, and forge the parameter values necessary to continue on at the next step.

Enabling URL access control takes some careful planning. Among the most important considerations are:

- Ensure the access control matrix is part of the business, architecture, and design of the application
- Ensure that all URLs and business functions are protected by an effective access control mechanism

  Forms authentication is used through out the AMIE system.
- Pay close attention to include/library files

  The AMIE does not use include files and library files are in compiled format.
- Do not assume that users will be unaware of special or hidden URLs or APIs

  All pages are secured and requests are validated against ACL.
- Block access to all file types that your application should never serve

  IIS will block access to unauthorized file requests.
- Keep up to date with virus protection and patches

  Virus protection will be automatic.

Additional Actions

No additional actions required to secure against URL access.

Chapter 500 – Auditing and Compliance

501: Audit Log Retention

AMIE audit logs documenting Authorized User access to the Viewer will be retained as required by HIPAA. AHCCCS is the custodian of these logs.

502: AMIE Logging

Microsoft (MS) SQL 2005 Server Standard is used to store most log files, which reside at the primary and secondary data centers for the AMIE. The basic architecture consists of an emulator (server) which will store EPHI and convert the data from the provider's information system to the HL7 CCD format required by the Exchange. Also required is a gateway (server) which will allow secure access of the data using the Exchange and Viewer. In most instances the emulator and gateway will be located at the Data Provider's data center.

Three types of logging in the Exchange Viewer process will be maintained: audit logging, transaction logging, and debug logging.

<u>Audit Logging</u>
Audit logging needs to be active at all times. Minimal information is gathered and stored in these logs, including:

- Viewer Users and Viewer Administrators user ID's, and their uses of the Viewer
- The name of the User requesting information using the Viewer
- The patient name, record type, record ID (a pointer to record), and source (Data Provider) involved in a request.

The results of the request are stored unencrypted in the SQL logging database in the AMIE data centers. This type of logging is not done on remote gateways and emulators. The results are viewable using the Viewer Admin Tools, and Utilization and Audit reports.

<u>Transaction Logging</u>
Transaction logging has two levels when active: minimal and verbose. Minimal transaction logging is active at all times, and verbose transaction logging is only being activated on an as-needed basis, to troubleshoot errors. The information gathered and stored in these logs, includes:

- System to system logging
- Display requests from a source device and the destination device, from the perspective of the Viewer

The contents of the verbose transaction logging include EPHI as requested, or returned as results, and are in XML format.

Logging results are stored encrypted in the SQL logging database in the AMIE data centers, as well as at each gateway. Transaction logging can be turned off completely, but no historical data would be available to troubleshoot errors. Turning off transaction logging completely is not recommended. The results of this logging can be viewed and queried in the Viewer Admin tool.

Logging is also performed and stored encrypted on remote gateways and emulators, until the information is copied to the SQL logging database in the AMIE data centers, and then deleted off the devices. Logging performed and stored on remote gateways and emulators can be copied to the SQL logging database and deleted off gateway every 15 minutes are nominal.

It is estimated that up to 95% of most Exchange Viewer related problems will be resolved with either audit logs, minimal transaction logs, or verbose transaction logging turned on.

Debug Logging
Debug logging records every single step of a transaction process, including data parsing, data validation, etc. Debug logging is verbose and will not be active unless needed for the unresolved 5% of issues. Debug logging:

- Includes all EPHI in a transaction
- Can be turned on and off as needed in production

Debug logging results are stored unencrypted in the SQL logging database or at a gateway or emulator until the data is copied to the SQL logging database and deleted off gateway or emulator. Debug logging can be turned on at remote gateway or emulator and the verbose output can be directed to a separate secure remote archive location, then the output can be deleted when no longer needed from the remote location. The results of debug logging are similar to transaction logging results, but with much more detail and no message or record detail.

No individual transaction logging will be available by any item, such as patient, user, etc.

503: Risk Assessment of Non-Compliance

AMIE Security will perform a risk assessment of any non-compliance on the part of AMIE staff or any Participant. An action plan for remediation will be produced and will be tracked weekly by the AMIE Security team. The Risk Assessment will include a review of potential threats and vulnerabilities, and associated risks to patient data. Risks will be ranked based upon potential impact and mitigating controls will be recommended to lower or eliminate any risk.

Chapter 600 – Security

601: AMIE Security Architecture

The AMIE architecture consists of duplicated physical configurations located at two different state government data centers in the central Phoenix area. The main data center is at AHCCCS, 801 E. Jefferson St. The second site is at the AZ Department of Administration, 100 N. 15th Avenue, which serves as the business continuity planning (BCP) site.

The architecture at each site consists of routers, firewalls, load balancers, web/application servers, database servers, and SAN storage. The F5 BigIP load balancers and Cisco PIX firewalls are in an active/active mode. The databases are clustered in an active/passive failover mode using Microsoft (MS) SQL 2005 Server Standard. The web and application servers are virtual servers on the same physical hardware. All servers use Microsoft Server 2003 release 2.

The architecture at the data provider locations may be different, based upon the system capabilities each provider has. The basis architecture consists of an emulator (server) which will store EPHI and convert the data from the provider's information system to the HL7 CCD format required by the Exchange. Also required is a gateway (server) which will allow secure access of the data using the Exchange and Viewer. In most instances the emulator and gateway will be located at the Data Provider's data center; however, there are other alternatives. The use of a Cisco AON device at the data provider's location will be explored and may be used. The AON would function similar to emulator and gateway and replace both servers.

Software products supporting the system include MS Windows 2003 Server, MS SQL Server 2005, MS Framework v2, BizTalk, an HL7 accelerator for BizTalk, anti-virus and backup software.

Protocols and services used to communicate between the application components include industry standard TCP/IP on the internal network, and encyrpted via HTTPS over the Internet. C# .NET Java, and MS-AJAX programming languages are used for the Viewer. All web sessions with confidential data are protected using SSL v2.0 minimum with 128 bit strength.

Authentication and access control information (user IDs and passwords validating user credentials) for all Authorized Users are located on the SQL Servers and are encrypted. The database cannot be queried directly, only the application can run storage procedures. Initial logon to the database is performed by an application-level authentication where the UserID and password are stored for the application. There are only two levels of users, an Authorized User and a System Administrator (admin). Controls are in place for administrative access.

The session management methodology uses persistent session cookies, and when a session dies the cookie dies as well. Session management controls are used to end a session and log off the user after a period of session inactivity. Time limit thresholds are configurable by admins on a system-wide basis.

Stress and load testing on the application and architecture are performed in the staging environment. Digital certificates are used to protect each web server, and are purchased and supplied by AHCCCS ISD.

Patch management procedures on servers, and timeframes to mitigate critical/high vulnerabilities are handled by ISD. This includes procedures for confirming systems are patched and up-to-date. On-site procedures for Data Provider locations will be established, and may possibly be done by Cisco or a reseller if the AON device is used.

An independent security provider will be retained to conduct periodic security audits, vulnerability scanning, and penetration testing on the network, server infrastructure, and website. Service Level Agreements (SLA's) are in place with Dell for server hardware. SLA's have been negotiated with Data Suppliers. AHCCCS ISD does not support SLA's.

More network bandwidth is available than needed. Steps will be taken to ensure acceptable response times for users by testing at a much higher rate than planned usage. Administration and monitoring of the production servers is the responsibility of ISD. An escalation procedure in case of a production failure is being defined. The capacity of the server tier can be increased both vertically and horizonally, with larger or more servers as is necessary. Remote access to the production servers is available via VPN.

602: Computer Incident Response Team (CIRT)

The Computer Incident Response Team (CIRT) is composed of AMIE staff and ISD Data Security personnel whose mission is to address security vulnerabilities and incidents on AMIE information resources.
The objective of the CIRT is to perform, coordinate and support responses to security incidents. Specific responsibilities of the CIRT or its delegates include:

1. Monitoring vendor and government vulnerability alerts, web sites and mailing lists.
2. Notifying resource administrators of vulnerabilities as they are identified.
3. Identifying, developing, and/or recommending security assessment tools.
4. Coordinating incident response activities both internally and externally.

603: Security Risk Management

A risk analysis of AMIE and the Exchange processes and technologies using industry standard processes for qualitative risk will be performed initially and periodically to assess potential risks and vulnerabilities to EPHI, and security measures and safeguards to reduce risk through compensating controls and mitigation.

604: Action Plan Tracking

AMIE Security will track Action Plans that are created as a result of risk assessments of AMIE and Exchange processes and technologies. Action Plans that are created as a result of a risk review of Participant noncompliance will also be tracked. All Action Plans will be tracked on a weekly basis until completed.

605: Incident Handling

Incident Handling – Initial Triage by Operations

Ways in which AMIE Operations may be notified of/discover an event
Call or email from:

1. Patient
2. User
3. ISD
4. OALS
5. AMIE Team

Types of events requiring attention:

1. Privacy violation
2. EPHI access for other than Treatment
3. Password sharing
4. Suspected user identity theft (username/password stealing)
5. User lockout for unknown reason
6. Intrusion attempt on access website or data
7. Obvious defacement of website
8. Log anomalies

Events requiring notification of CIRT:

1. Suspected user identity theft (username/password stealing)
2. User lockout for unknown reason
3. Intrusion attempt on access website or data
4. Obvious defacement of website
5. Log anomalies

Incident Handling

There are six steps in the incident handling process:

1. Preparation
2. Identification
3. Containment
4. Eradication
5. Recovery
6. Lessons learned

Preparation

The first step of preparation requires the naming of the Incident Response Team, and developing a plan of what to do when an incident occurs (See **Appendix B**). The team should identify specific contacts in senior management and the legal department, as well as to evaluate whether to include law enforcement in advance, should an attack be severe. The team should be aware of the disaster recovery plan, including checklists, procedures, and emergency communication.

Identification

An event it is not an incident until someone identifies it as such. The Team must be called together to assess the situation and take the proper steps or give the all clear. Even when an event *could be* an incident, the Team members must be willing to alert the rest of the Team early enough to act, but not too early and sound a false alarm and needlessly panic the staff.

AMIE Operations staff will begin to identify incidents by tracking trouble calls. Many incidents affect many people, so multiple calls about the same subject from different people can point to a possible incident.

Other signs of an incident can include alerts by an Intrusion Detection or Prevention System (IDS), unexplained entries in a log file, failed or unexplained events, system reboots or crashes, poor system or network performance. Once an event is identified, a primary response team should be assigned to determine whether an event is an incident. For legal purposes the team must maintain a provable chain of custody of all evidence in case the matter goes to court.

Containment

An incident handler should not worsen the situation. The area and/or equipment should be made secure and a copy of all evidence should be made as well. In some instances, the system should be pulled off the network if at all possible, to prevent a spread of the problem. Passwords should also be changed immediately, if the incident involved unauthorized access, and password compromise was possible. Refer to incident handling best practices for more information.

Eradication

The problem must be fixed before putting the system back on line. This does not mean simply reinstalling the operating system. The incident handler must determine cause and symptom of the attack in order to improve defenses in the future. If the vulnerability was previously unknown, it should be recorded and analyzed sufficiently to prevent a reoccurrence. Refer to incident handling best practices for more information.

Recovery
The incident Team or handler must determine if there are the corrupted and compromised files and where these files are located. Each potentially compromised machine must be tested and the regular user should give his/her approval that the machine is functioning correctly before putting it back online. Each of these systems should be monitored for possible re-infection. The entire Team should hold a follow-up meeting after every incident to review how the process worked. Refer to incident handling best practices for more information.

Lessons Learned
The Team should make appropriate recommendations to management on how to better prevent future similar incidents, if possible. It is very important for a report to be written within 48 hours that shows a factual complete consensus amongst the Team in case the matter goes to court. The report should focus on improvements that can be made, suggested changes to the Incident Handling plan, processes, and procedures.

Types of Incidents
A computer security incident is described as a violation or imminent threat of violation of computer security policies, acceptable use policies, or standard security practices. Examples of an incident are:

Denial of Service
- An attacker sends specially crafted message to a web-sever causing it to crash
- An attacker directs hundreds of compromised workstations to send as many request as possible to the organizations network

Malicious Code
- A worm uses open files shares to quickly infect several hundred workstations within an organization
- An organization receives a warning from an anti-virus vendor that a new virus is spreading rapidly via email throughout the Internet. The virus takes advantage of a vulnerability that is present on many of the organization’s host.

Unauthorized Access
- An attacker runs a tool to gain access to a server’s password file.
- A perpetrator obtains unauthorized administrator-level access to a system and then threatens the victim that the detail of the break in will be released to the press if the organizations does not pay a designated sum on money

Inappropriate Usage
- A user provides illegal copies of software to others through peer-to-peer file sharing services.
- A person threatens another user through e-mail

Severity Levels
Incidents occurring within AMIE are given different levels of severity depending on the extent of the incident.

- **Low -** Only a small number of systems are affected, such as with a virus that has been known and that the current anti-virus protect is handling. A small number of machines are being scanned from an outside source, but no attack or penetration has taken place.
- **Medium -** A significant numbers of system are being scanned or penetration or denial of service attacks attempted with no affect on operations; such many occurrences of a know virus, but they are being handled by anti-virus software. A few isolated instances of a new computer virus not handle by anti-virus software. Suspected access of sensitive data has occurred or is likely to occur.
- **High -** Penetration has occurred or Denial of Service attacks have been detected with significant impact on operations. Known unauthorized access to sensitive data has occurred or is likely to occur.

Goal of the Incident Response Team
The goal of the incident response team is to respond to all computer security incidents in a quick and effective manner in order to protect the infrastructure and data that resides on the network and at rest and to document and report those incidents.

Performance Measures
In order to effectively operate a CIRT it is important to gain an understanding of what incidents are occurring within the organization and how those incidents were handled. The development of performance measurements, or lessons learned is a critical aspect of this. Procedures and guidelines will be followed after each incident.

Reporting an Incident
AMIE Operations will maintain a 24x7 phone number where incidents can be reported along with an e-mail address that employees can use to submit questions.

All users are encouraged to report all suspected security incidents to AMIE Operations as soon as possible. The user should call AMIE Operations and report the incident. The Operations Staff will gather pertinent information about the incident and about the person reporting the incident. Once the incident has been evaluated it now becomes the responsibility of the CIRT to report significant incidents to the appropriate agency management.

Responding to a Computer Security Incident
When responding to a computer incident it is important that acute information is gathering at the beginning of the incident. When a call is received the person receiving the call should fill out the Incident Report Form. This will provide security personnel with certain information that will assist them in responding to the incident. The basic form will contain information about the user such as name, location, phone number, e-mail address. The form will also request information about the incident, such as the type of incident, whether its malicious code, unauthorized access, inappropriate use, or

undefined. The user will be asked whether or not the system affected is currently up or is the system down. Please see **Appendix A** for an Incident Report Form example.

Once the information has been gathered the person receiving the call will contact AMIE Security or ISD Data Security to handle the incident described in the report.

606: Emulator Backup Strategy

The production emulator will store records forwarded by the Data Supplier in an MS-SQL database on its local hard drives. The total amount of storage initially available on these servers will be less than 600 gigabytes total. The information in this database will be backed up daily to prevent loss in the event of a hardware failure. It is anticipated that the amount of data to be backed up initially will be very small, but the volume will grow over time.

The database on the production emulator will at minimum have a full backup done once a week and incremental backups every day. The Data Supplier will ensure that at least one copy of the backup data is kept offsite for disaster recovery.

607: Time Syncing

Two US Naval Observatory time keeping servers have been selected to update time on all AMIE servers and have been added to the standard server image build.

The servers are named "tick.usno.navy.mil" and "tock.usno.navy.mil"

The primary will be named "tick" and the secondary server (backup) will be "tock".

Chapter 700 – AMIE Technology Standards

701: Messaging Standards

AMIE will use the following messaging Standards for data exchange:
HL7, DICOM, NCPDP, ASCX12, CDISC, IEEE 1073, ELINCS, HL7 V3 RIM (Conceptual)

702: Terminology Standards

AMIE will use the following terminology Standards:
ICD 9, SNOMED, UMLS, LOINC, CPT

703: Document Standards

AMIE will use the following document Standards:
CCR, CDA

704: Technology Organization Standards

AMIE will follow technology Standards set by these organizations:
Certification Commission for HIT (CCHIT)
Medicaid Information Technology Architecture (MITA)

705: Other Standards

AMIE will follow these other Standards:

Security and Privacy standards and best practices (including HIPAA)
Application and Architectural Standards and best practices
Standards common to web technologies and internet connectivity

AMIE will follow Health Information Technology Standards Panel (HITSP) Interoperability Specifications for Electronic Health Records for Web Browser Authentication, Patient ID cross-matching, time synchronization, etc. C4420, TP22, T16.

Chapter 800 – Policy Creation and Maintenance Procedure

801: Policy Creation and Maintenance

Purpose

To provide a standard method by which policies relating to the AMIE are introduced or modified.

Scope

This procedure applies to all policies relating to the administration and operation of AMIE; except that any policy that: (1) affects the operations of another division of the AHCCCS Administration or requires the commitment of resources of another division of the AHCCCS Administration, or (2) requires funding outside of the established AMIE budget, which must be approved by AHCCCS Executive Management using AHCCCS Administrative Policy 102.

Definitions

"Draft policy" – a proposed new policy or proposed modification to an existing policy.

"AMIE Grant" – the Medicaid Transformation Grant submitted by the AHCCCS Administration to and approved by the Center for Medicare and Medicaid Services as the "Medicaid Health Information Exchange and Project."

Procedure

The following procedure must be employed and documented when introducing a new policy or modifying an existing policy.

802: Draft Policy Development

The author is responsible for:

a. Ensuring that the Draft Policy is consistent with the terms of the AMIE Grant, including the Project Goals and Outcomes described in the AMIE Grant, and existing AHCCCS and AMIE policies.
b. Identifying any issues raised by the Draft Policy and recommendations for addressing those issues.
c. Discussing the Draft Policy with any other staff affected by the policy. As appropriate to the nature of the policy, this may include managers of the AMIE with expertise in privacy, security, information systems, and partner/provider relations.
d. Obtaining legal review when appropriate.

e. Identifying whether comments or input from external stakeholders regarding the draft policy is appropriate.
f. Identifying any financial impact or estimated costs of the draft policy to the Project, the Administration, or other stakeholders.
g. Identify the need for and method of disseminating the draft policy if approved.

803: Draft Policy Format and Content

When preparing a Draft Policy:

a. All pages must be marked "draft" until the Draft Policy is approved and is incorporated into the appropriate policy manual.
b. Formatting should be the same as is used in the AMIE Policy Manual.
c. When drafting a revision to an existing policy, all policy language changes will be identified as follows:
    i. Text to be deleted - use strikeout (example: ~~strikeout~~)
    ii. New text - use bolded-red or double-underline (example: **This is new text,** or this is new text)
d. All Draft Policies must contain the following elements:
    i. Policy title
    ii. Policy purpose
    iii. Policy statement (the text of the policy statement in clear, concise language) including, when appropriate, procedures to explain how to perform a task identified in the policy
    iv. Effective date
    v. Revision date(s), as needed
e. If appropriate, the Draft Policies may also include:
    i. Background
    ii. Scope (statement of the circumstances when the policy does and does not apply)
    iii. Definitions (defines terms for reader's convenience)
    iv. Authorized signature (indicates formal policy adoption)
    v. References to authorities or source documents upon which the policy is based.

804: Presentation and Approval

a. If the AMIE Steering Committee has requested that policies be submitted to it for approval, any such Draft Policies must be presented first to the Legal & Policy Taskforce Work Group for preliminary approval, then to the AMIE Steering Committee.
b. Any Draft Policy that requires a material commitment of resources from an AHCCCS Division that is outside the existing AMIE budget must be approved by the Assistant Director(s) from the Division(s) prior to presentation to the Legal & Policy Taskforce Work Group for approval.
c. Any Draft Policy that requires a material change to the AMIE budget (a change of 10% or more) but does not require funding from outside the AMIE budget must be approved by the Chair of the AMIE Steering Committee after

it has received preliminary approval from the Legal & Policy Taskforce Work Group.

d. One week prior to the presentation of a Draft Policy to a body for approval, the author must e-mail the approval body the Draft Policy and any other supporting documentation needed by the members for review.

e. Once the Draft Policy is approved in accordance with this policy, the presenter will distribute the new or revised policy as appropriate.

Version Information

Version 1.0 – dated 9/29/2008

Change Logs

Listed below will be significant changes to this Manual. These changes are to be reflected in the latest version. Minor changes, typographical errors, capitalizations, and other such miscellaneous changes have been made without being recorded.

Date	Section	Comment	Item	Suggested by

APPENDIX A:

INCIDENT REPORT FORM:

Name of Person completing Report: ___________________________

Date of Report: ___________________ Time of Report: __________

Person Reporting Incident:

Name: ____________________________ Department: _____________

Phone number: _______________ E-mail Address: __________________

Incident:

Severity level: ☐Low ☐Medium ☐High

System Down: Y/N ____ Sensitive Data Compromised: Y/N ____

Machine Type: ________________ O/S: __________

Incident Type: ___________________________________

Description of Incident:

____

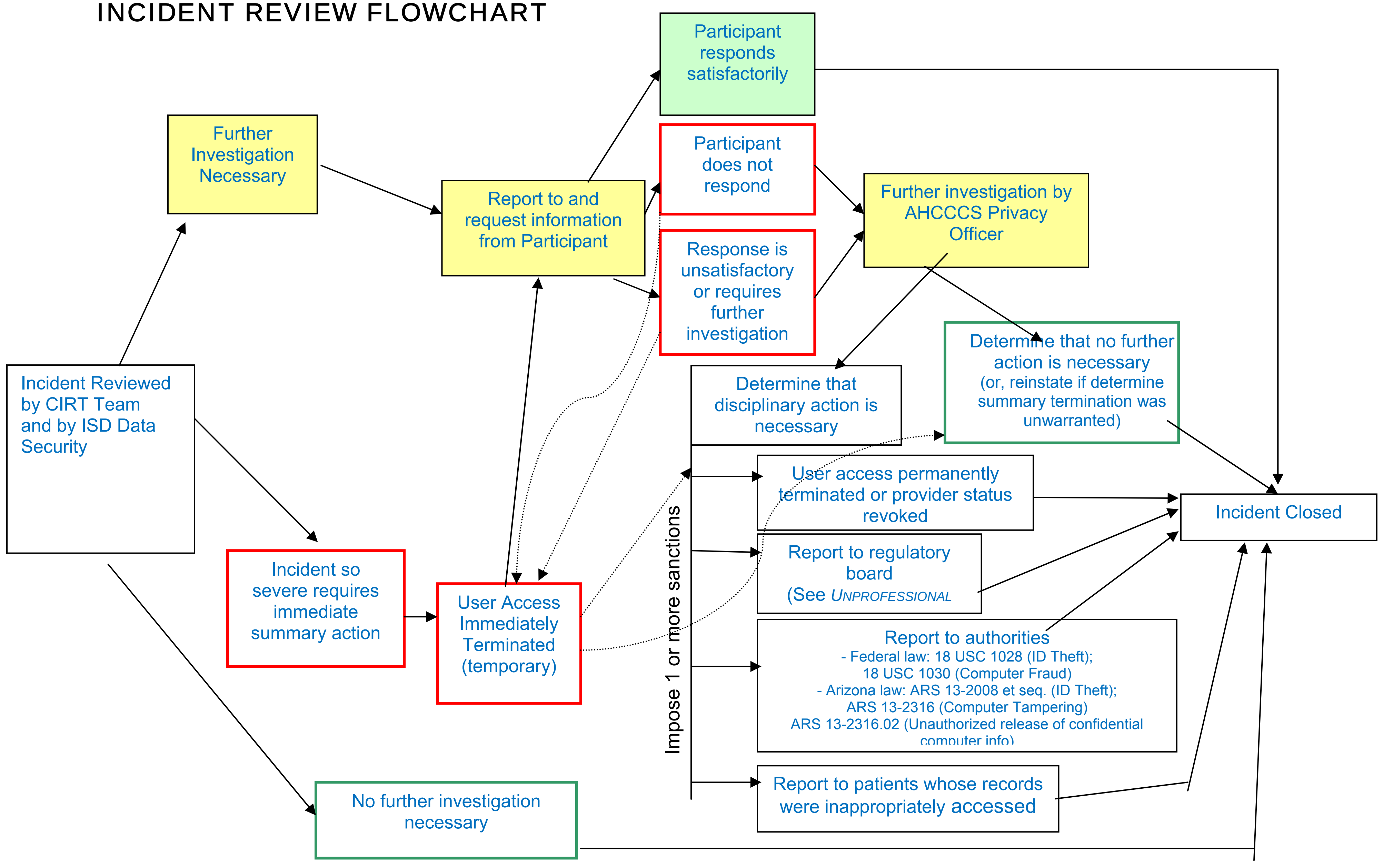
____

Status:

Assigned to: _______________________

Date and Time Assigned: _____________

<u>APPENDIX B:</u>
INCIDENT REVIEW FLOWCHART


Participant responds satisfactorily
Further Investigation Necessary
Participant does not respond
Report to and request information from Participant
Response is unsatisfactory or requires further investigation
Further investigation by AHCCCS Privacy Officer
Determine that no further action is necessary (or, reinstate if determine summary termination was unwarranted)
Incident Reviewed by CIRT Team and by ISD Data Security
Determine that disciplinary action is necessary
User access permanently terminated or provider status revoked
Incident Closed
Impose 1 or more sanctions
Report to regulatory board (See UNPROFESSIONAL
Incident so severe requires immediate summary action
User Access Immediately Terminated (temporary)
Report to authorities - Federal law: 18 USC 1028 (ID Theft); 18 USC 1030 (Computer Fraud) - Arizona law: ARS 13-2008 et seq. (ID Theft); ARS 13-2316 (Computer Tampering) ARS 13-2316.02 (Unauthorized release of confidential computer info)
Report to patients whose records were inappropriately accessed
No further investigation necessary