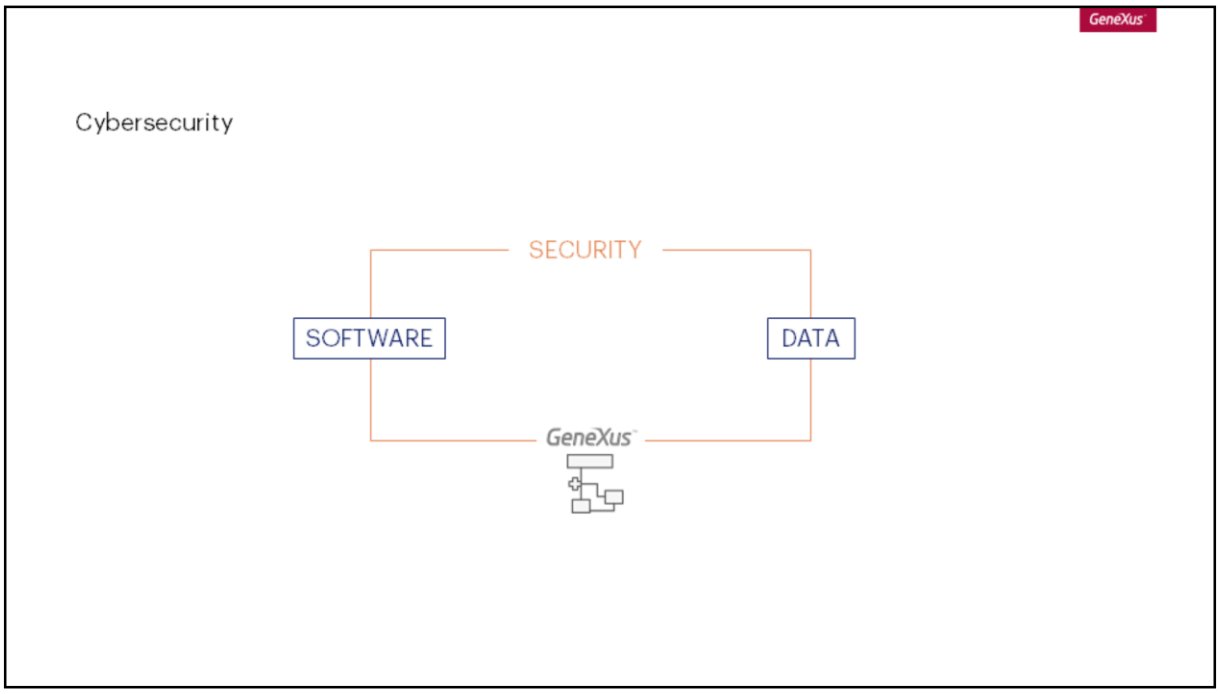
Seguridad en GeneXus

Curso GeneXus

GeneXus 16

GeneXus

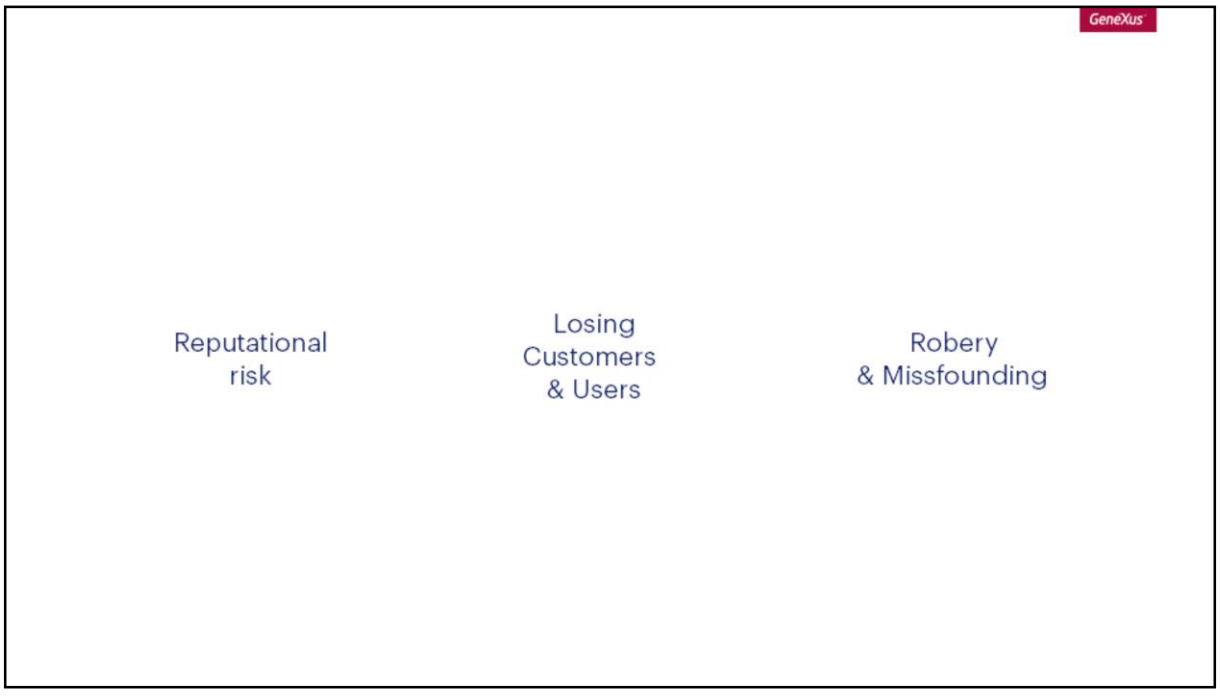
CYBERSECURITY



La **seguridad informática** (o ciberseguridad) se enfoca en la protección de la infraestructura del software y, especialmente, en la protección de datos. En esta disciplina se diseñan métodos, normas y técnicas que ayudan a desarrollar sistemas seguros y confiables.

La ciberseguridad afecta distintas capas de los sistemas: red, servidores, bases de datos, aplicaciones, etc. y en cada una de ellas existen amenazas y contramedidas. Las aplicaciones desarrolladas con GeneXus no son la excepción.

La seguridad de los sistemas es importante.



Las consecuencias de brechas de seguridad van desde un riesgo reputacional para la organización, desembocando en una pérdida de usuarios o clientes en caso de publicar información no autorizada, hasta robos en dinero o mercancía de la organización, demandas de parte de los usuarios involucrados y multas de diferentes organismos estatales.

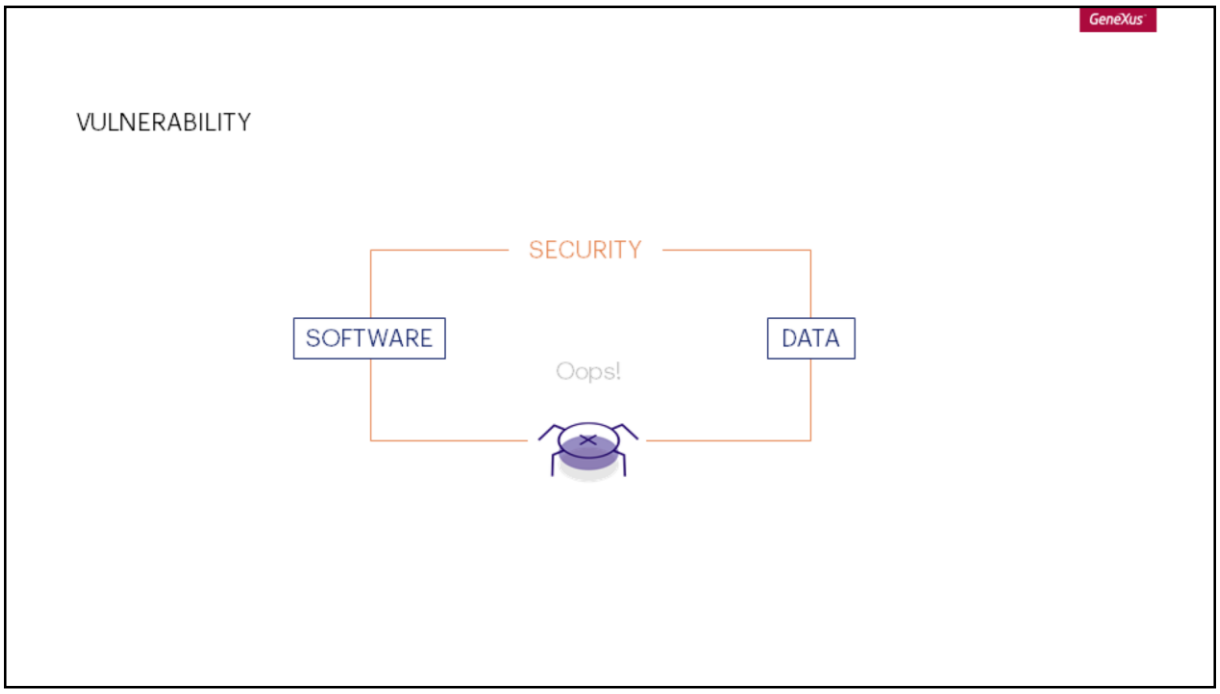
En la actualidad se están emitiendo nuevas y más estrictas regulaciones sobre privacidad, protección de datos y publicación de brechas de seguridad.

Un ejemplo de este tipo de regulación es la **Regulación General de Protección de Datos** europea en un mundo en el cual los abusos de los sistemas informáticos son más usuales.

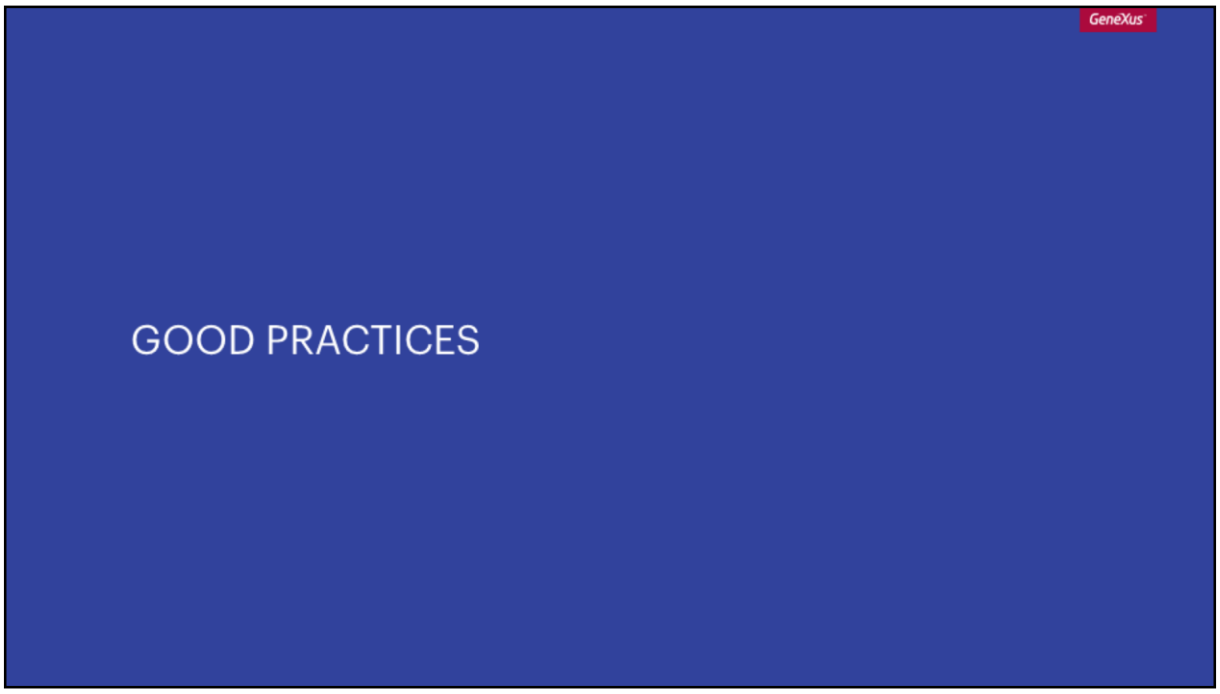
http://www.informationisbeautiful.net/visualizations/worlds-biggest-data-breaches-hacks/

GeneXus

VULNERABILITY

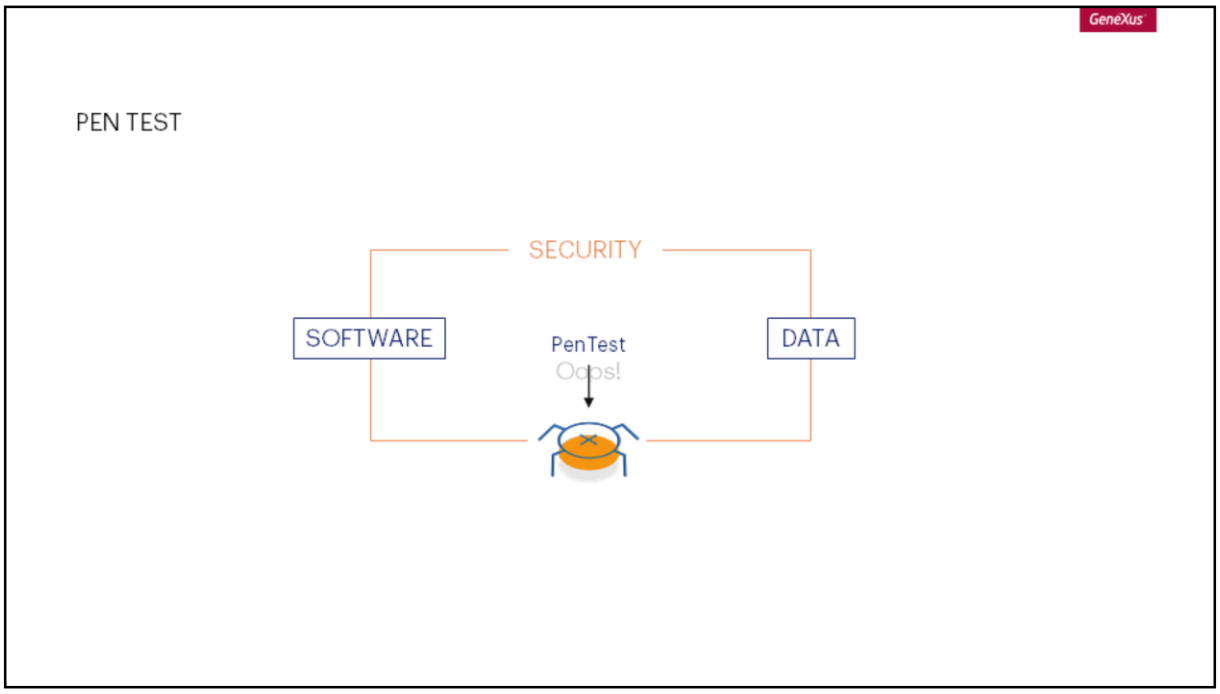


Una **vulnerabilidad** es una falla o bug del sistema que algún agente puede aprovechar para acceder al mismo de manera no autorizada ya sea para robar información o abusar de los recursos de forma tal que no pueda funcionar en forma apropiada.



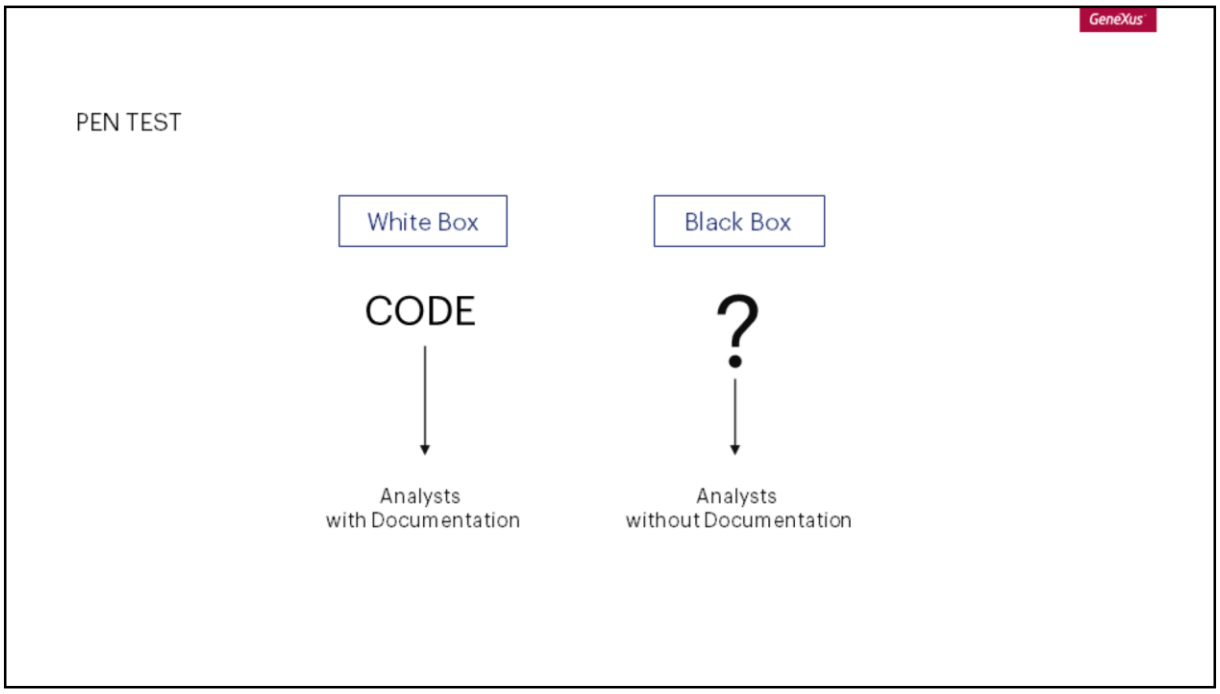
Utilizando **buenas prácticas de programación** es posible reducir estos riesgos

Tal y como propone el **Proyecto Abierto de Seguridad de Aplicaciones Web**, mediante directrices, manuales y procedimientos de codificación disponibles en la web.



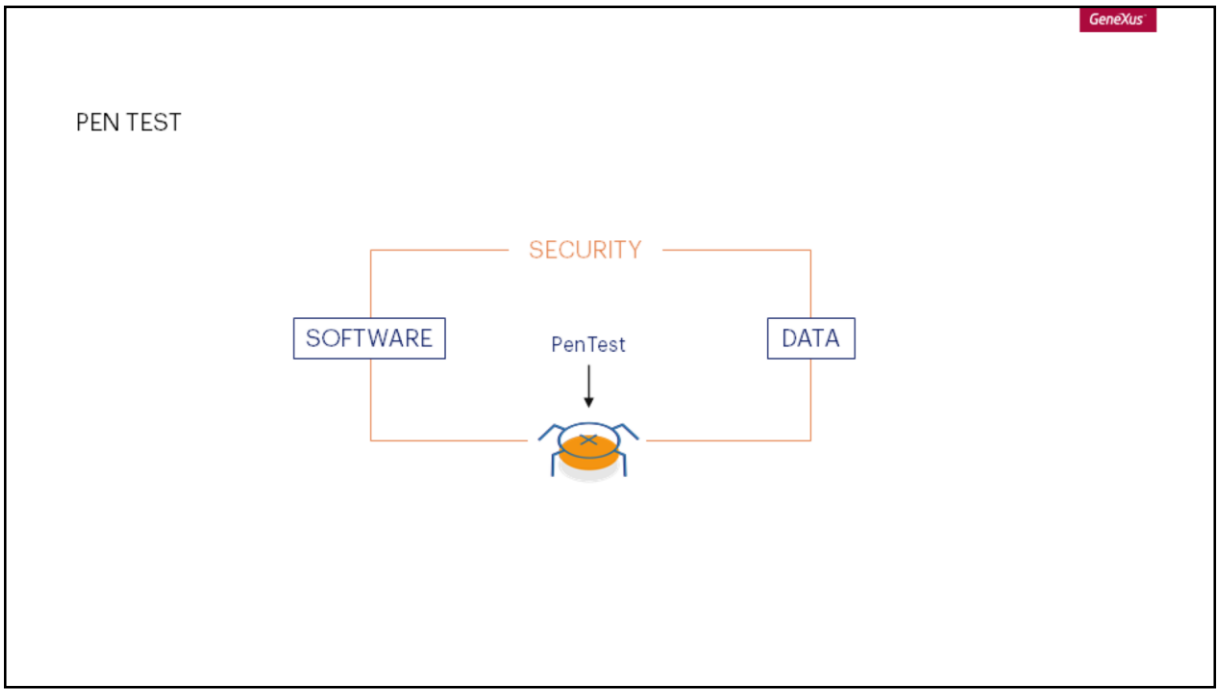
En este sentido, la instancia más conocida del ciclo de desarrollo seguro es el “pen test” (*penetration test*).

Se trata de una instancia de ataque simulado previamente autorizado para evaluar la seguridad del sistema e identificar vulnerabilidades que no se previeron durante la instancia de desarrollo.



Los pen test pueden ser de caja blanca o caja negra dependiendo si el analista tiene a su disposición el código y la documentación del sistema o no.

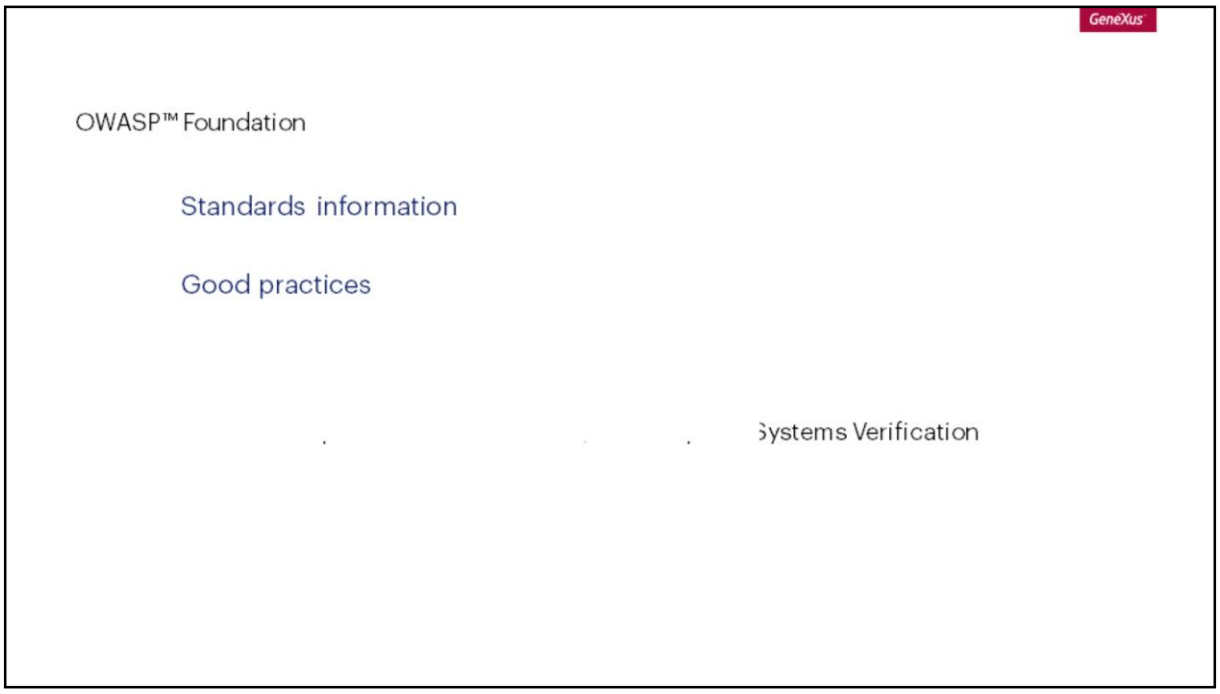
Para ello se suelen usar herramientas de análisis automatizado de código estático o dinámico, cuya validez se limita a la interpretación del analista de seguridad y al contexto del sistema analizado.



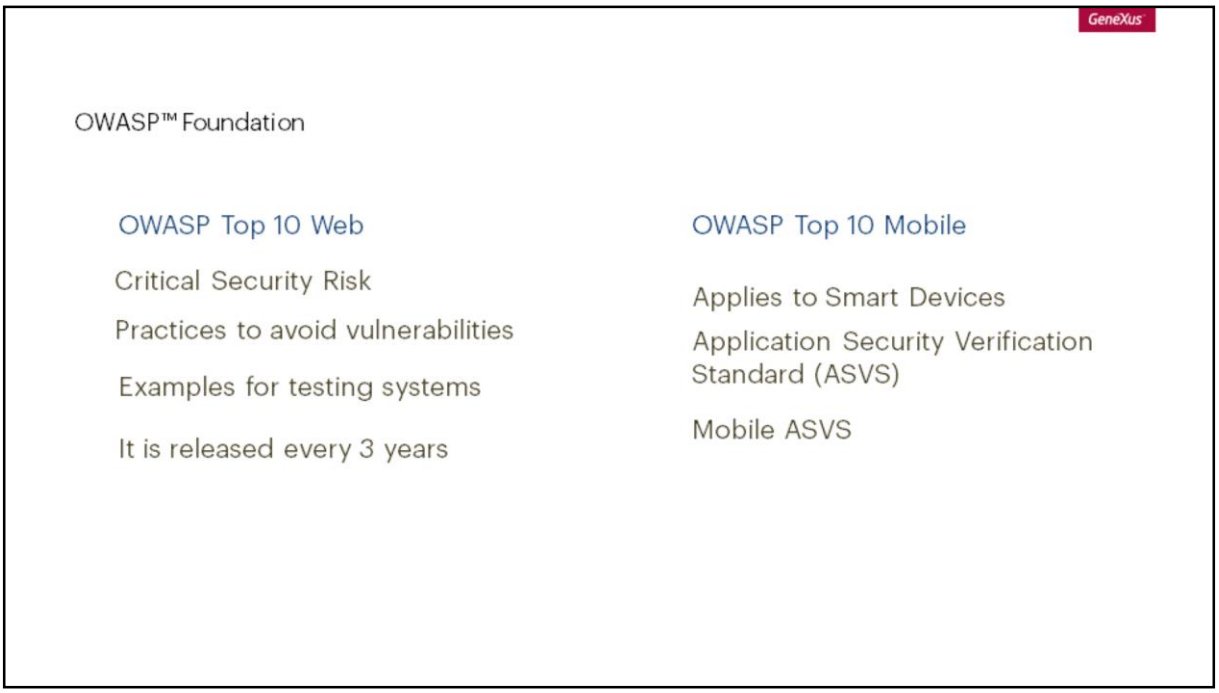
Es importante destacar que el pen test, a pesar de ser la instancia más conocida del ciclo de desarrollo seguro no es la única. La corrección de vulnerabilidades detectadas en esta etapa es más costosa frente a vulnerabilidades detectadas o prevenidas en etapas tempranas.

Un pen test exitoso y con buenos resultados tampoco garantiza que un sistema no tenga ninguna vulnerabilidad, de la misma manera que un test funcional no garantiza que un sistema no tenga ningún bug.

El Proyecto Abierto de Seguridad de Aplicaciones Web, es una comunidad abierta, sin fines de lucro y agnóstica de la tecnología dedicada a facilitar a las organizaciones el desarrollo, compra y mantenimiento de aplicaciones confiables.



Define y provee información sobre estándares, buenas prácticas y herramientas para el desarrollo y la verificación de código y sistemas informáticos desde una perspectiva de seguridad.



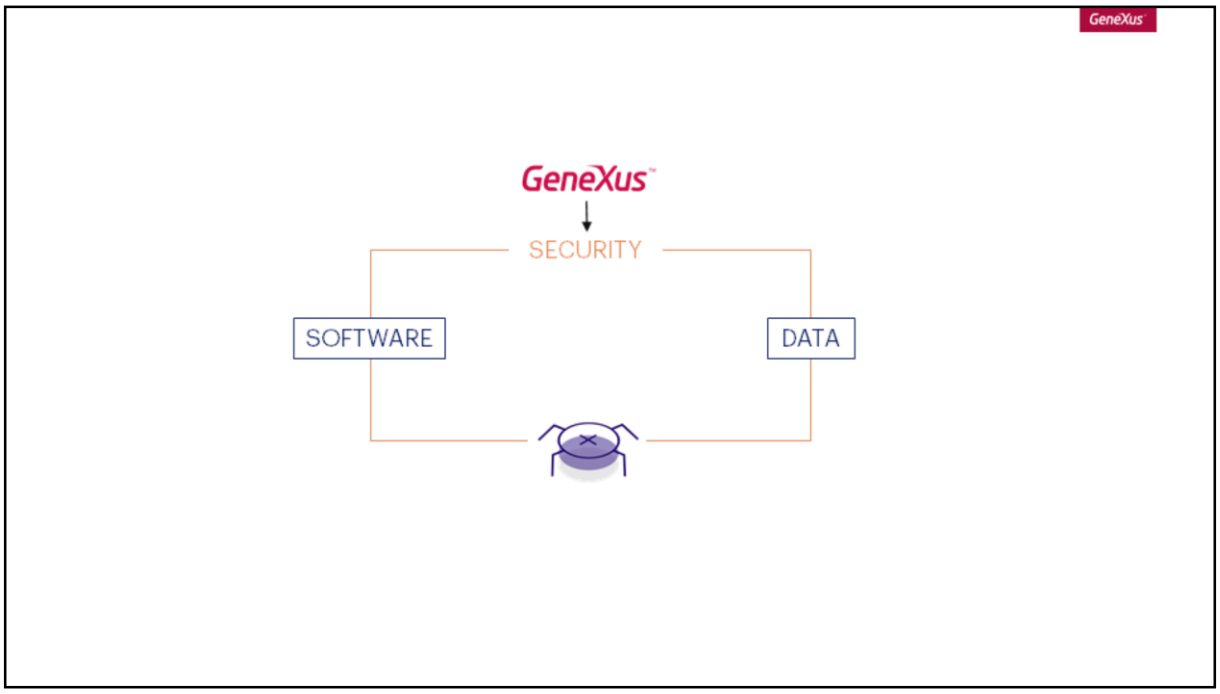
Dentro del Proyecto Abierto de Seguridad para Aplicaciones Web, existen varios proyectos.

Uno de los más destacados y con mayor relevancia es el **Top 10 Web**, un documento que representa el consenso de los expertos sobre los riesgos de seguridad más críticos dado su impacto y frecuencia. El objetivo del documento es generar conciencia de la problemática de seguridad de sistemas, brindar guías de buenas prácticas para evitar las vulnerabilidades listadas, proveer ejemplos para testear sistemas que puedan tener esas vulnerabilidades y recomendar herramientas de test automatizado para su análisis.

Este Top 10 Web se libera cada 3 años siendo el último liberado y vigente en la actualidad el correspondiente al año 2017.

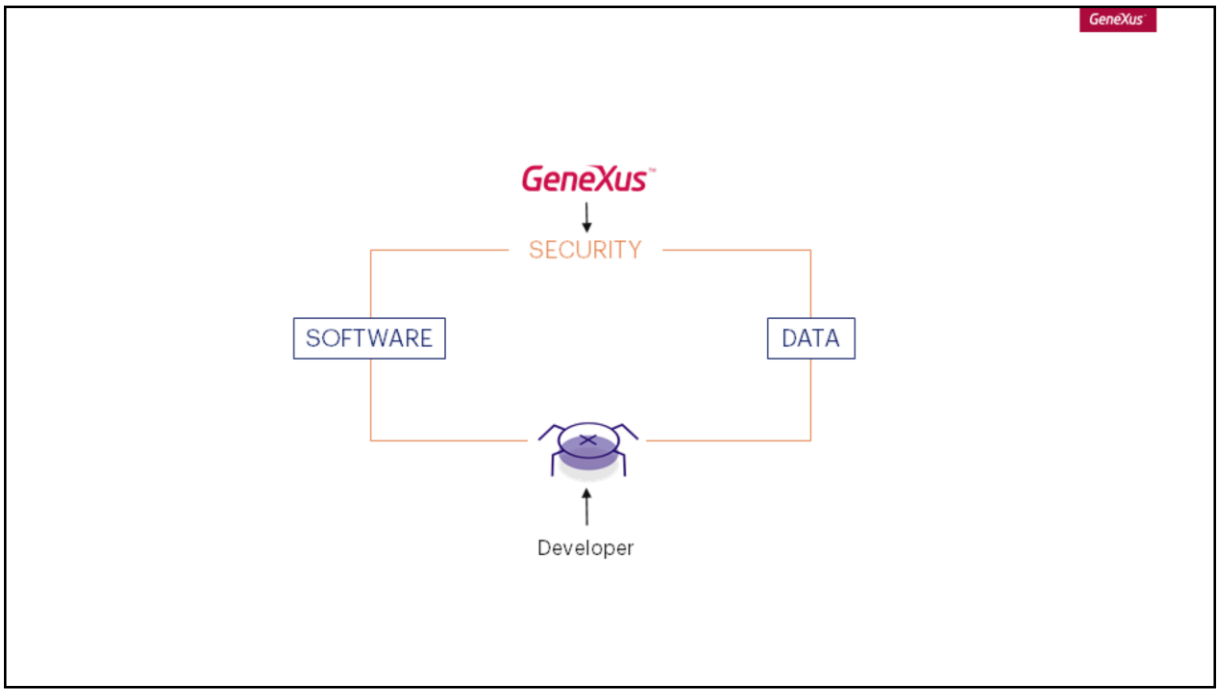
Existe también el **Top 10 Mobile** siendo el último liberado el correspondiente al año 2016 que aplica a Smart Devices. Además de esto, el Proyecto Abierto de Seguridad para Aplicaciones Web incluye un proyecto llamado **ASVS** ***(Application Security Verification Standard*** **)** que define niveles de verificación de seguridad mínimos y necesarios según el contexto de la aplicación.

Existe también el **Mobile ASVS**, siendo la versión vigente la 1.1 (al año 2018).



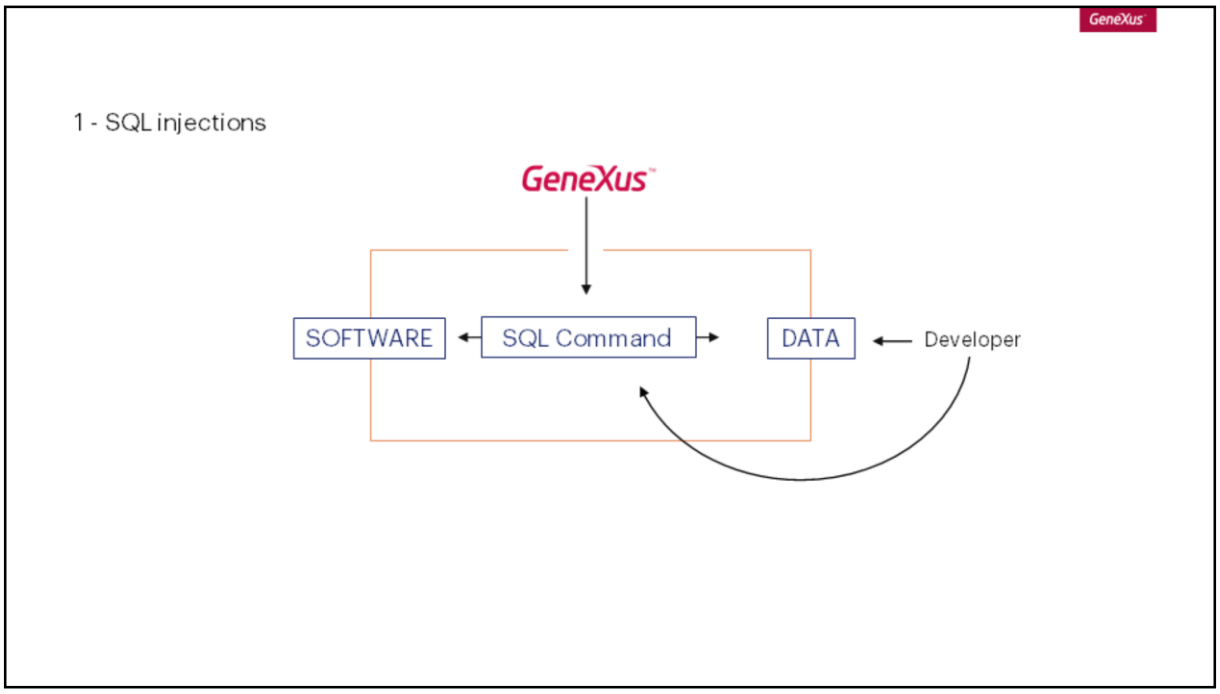
Ahora bien. ¿Qué sucede con GeneXus y los controles de seguridad?

Genexus implementa de forma automática aquellos controles de seguridad que pueden ser inferidos de la base de conocimiento. Dicho en otras palabras, aquellas vulnerabilidades provenientes por errores de codificación son mitigadas al utilizar un lenguaje como GeneXus.



No obstante, el desarrollador a la hora de utilizar la herramienta puede introducir vulnerabilidades o defectos de seguridad que se convierten en fallas y errores, tanto a nivel de la lógica de negocios como del código procedural GeneXus.

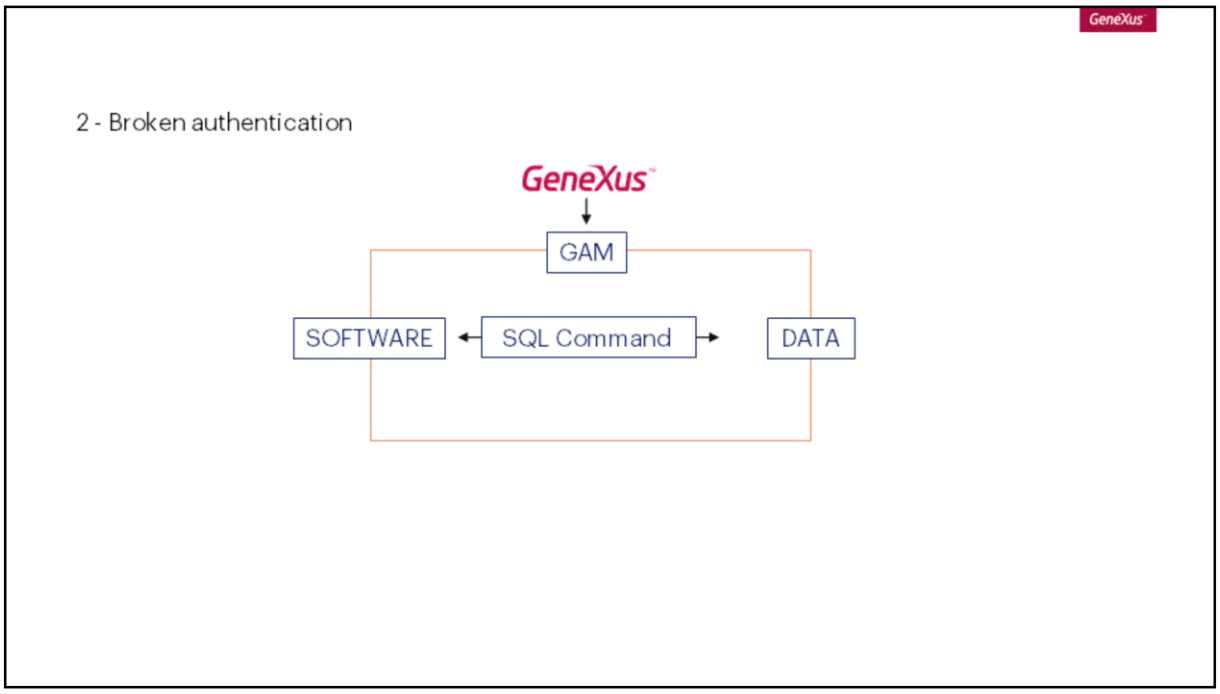
A modo de ejemplo, utilizando el Top 10 2017 del Proyecto Abierto de Seguridad para Aplicaciones Web, se pueden enumerar ciertas mitigaciones que GeneXus realiza automáticamente y casos en los que el desarrollador puede introducir fallas a pesar de las medidas que toma GeneXus.



El primer ítem del Top 10 2017 son las **Inyecciones**.

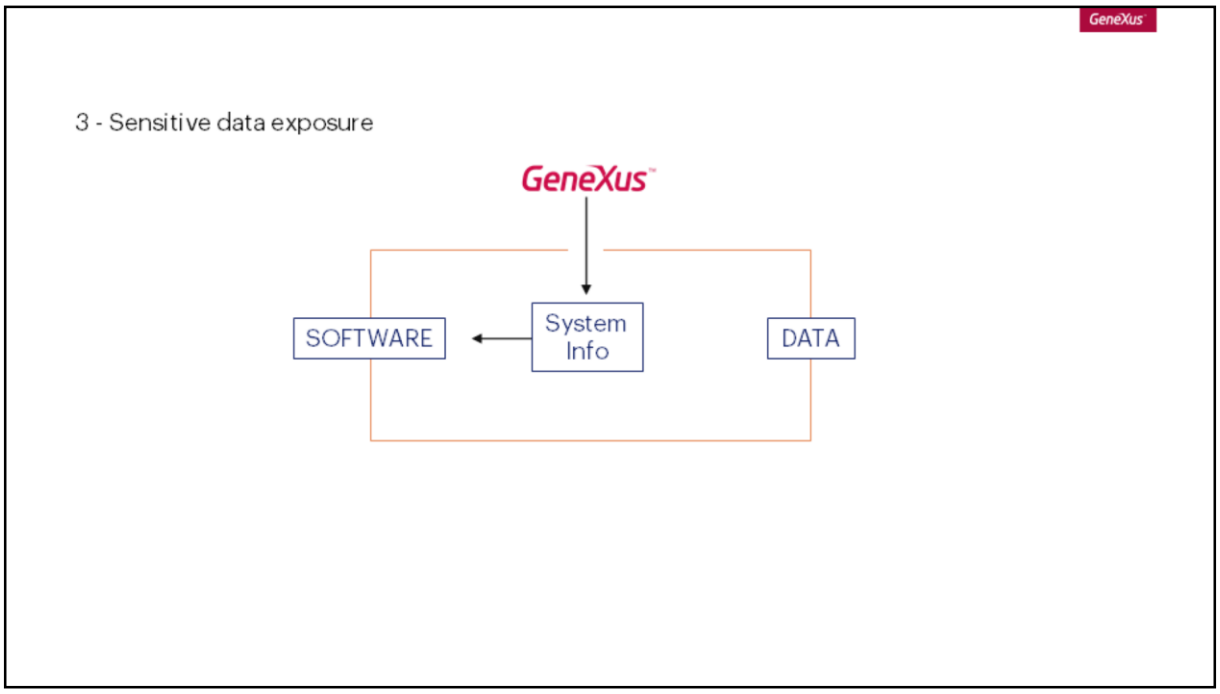
Las más conocidas son las inyecciones SQL, aunque no son las únicas. Estas vulnerabilidades se suceden cuando información proveniente del usuario se envía a la base de datos sin validar.

GeneXus expone el Comando SQL que dota de más versatilidad a la aplicación para que el desarrollador pueda utilizar la base de datos como recurso. En este caso, es el desarrollador quien debe sanitizar las entradas del comando para prevenir una inyección SQL.



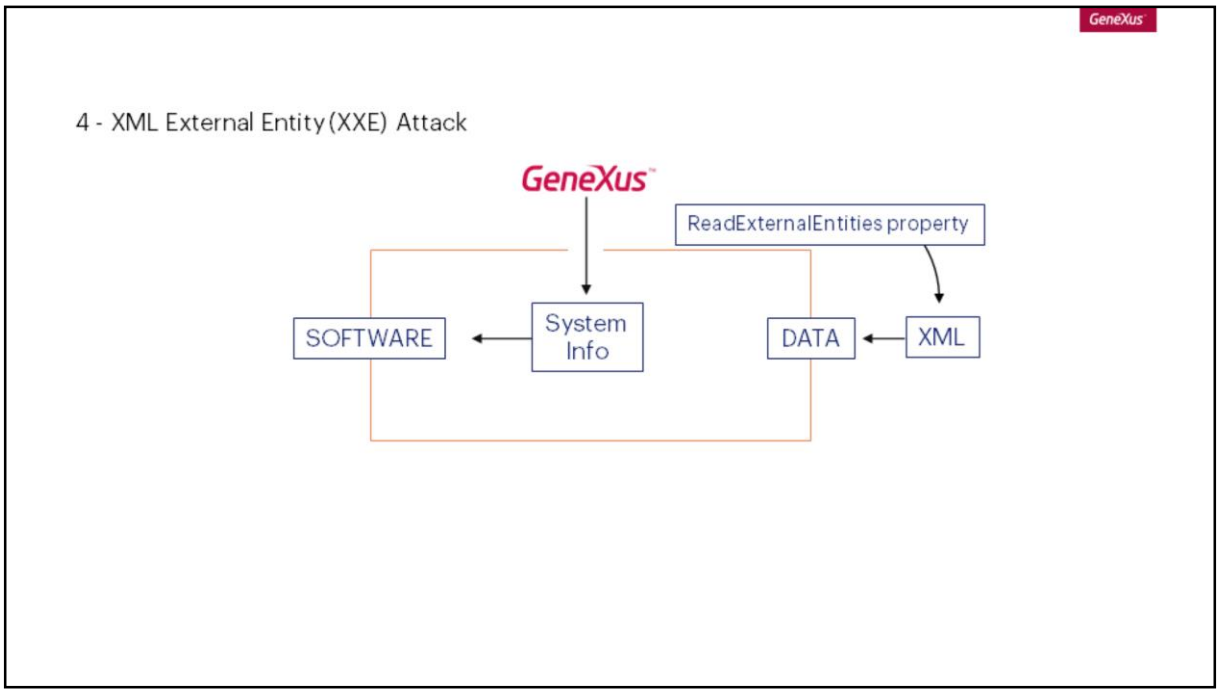
En cuanto al segundo ítem, **el Broken Authentication**, GeneXus ofrece un módulo de Autenticación/Autorización denominado GAM (*Genexus Access Manager*), permitiendo ejecutar estas tareas en forma automática. La configuración adecuada de la herramienta depende del desarrollador y su escenario concreto.

También es posible utilizar un módulo propio siendo entonces responsabilidad del desarrollador el manejo de contraseñas, sesiones, etc.



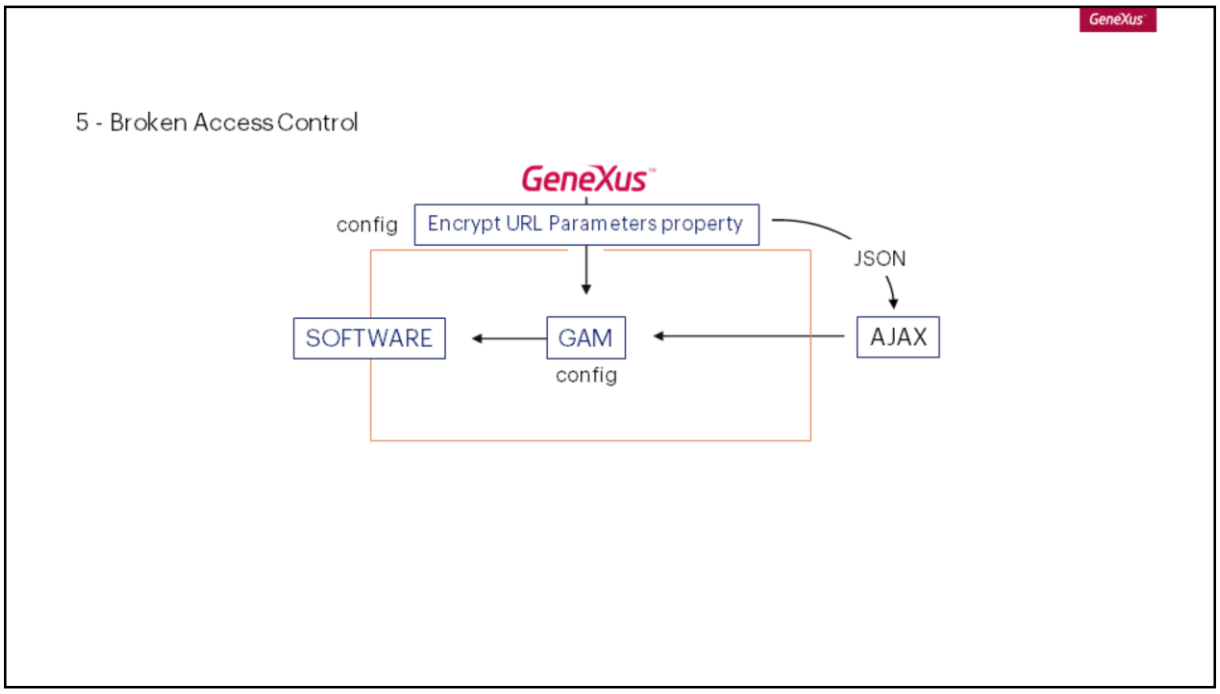
El tercer punto del Top 10 2017, corresponde a la **Exposición de datos sensibles**, y es parte del análisis de requerimientos determinar cuál es la información sensible del sistema.

Por lo tanto, GeneXus no ejecuta ninguna acción en particular, pero sí brinda las funciones necesarias para la protección de la información sensible; como ser, funciones de cifrado, hash, manejo de claves y certificados.



El siguiente punto es el **Ataque de una Entidad externa XML**.

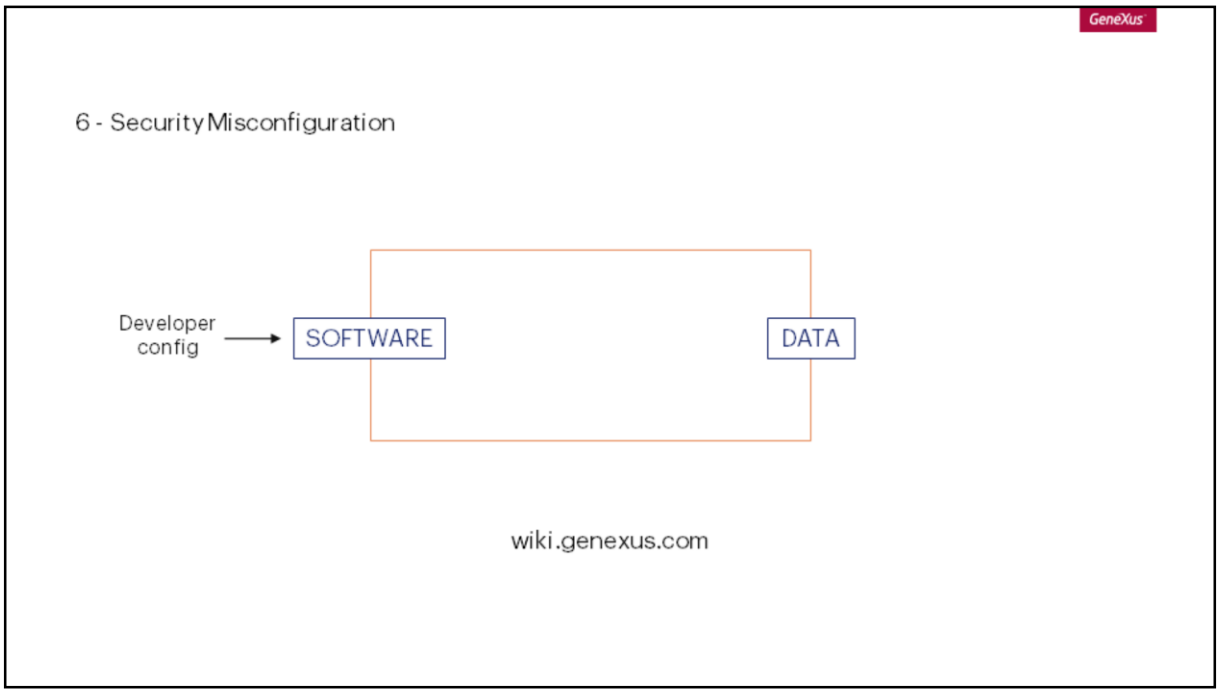
En este caso, por defecto, GeneXus no procesa ninguna entidad externa de XML. Para habilitar el procesamiento de External Entities es necesario configurar la propiedad ReadExternalEntities. En ese caso, es también responsabilidad del desarrollador controlar el XML si cruza la frontera del sistema.



El siguiente item corresponde al **Contro de Acceso roto**, que se produce cuando se dan accesos no permitidos al sistema, alterando URLS, llamadas AJAX, etc.

Para esto GeneXus dispone de una propiedad que permite encriptar parámetros URL con la cual se cifran todos los parámetros que recibe un objeto, controla el acceso con GAM, ejecuta la codificación para JSON en las llamadas AJAX por defecto y ejecuta también chequeos de validación, no solamente del lado del cliente, sino también del lado del servidor.

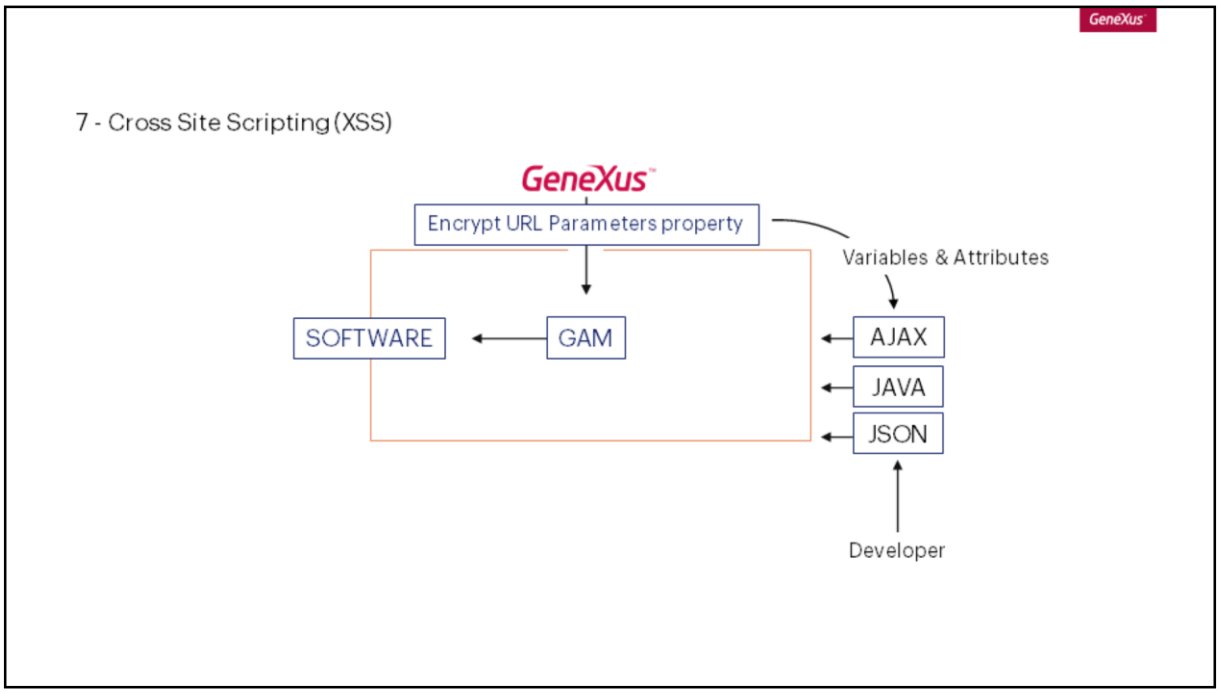
En este caso, es responsabilidad del desarrollador agregar controles en los eventos, configurar GAM de manera apropiada y configurar también esta propiedad que permite encriptar los parámetros URL.



El sexto punto de este Top 10, corresponde a las **Malas configuraciones de Seguridad**, y refiere mayormente a configuraciones erróneas en el servidor de aplicaciones de producción. Para este caso GeneXus tiene disponibles algunas propiedades que, si bien no es aconsejable deshabilitar, el desarrollador puede ajustarlas para satisfacer las necesidades del prestador.

En el Wiki de GeneXus se pueden encontrar algunos consejos y lineamientos para una configuración de servidores segura.

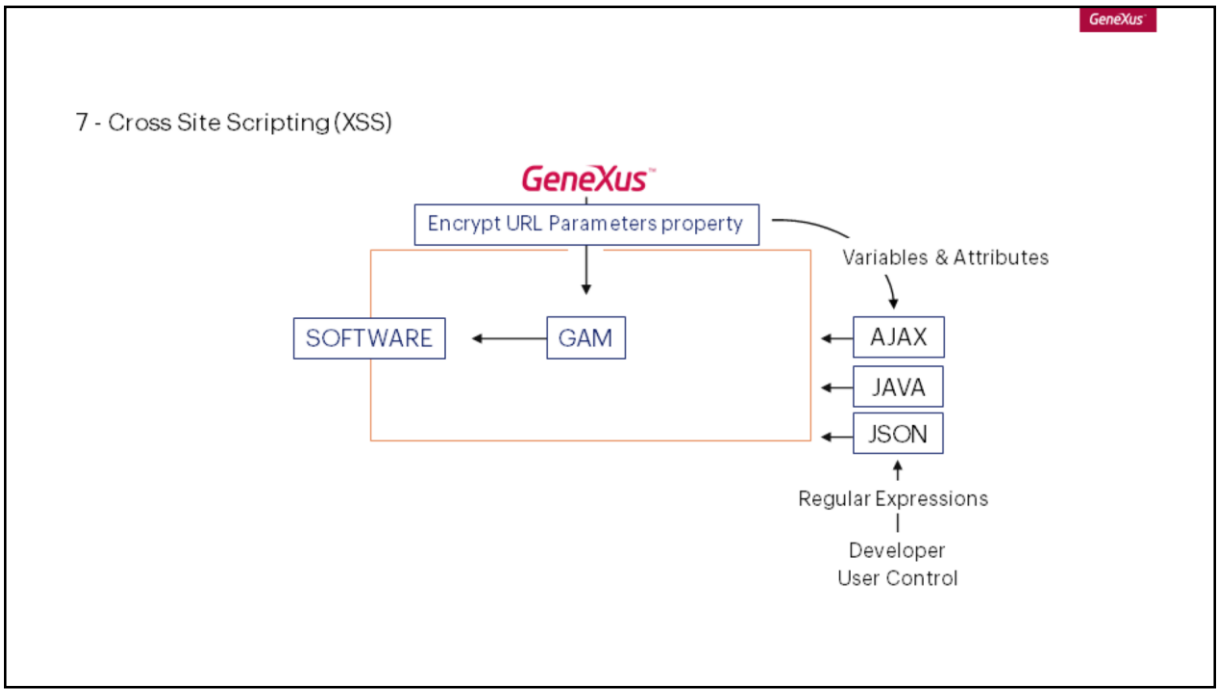
Sin embargo, el nivel de seguridad en producción depende en mayor medida del encargado de infraestructura que configura el servidor de la aplicación.



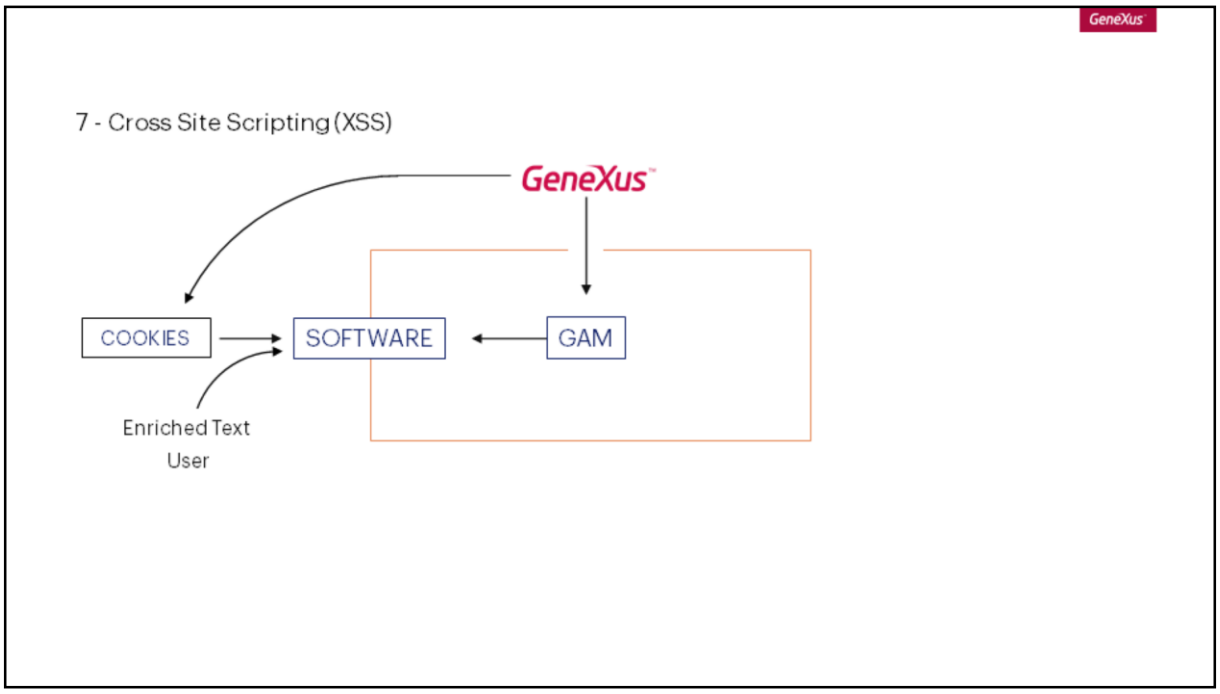
Como séptimo punto del Top 10 2017 aparece la **Secuencia de comandos de sitios cruzados**.

Para este tipo de vulnerabilidades, Genexus valida automáticamente el largo y tipo de las variables y atributos, y ejecuta la codificación de los datos que recibe el sistema según su contexto (ya sea HTML JavaScript, JSON, etc.) también de manera automática.

En el caso que el desarrollador introduzca código HTML o Javascript propio, es su deber validarlo y controlarlo.

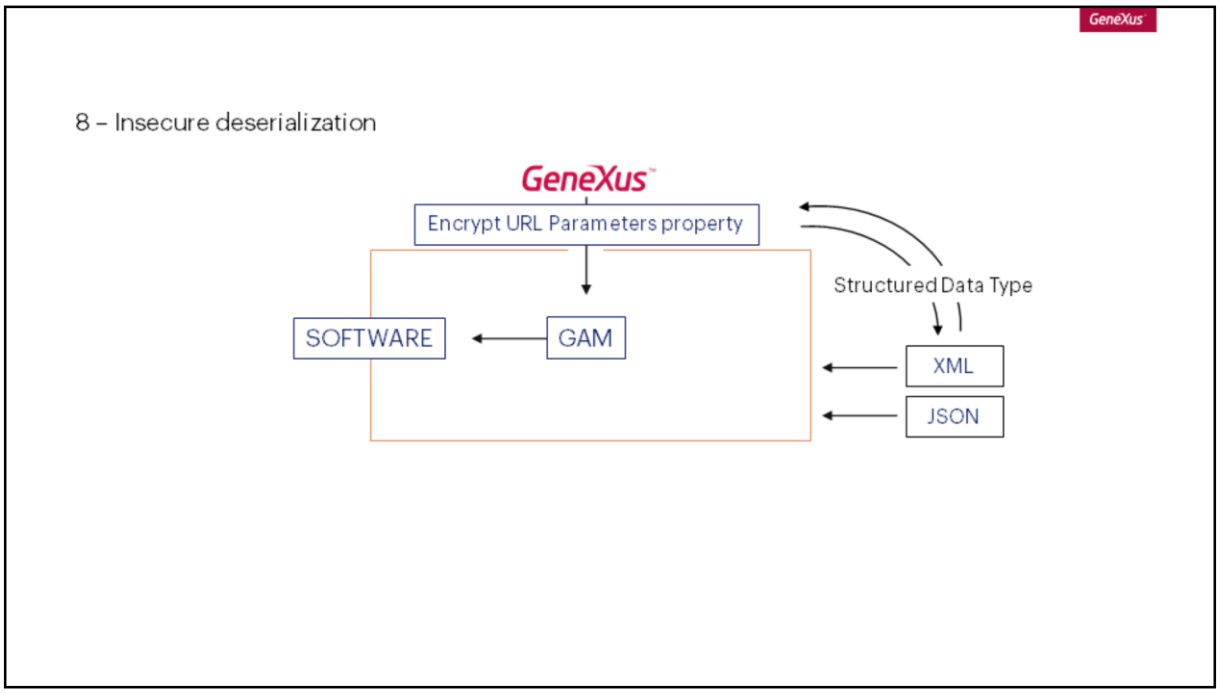


Asimismo, GeneXus brinda la posibilidad a los desarrolladores de definir Expresiones Regulares para validar las entradas de los usuarios o bien utilizar algún User Control propio o de terceros (o sea, ajeno a GeneXus), siempre y cuando el desarrollador tenga la plena seguridad y confianza que sea seguro, y cumpla con los controles de validación previamente definidos.



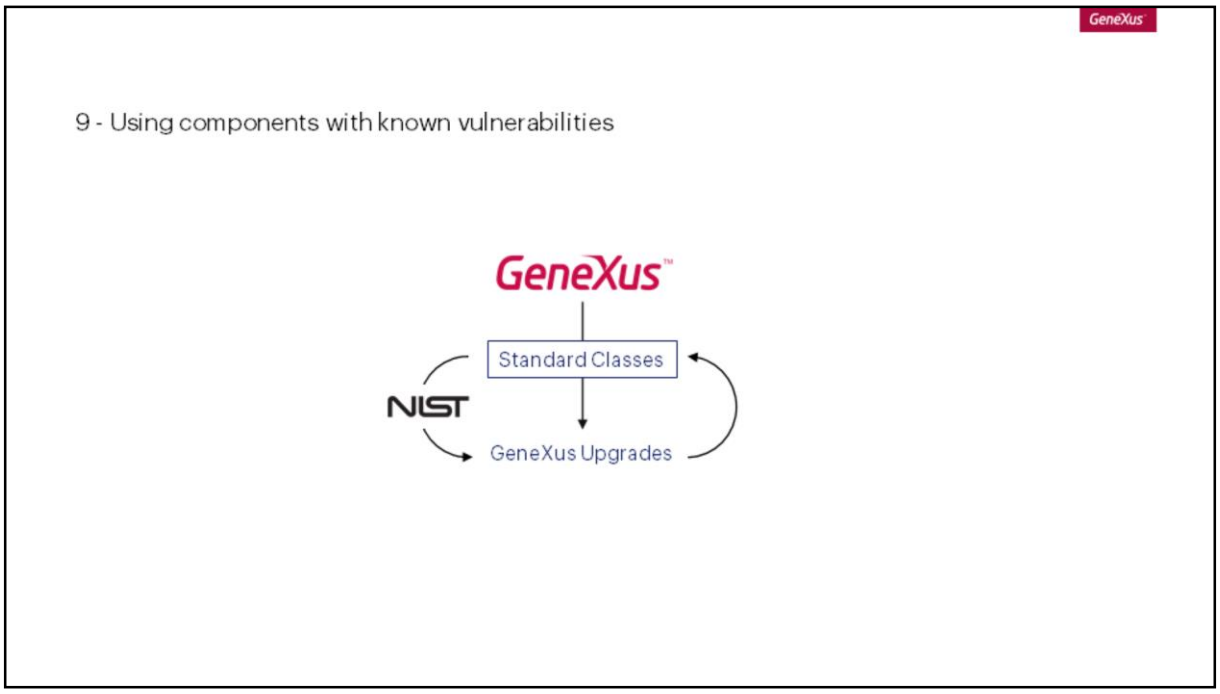
También hay que tener en cuenta la configuración apropiada de las *cookies*, y evitar deshabilitar los chequeos generados automáticamente por GeneXus (si no es imprescindible).

En caso de admitir texto enriquecido por parte del usuario, no olvidar nunca controlar, validar y dar formato a esta información de manera segura y acorde.



El octavo punto refiere a la **Deserialización insegura**, para el cual GeneXus implementa mecanismos de serialización seguros cuando se manejan tipos de datos estructurados hacia JSON y/o XML.

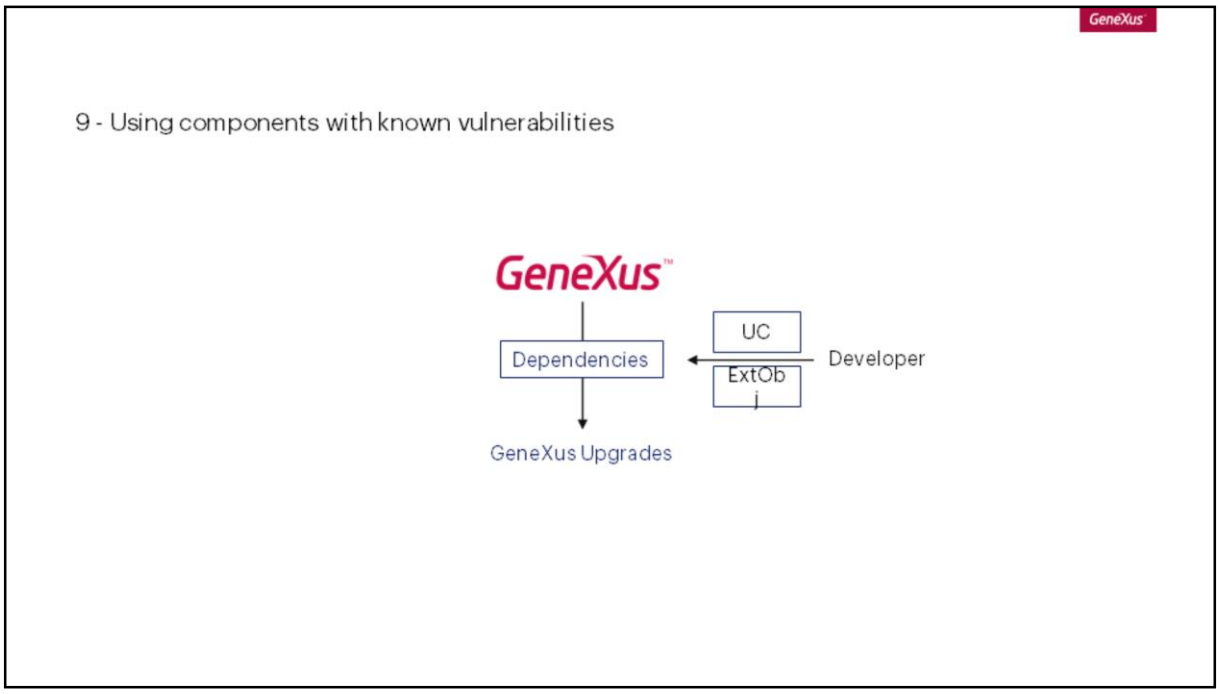
Cuando se serializan objetos desde XML o JSON utilizando los métodos o tipos de datos es necesario codificar las entradas.



El noveno ítem del listado del Proyecto abierto de Seguridad para Aplicaciones Web, corresponde al **Uso de componentes con vulnerabilidades conocidas**.

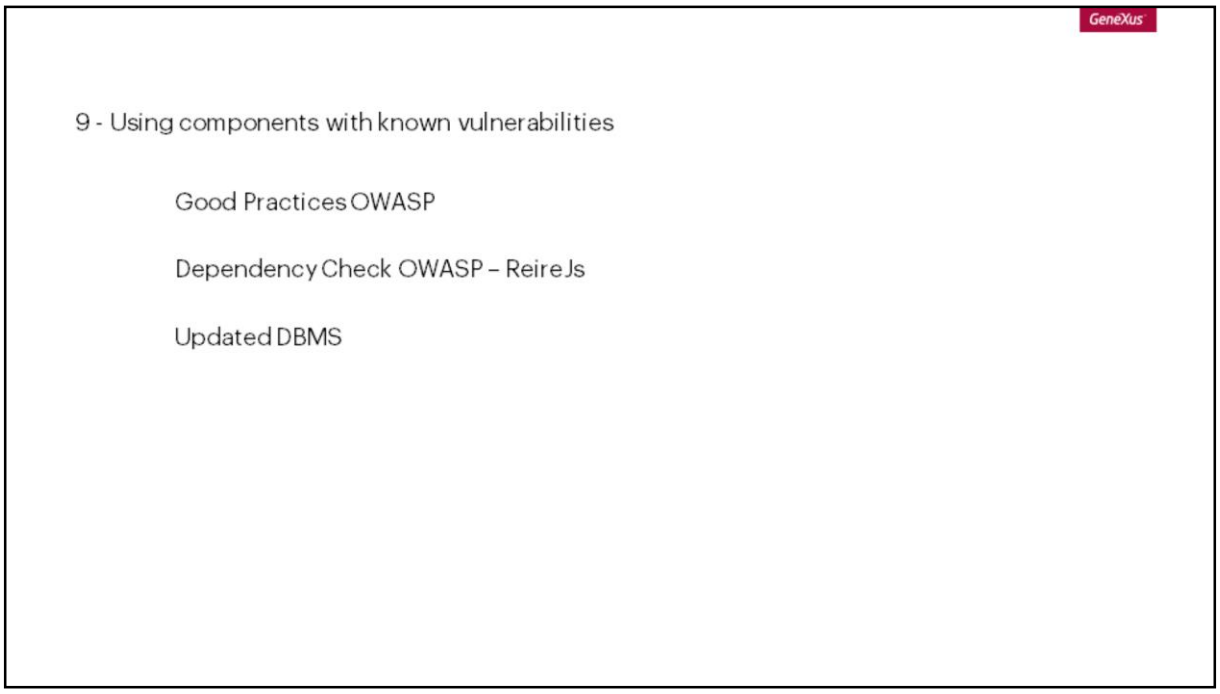
Para este caso GeneXus tiene sus propias clases estándares que están siendo permanentemente revisadas y corregidas en los sucesivos *upgrades*.

Pero también utiliza dependencias de terceros que son validadas contra la base de datos de vulnerabilidades del Instituto Nacional de Estándares y Tecnología, actualizándose, de ser necesario, en cada upgrade de GeneXus.



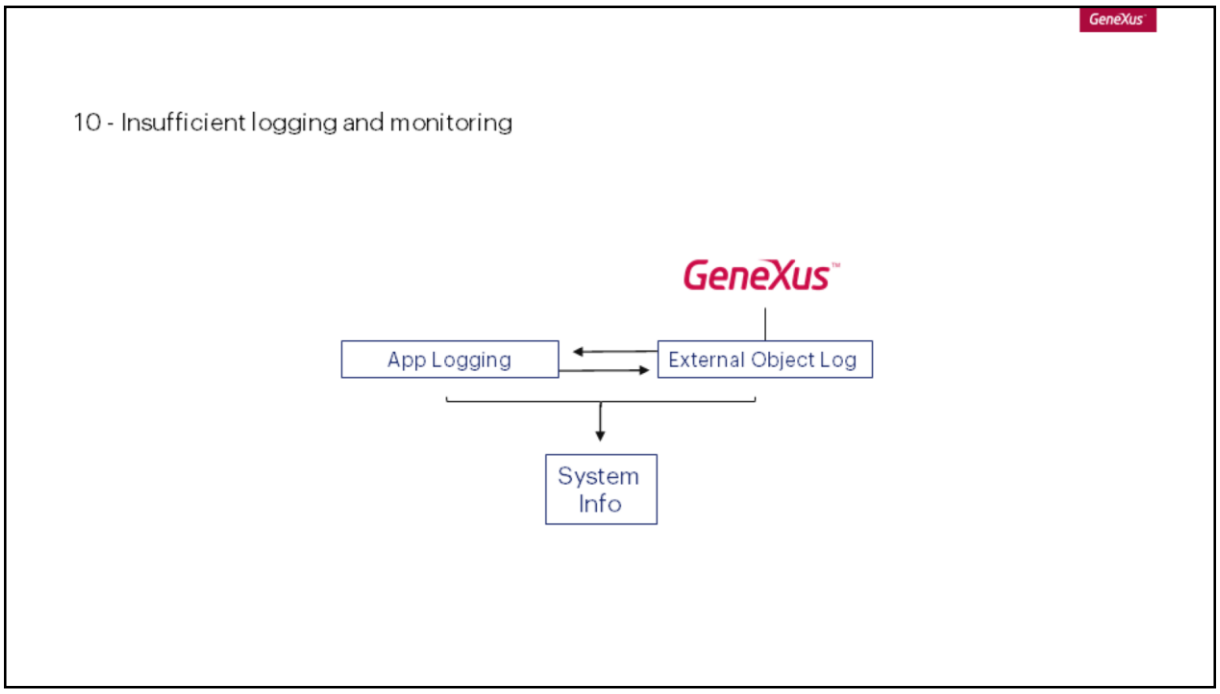
Es decir, que GeneXus se encarga de sus propias dependencias, actualizandolas con los distintos upgrades.

En caso de que el desarrollador decida agregar un User Control o un External Object se recomienda seguir las buenas prácticas de desarrollo seguro de este Proyecto Abierto de Seguridad para Aplicaciones Web, y revisar las dependencias que se incluyan



Se sugiere para esto el uso de diveras herramientas., y se recomienda también mantener los DBMS (*DataBase Managment System*) y sus drivers actualizados, así como también instalar actualizaciones y parches de seguridad en los servidores.

.



El último punto de la lista corresponde al **Insuficiente Logging y Monitoreo**.

En todos los casos se debe definir un proceso de monitoreo para detectar posibles eventos o incidentes de seguridad.

Si bien GeneXus provee mecanismos de logging a nivel de la aplicación y el External Object Log para agregar informacion personalizada al log de la aplicacion, es necesario definir la información relevante provista por los distintos dispositivos involucrados (como por ejemplo los servidores web o firewalls), y además definir los protocolos para el relevo y categorización de la información recogida.

De esta forma, se pueden detectar eventos que amenacen el sistema, y así prevenir un posterior incidente.

GeneXus

Security Scanner

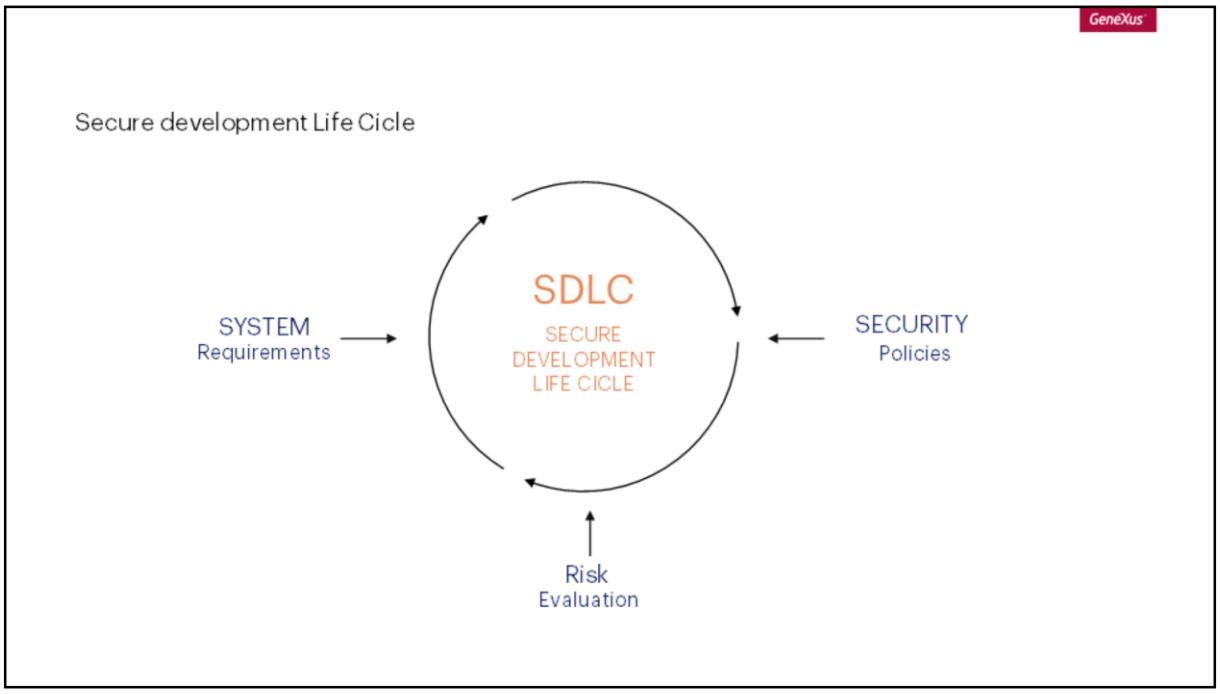
Para detectar casos conocidos en los que los desarrolladores pueden introducir errores se puede utilizar el Security Scanner.

Se trata de es una Extensión para el IDE de GeneXus, disponible en el marketplace, que ejecuta un análisis estático de los objetos de la base de conocimiento, buscando indicios de problemas de seguridad ya conocidos.

La última versión de la extensión (3.6.0.0) es compatible con Genexus 16 y está actualizada para ejecutar una revisión basada en el Top 10 2017 del Proyecto Abierto de Seguridad para Aplicaciones Web con documentación disponible en el Wiki.

GeneXus

Secure development Life Cicle



Para desarrollar sistemas seguros es necesario implementar un **Ciclo de vida seguro**.

Esto es, tomar en cuenta la seguridad desde el momento en que se recaban los requerimientos del sistema. Esto se realiza mediante una evaluación de riesgos en base a una política de seguridad que debe persistir durante todo el ciclo de desarrollo (om sea, durante el diseño, implementación, test, implantación y mantenimiento).

GeneXus™

Videos	training.genexus.com
Documentation	wiki.genexus.com
Certifications	training.genexus.com/certifications