UNIVERSITY OF NAIROBI



College of Biological and Physical Science

School of Computing and Informatics

**Enhancing data security in software testing in a sandbox environment**

By

PETER CHEGE KARIUKI

P54/12142/2018

A proposal submitted to the university of Nairobi in partial fulfillment of the Master of Science Degree in Information Technology Management

**ABSTRACT**

Protecting data from unauthorized access is a major concern in many organizations. Big data security becomes a concern due to its vulnerability to security breach either in transit or at rest. A lot of emphasis and measures are put in place to protect data in transit than at rest. For data at rest, the major threat who are overlooked in enforcing CIA are the Software developers. The security breach in a sandbox environment occurs when programmers gain access to server to debug an issue. They gain access to confidential data thereby violating the CIA rule of information security. The organizations affected believe in ethics and rights regarding privacy of information thereby trusting the programmers with their data. Tests on the data correlation and system performance are performed in a sandbox environment before being transferred to a live environment. This research will seek to address how to enhance data security during software testing.

beThe problem Ethics in the programmers foregoing the rule ethics play a key role in programming systems or in transit necessitate enforcement of CIA in every stage. For software developers it becomes critical to

List of abbreviations

# CHAPTER ONE: INTRODUCTION

## Background

A sandbox is an isolated testing environment that enables users to run programs or execute files without affecting the application, system or platform on which they run. Software developers use sandboxes to test new programming code. Sandboxing protects "live" servers and their data, vetted source code distributions, and other collections of code, data and/or content, proprietary or public, from changes that could be damaging (regardless of the intent of the author of those changes) to a mission-critical system or which could simply be difficult to revert. Sandboxes replicate at least the minimal functionality needed to accurately test the programs or other code under development. The common approach used is to copy production data to test. This helps the tester, to detect the same issues as a live production server, without corrupting the production data. (e.g. usage of the same environment variables as, or access to an identical database to that used by, the stable prior implementation intended to be modified; there are many other possibilities, as the specific functionality needs vary widely with the nature of the code and the application[s] for which it is intended).

The concept of the sandbox also called a working directory, a test server or development server) is typically built into revision control software such as CVS and Subversion (SVN), in which developers "check out" a copy of the source code tree, or a branch thereof, to examine and work on. Only after the developer has fully tested the code changes in their own sandbox should the changes be checked back into and merged with the repository and thereby made available to other developers or end users of the software.

Data security prevents unauthorized access, use, disclosure, disruption, modification, inspection, recording or destruction of information. The Information security's primary focus is the balanced protection of the confidentiality, integrity and availability of data (also known as the CIA triad) Confidentiality ensures that information is not made available or disclosed to unauthorized individuals. Integrity is maintaining and assuring the accuracy and completeness of data over its entire lifecycle entities, or processes. Availability ensure the information must be available when it is needed. This focuses on efficient policy implementation, all without hampering organization productivity.

To standardize this discipline, academics and professionals(give citation) collaborate and seek to set basic guidance, policies, and industry standards on encryption software, legal liability and user/administrator training standards. This standardization may be further driven by a wide variety of laws and regulations that affect how data is accessed, processed, stored, and transferred. However, the implementation of any standards and guidance within an entity may have limited effect if a culture of continual improvement isn't adopted.

In kenya the Data Protection Bill (the “Bill”) was introduced to establish a comprehensive data protection. The principal object of the Bill is to protect personal data collected, used or stored by both private and public entities. The Bill recognizes that data protection forms part and parcel of the expectation of the right to privacy. It provides for the legal framework for protection of a person’s privacy in instances where personal information is collected, stored, used or processed by another person.

According to the Bill, personal data is accessible to those whom it concerns, and provides redress to individuals if there are inaccuracies. The law also gives users the right to decline to have their data collected or processed as well as demand to have false data corrected or deleted upon demand. There is urgent need to put in place rules to regulate the collection, use, storage and processing of personal information. The new law will compel companies to inform users when their personal data is being actively collected and processed and report on the outcome of this processing

Data Protection Act is used to ensure that personal data is accessible to those whom it concerns, and provides redress to individuals if there are inaccuracies. The international standard ISO 27001 covers data and information security.

Audit Standards

Data Security is subject to several types of audit standards and verification, the

most common are ISO 27001, PCI, ITIL. Security Administrators are responsible

for creating and enforcing a policy that conforms to the standards that apply to

their business.

IT certification audits are generally carried out by 3rd parties although regular

internal audits are recommended. Clients can also carry out audits before they

begin doing business with a company to ensure that their clients data is secured

to their standards.

Security Policy

A security policy is a comprehensive document that defines a company’s

methods for prevention, detection, reaction, classification, accountability of data

security practices and enforcement methods. It generally follows industry best

practices as defined by ISO 27001, PCI, ITIL or a mix of them. The security

policy is the key document in effective security practices. Once it has been

defined it must be implemented and modified and include any exceptions that

may need to be in place for business continuity. Most importantly all users need

to be trained on these best practices with continuing education at regular

intervals.

Securing data

Data needs to be classified in the security policy according to its sensitivity. Once

this has taken place, the most sensitive data has extra measures in place to

safeguard and ensure its integrity and availability.

All access to this sensitive data must be logged. Secure data is usually isolated

from other stored data and it is important that controlling physical access to the

data centre or area where the data is stored is implemented.

Active Directory for example is used by many companies and is a centralised

authentication management system that is used to control and log access to any

data on the system.

Encryption of the sensitive data is critical before transmission across public

networks. The use of firewalls on all publicly facing WAN connections needs to

be in place and also the deployment of VLANs’ to isolate sensitive departments

from the rest of the network. It is important to shut down unused switch ports.

If Wi-Fi is deployed then it is important to use authentication servers to verify

and log the identity of those logging on. Finally the deployment of anti-virus and

malicious software protection on all systems.

The approach for copying production data to test data includes,

Set up production jobs to copy the data to a common test environment

All PII (Personally Identifiable Information) is modified along with other sensitive data. The PII is replaced with logically correct, but non-personal data.

Remove data that is irrelevant to your test.

Testers or developers can copy this to their individual test environment. They can modify it as per their requirement.

Privacy is the main issue in copy production data. To overcome privacy issues you should look into obfuscated and anonymized test data.

For Anonymization of data two approaches can be used,

BlackList: In this approach, all the data fields are left unchanged. Except those fields specified by the users.

WhiteList: By default, this approach, anonymizes all data fields. Except for a list of fields which are allowed to be copied. A whitelisted field implies that it is okay to copy the data as it is and anonymization is not required.

Also, if you are using production data, you need to be smart about how to source data. Querying the database using SQL script is an effective approach.

Data Leaks Compound the Issue

Even teams that do not use copies of large production datasets should scrutinize their processes and monitor their data more closely. If development and testing teams have any access to production data, there is potential for accidental leakage and outright theft.

Production data can be exposed in a number of ways, from an employee copying it to a laptop for off-site (e.g., home) testing to a team member sending it to someone in another department or another company. It may seem illogical that anyone would do such a thing in today’s threat-laden environment, but I assure you that it happens every day.

Furthermore, spearfishing—whereby a cybercriminal spoofs an email to look legitimate and then embeds malicious links in it—remains one of the most prevalent sources of data breaches. With software teams often connected to servers that might not be as heavily secured as a firm’s core data servers, the outcome if someone clicks on one of those links could be disastrous.

Disgruntled employees are also a leading source of data theft. Organizations must be vigilant about not allowing unbridled access to copies of production data, and they must immediately revoke access to all corporate resources when any employee leaves. Many organizations neglect to create/implement/utilize an off-boarding process to mitigate this threat.

EWE3

If development teams are geographically separated from production and each other, organizations must exercise even greater caution. With remote operations, it is likely that both infrastructure and processes are optimized for remote access, potentially making it easier for criminals to steal cloned production data.

Production Servers Can Post Significant Risk

Production servers are another area where data is often endangered by development and testing activities. For web applications in particular, it is quite common for organizations to develop and test directly on production servers.

Exacerbating the problem, internal development and testing applications are often accessible online yet inadequately secured. If development and testing are being done on a production server, such applications can easily be leveraged by a skilled hacker running sophisticated (and sometimes even basic) scripts to compromise and access the production server.

Also, it is not unusual for teams to archive test sites hosting unmasked production data in public directories that even a moderately tech-savvy person can find. Because these applications are under development, they often have vulnerabilities that hackers can exploit to attain access. Web application content and server-side scripting create additional vulnerabilities, especially in unfinished, not-yet-security-tested projects.

## Statement of Research Problem

## Data breaches are becoming impossible to escape. They occur in every business sector and in companies of all sizes. Juniper Research estimates that the average cost of a single data breach will exceed $150 million by 2020, with the global exposure exceeding $2 trillion by 2019.

Yet, many organizations still struggle to secure their production data effectively. In many cases, business leaders think they are protected, only to discover after a data breach that the opposite was true. For companies that develop, maintain, and/or customize software, either for sale or for internal use, the risk is even greater.

Software teams are under increasing pressure to deliver releases more quickly, yet security budgets haven’t risen to counter the escalating threat. Additionally, accelerated methodologies such as DevOps can create additional pressure, and meeting delivery timetables often takes precedence over security. For example, the U.S. Department of Homeland Security (DHS) estimates that 90 percent of security incidents result from problems with software.

The time has come for organizations to accept responsibility for comprehensive, end-to-end security across the continuous delivery pipeline. In this article, we’ll cover some of the challenges and best practices specific to data security. Hine-based sandbox to contain and examine suspicious programs.

At this point, you may be saying to yourself, “My firm doesn’t take any of these risks. We use best practices to protect data.” Maybe you clone production data for testing but believe your development and testing servers are adequately secured. Or, perhaps you use synthetic data, and only a handful of people can access production data for specific purposes. No matter what safeguards you have taken to protect your data, unless you have adopted end-to-end processes or technologies that mask and secure production data across the entire enterprise, your firm is painting a target on its back.

It has become a cliché: Data security is only as strong as its weakest link. The problem with clichés is that they are true.

The weak link is often the development and testing environment. A company takes every reasonable precaution to button up sensitive databases and then uses actual production data to develop and test new applications. The exposure is exacerbated when the work is outsourced, often to foreign partners.

This once made perfect sense: Develop and test with actual customer records to make sure your apps work when they go into production. Industry and regulatory standards such as PCI and SOX, and best security practice changes all that.

So, companies can either generate test data or mask real data. The challenge is producing data that works without exposing customer information to the world. Most organizations that take this risk seriously are doing one or the other in house, and, of those, most prefer data masking, because it is easier to scale across multiple applications.

"Ninety percent of organizations prefer to mask data," said Noel Yuhanna, a principal analyst at Forrester Research Inc. "Take a copy and mask only the sensitive data, which may be three or four columns at most and be done with it. Data creation is more complex and doesn't always represent actual business scenarios."

It's largely a question of scale, according to Yuhanna. If you need to generate complex test data, with multiple fields and different combinations of fields, chances are you'll only be able to apply it to an application or two. Data generation scales to multiple applications if the test data requirement is simple, say, just a name and address, but that's not typically the case. Consider, he said, the complexity for an insurance company, which has a customer who has been married three times, divorced twice and has five children.

## 1.3 Objectives

## General objective

The main aim of conducting this project was to protect user data from breach by software developers during testing in a sandbox environment.

## Specific objective

Use of one way data encryption at file level

Isolation of servers from the internet disallowing connection to production data and databases

Different Data Security Technologies

Data security technology comes in many shapes and forms and protects data from a growing number of threats. Many of these threats are from external sources, but organizations should also focus their efforts on safeguarding their data from the inside, too. Ways of securing data include:

Data encryption: Data encryption applies a code to every individual piece of data and will not grant access to encrypted data without an authorized key being given

Data masking: Masking specific areas of data can protect it from disclosure to external malicious sources, and also internal personnel who could potentially use the data. For example, the first 12 digits of a credit card number may be masked within a database.

Data erasure: There are times when data that is no longer active or used needs to be erased from all systems. For example, if a customer has requested for their name to be removed from a mailing list, the details should be deleted permanently.

Data resilience: By creating backup copies of data, organizations can recover data should it be erased or corrupted accidentally or stolen during a data breach.

Ways of enhancing data security

Production Data Handling

If production data is used for development and testing, it must be scrubbed or masked.

The data masking/scrubbing solution should be able to deliver data on demand, within minutes, and it should be self-service. This will help team members resist the tendency to circumvent security policies and use production copies.

The organization should also deploy data virtualization in tandem with the data masking/scrubbing solution. Such a solution provides data snapshots that give all team members access to consistent data from the same masked dataset, reducing the overall data footprint and eliminating the temptation to access production data.

Optimally, even with the best data masking solution, the organization should have a comprehensive set of validation procedures.

Production Data Storage

Data on production servers should be protected in transit and at rest with the highest possible grade of encryption (preferably at file-level).

Direct data access should be limited to a handful of trusted individuals, accessible only via multi-factor authentication. Permissions should be assigned using role-based access control.

Development and testing of web applications should always be done on servers isolated from the Internet and should never use or connect to production data and databases.

Web application or website data, code files, and scripts should always be on a separate partition or drive from that of any database, operating system, system files, or logs.

Policies and Procedures

Stringent policies and procedures should be developed and enforced, and the ramifications for accessing and/or distributing copied production data should be clear.

All employees (development, quality assurance, project managers, support personnel, third-party contractors, etc.) should review and sign off on the policies and procedures.

A mechanism for creating a testing activities audit trail with a risk assessment report is strongly recommended. This is a requirement for organizations subject to regulatory control and compliance. However, audit trails and reports are valuable for any firm that becomes the victim of a data breach. Not only can the documentation help identify and eliminate security holes, it can also establish proof of a consistent effort to take security precautions.

What to Look for in a Data Handling Platform

All data masking/scrubbing solutions are not created equally. Orasi finds Delphix to be an attractive choice for its ability to provide end-to-end data masking and data virtualization in a single, unified platform. Some of the features we like in Delphix, that organizational leadership should look for in their own solutions, include:

The capability to profile the data and automatically identify “at-risk” fields, rather than requiring users to define them.

The sophistication to provide data that, while not real, is realistic. Some data (e.g., social security numbers, credit cards, vehicle identification numbers, etc.), has very specific requirements and the solution must address those restrictions.

The confidence that the masked data is sufficiently anonymized, so it cannot be correlated back to the original data.

The complexity to maintain referential integrity so that data keys—as well as connections between lists, data subsets, and other pivot points—are not lost in the masking process.

The assurance of stringent, built-in validation of the resulting masked data.

The opportunity to subset the masked production data in order to minimize the organization’s total data footprint.

## 1.4Scope of the Study

## 1.5 Significance of the study

<https://eyeonquality.com/solving-the-challenge-of-data-security-in-development-and-testing/>

A sandbox is an isolated testing environment that enables users to run programs or execute files without affecting the application, system or platform on which they run. Software developers use sandboxes to test new programming [code](https://whatis.techtarget.com/definition/code). Cybersecurity professionals use sandboxes to test potentially malicious software. Without sandboxing, an application or other system process could have unlimited access to all the user data and system resources on a network.

Sandboxes are also used to safely execute malicious code to avoid harming the device on which the code is running, the network or other connected devices. Using a sandbox to detect malware offers an additional layer of protection against security threats, such as stealthy attacks and exploits that use [zero-day vulnerabilities](https://searchsecurity.techtarget.com/definition/zero-day-vulnerability).

**Importance of sandboxes**

As malware becomes more sophisticated, monitoring suspicious behavior to detect malware has become increasingly difficult. Many threats in recent years have employed advanced [obfuscation](https://searchsoftwarequality.techtarget.com/definition/obfuscation) techniques that can evade detection from endpoint and network security products.

Sandboxing protects an organization's critical infrastructure from suspicious code because it runs in a separate system. It also allows IT to test malicious code in an isolated testing environment to understand how it works within a system as well as more rapidly detect similar malware attacks.

**Uses of sandboxes**

In general, a sandbox is used to test suspicious programs that may contain viruses or other malware, without allowing the software to harm the host devices.

Sandboxing is an important feature of the [Java](https://www.theserverside.com/definition/Java) programming language and development environment, where the sandbox is a program area and set of rules that programmers need to use when creating Java code (called an [applet](https://searchmicroservices.techtarget.com/definition/applet)) that is sent as part of a web page.

A sandbox can also enable a mirrored production environment that an external developer can use to develop an app that uses a web service from the sandbox. This enables third-party developers to validate their code before migrating it to the production environment.

An API sandbox is targeted at API developers and testers. It mimics the characteristics of the production environment to create simulated responses for APIs that reflect the behavior of a real system.

**Java sandboxing**

Java applets are sent automatically to the user's browser as part of the web page transmission and can be executed as soon as they arrive at the browser. Without any other protection, the malicious code could run without restriction and easily do harm; use of a sandbox to isolate the code can help protect against both malicious attacks and harm done by buggy Java programs with unlimited access to memory or operating system services. The sandbox restrictions strictly limit what system resources an applet can request or access.

The Java sandbox comprises the program area and a set of rules that programmers need to use when creating Java code sent with web content. The sandbox restrictions set strict limits on what system resources the applet can request or access. Essentially, the programmer must write code that "plays" only within the sandbox, much as children are allowed to make anything they want to within the confined limits of a real sandbox. The sandbox can be conceived as a small area within your computer where an applet's code can play freely -- but it's not allowed to play anywhere else.

The sandbox is implemented not only by requiring programmers to conform to certain rules but also by providing code checkers. The Java language itself provides features such as automatic memory management, [garbage collection](https://searchstorage.techtarget.com/definition/garbage-collection) and the checking of address ranges in strings and arrays that inherently help to guarantee safe code.

Java's compiled code, known as [bytecode](https://whatis.techtarget.com/definition/bytecode), includes a verifier that guarantees adherence to certain limitations. Java also provides for a local [name space](https://whatis.techtarget.com/definition/namespace) within which code may be restricted. The Java [virtual machine](https://searchservervirtualization.techtarget.com/definition/virtual-machine) (the layer that interprets the Java bytecode for a given computer platform) also mediates access to system resources and ensures that sandbox code is restricted.

In the original sandbox security model, the sandbox code is generally known as *untrusted code*. In later versions of the Java Development Kit (JDK) -- the programmer's development environment -- the sandbox has been made more sophisticated by introducing several levels of trust that the user can specify for sandbox code. The more trust the user allows, the more capability the code has to "play" outside of the sandbox. In the Java Development Kit 1.1 version, the concept of a *signed applet* was introduced. An applet accompanied by a [digital signature](https://searchsecurity.techtarget.com/definition/digital-signature) can contain trusted code that is allowed to execute if the client browser recognizes the signature.

In JDK 2.0, Java provides for assigning different levels of trust to all application code, whether loaded locally or arriving from the Internet. A mechanism exists to define a security policy that screens all code -- whether signed or not -- as it executes.

**Benefits of using a sandbox**

Using a sandbox to test software changes before they go live means there are fewer problems during and after testing because the testing environment is totally separate from the production environment.

Sandboxing is also great for quarantining zero-day threats that exploit unreported vulnerabilities. Although there's no guarantee that sandboxing will stop zero-day threats, it offers an additional layer of security by separating the threats from the rest of the network. When threats and viruses are quarantined, cybersecurity experts can study them to identify patterns, helping to prevent future attacks and identify other network vulnerabilities.

Sandboxing also complements other security programs, including behavior monitoring and virus programs. It offers added protection against certain strains of malware that an antivirus program may not detect.

**Examples of using a sandbox**

Sandboxes can be used to isolate code execution in almost any situation that software code is being executed. Some specific examples of using a sandbox to isolate code execution include:

* Web browsers: A trusted web browser can be run inside a sandbox. Then if a website exploits a vulnerability in that web browser, the damage is limited to the sandbox and minimized.
* Software protection: There are tools that enable users to run software they don't trust in sandboxes so that the software can't access their private data or harm their devices. Because a sandbox appears to be a complete system to the software, the software usually can't detect that it's constrained to a virtual environment.
* Security research: Information security professionals use sandboxes for research or to detect malicious code. For instance, a security tool could visit websites to monitor what files are ultimately changed or it could install and run software.
* Virtualization: A virtual machine is basically a type of sandbox. This approach uses a virtual machine-based sandbox to contain and examine suspicious programs.

**Sandbox applications**

Sandbox applications include:

* Browser plug-in content often depended on using a sandbox to screen content loaded by browser plug-ins, including [Microsoft Silverlight](https://searchwindevelopment.techtarget.com/definition/Silverlight) and Adobe Flash. However, this type of content has been notoriously difficult to keep safe. While it was safer to play a flash game on a web page than to download the game and run it as a standard program, content publishers have largely moved away from such plug-ins in favor of publishing active content using [HTML5](https://searchmicroservices.techtarget.com/definition/HTML5) -- which includes the sandbox attribute to instruct the browser to disable any features that may present security risks.
* PDFs and other documents may include executable code, so Adobe Reader runs PDF files in a sandbox, which stops them from escaping the PDF viewer and interfering with the rest of the computer. Microsoft Office also has a sandbox mode to stop unsafe macros from tampering with a system.
* Mobile apps are generally executed by mobile platforms in sandboxes. Apps for iOS, Android and Windows are prohibited from doing many of the things standard desktop apps can do. For example, to access a user's location, they have to declare permissions. Additionally, the sandbox isolates the applications, preventing them from tampering with each other.

Following are few common places where sandbox technique is used.

1. **Browsers**: Browsers like chrome / firefox implement sandbox technique to run pages / plugins / part of pages like iframes inside a sandbox.
2. **Network software**: Programs like FTP have options to configure ‘Chroot’ for user. In this user will be restricted inside a directory.
3. **Online Judges**: Online Judges like SPOJ run each program in a sandbox environment.
4. **Virtual Machine**: These emulate entire host os inside a sandbox.
5. **Linux Chroot**: Linux systems have a command chroot (change root) where the user can change root from / to some subfolder like /mnt/new/root and thus all programs run afterwards will treat /mnt/new/root as /. Thus all their action are restricted inside the subfolder.

<https://searchsecurity.techtarget.com/definition/sandbox>

Data breaches are becoming impossible to escape. They occur in every business sector and in companies of all sizes. Juniper Research estimates that the average cost of a single data breach will exceed $150 million by 2020, with the global exposure exceeding $2 trillion by 2019.

Yet, many organizations still struggle to secure their production data effectively. In many cases, business leaders think they are protected, only to discover after a data breach that the opposite was true. For companies that develop, maintain, and/or customize software, either for sale or for internal use, the risk is even greater.

Software teams are under increasing pressure to deliver releases more quickly, yet security budgets haven’t risen to counter the escalating threat. Additionally, accelerated methodologies such as DevOps can create additional pressure, and meeting delivery timetables often takes precedence over security. For example, the U.S. Department of Homeland Security (DHS) estimates that 90 percent of security incidents result from problems with software.

The time has come for organizations to accept responsibility for comprehensive, end-to-end security across the continuous delivery pipeline. In this article, we’ll cover some of the challenges and best practices specific to data security.

The Development Data Challenge

Delphix, a data virtualization firm, commissioned a report in 2015 demonstrating that copies of production data used in development and testing represent one of the greatest data breach risks in an enterprise. Furthermore, 62% of DevOps leaders say full data access for non-production environments is a requirement for success. Moreover, as many as 80% of firms use (and often store) production data at some point in their development and testing activities.

This approach is extremely dangerous, since development and testing environments are not often within the organization’s strongest layers of security. Instead of the data residing in tightly monitored systems with strong access restrictions and a small group of authorized users, it is accessible to a larger number of people with minimal security clearance.

In some cases, the data is being shared with disparate or even outsourced teams. As a result, sensitive data is crossing networks that may not be adequately encrypted or is leaving the confines of the U.S.

Data Leaks Compound the Issue

Even teams that do not use copies of large production datasets should scrutinize their processes and monitor their data more closely. If development and testing teams have any access to production data, there is potential for accidental leakage and outright theft.

Production data can be exposed in a number of ways, from an employee copying it to a laptop for off-site (e.g., home) testing to a team member sending it to someone in another department or another company. It may seem illogical that anyone would do such a thing in today’s threat-laden environment, but I assure you that it happens every day.

Furthermore, spearfishing—whereby a cybercriminal spoofs an email to look legitimate and then embeds malicious links in it—remains one of the most prevalent sources of data breaches. With software teams often connected to servers that might not be as heavily secured as a firm’s core data servers, the outcome if someone clicks on one of those links could be disastrous.

Disgruntled employees are also a leading source of data theft. Organizations must be vigilant about not allowing unbridled access to copies of production data, and they must immediately revoke access to all corporate resources when any employee leaves. Many organizations neglect to create/implement/utilize an off-boarding process to mitigate this threat.

If development teams are geographically separated from production and each other, organizations must exercise even greater caution. With remote operations, it is likely that both infrastructure and processes are optimized for remote access, potentially making it easier for criminals to steal cloned production data.

Production Servers Can Post Significant Risk

Production servers are another area where data is often endangered by development and testing activities. For web applications in particular, it is quite common for organizations to develop and test directly on production servers.

Exacerbating the problem, internal development and testing applications are often accessible online yet inadequately secured. If development and testing are being done on a production server, such applications can easily be leveraged by a skilled hacker running sophisticated (and sometimes even basic) scripts to compromise and access the production server.

Also, it is not unusual for teams to archive test sites hosting unmasked production data in public directories that even a moderately tech-savvy person can find. Because these applications are under development, they often have vulnerabilities that hackers can exploit to attain access. Web application content and server-side scripting create additional vulnerabilities, especially in unfinished, not-yet-security-tested projects.

The Problem Is Universal

At this point, you may be saying to yourself, “My firm doesn’t take any of these risks. We use best practices to protect data.” Maybe you clone production data for testing but believe your development and testing servers are adequately secured. Or, perhaps you use synthetic data, and only a handful of people can access production data for specific purposes. No matter what safeguards you have taken to protect your data, unless you have adopted end-to-end processes or technologies that mask and secure production data across the entire enterprise, your firm is painting a target on its back.

As an aside, synthetic data presents an additional challenge not related to security. Being a simulation, it may not adequately replicate what exists in production. Since synthetic data is a model, it may not account for scenarios outside the vision of its creator.

The hard truth for software teams—and the companies that develop, maintain, and/or customize software—is that the current state of software development and testing encourages security problems.

Whether due to competitive pressure or demand from users (internal or external), application development cycles are being shortened, yet resources and budgets are not increasing at the same pace. This is especially true for mobile apps, which are becoming requisite in both the consumer and enterprise arenas.

Gartner estimates that market demand for mobile application development services is growing at least five times faster than internal IT organizations’ capacity to deliver them. It is inevitable that these pressures will cause project leaders and their teams to take shortcuts, circumvent security protocols, and adopt “just-this-once” mentalities.

Gears cloudData Security Best Practices

Now that I have your attention, let’s talk about solving the problem. The following are some recommendations for best practices. This list is intended as a starting point—it is not all-encompassing, and every organization’s needs are different.

Conclusion

With organizations increasingly being subjected to compliance mandates such as the Federal Information Security Management Act of 2002 (FISMA), Sarbanes-Oxley (SOX), the Health Insurance Portability and Accountability Act (HIPAA), and the Payment Card Industry Data Security Standard (PCI-DSS), the need to protect production data at all levels will only increase. New standards such as the EU’s General Data Protection Regulation (GDPR) will add pressure for firms conducting business with major European partners.

Even for firms not subject to any of these mandates, the realities of data breaches can be financially catastrophic. Taking a forward-thinking approach to data security now can minimize the risk of significant disruption—and potential disaster—in the future.

**Data masking hides information from testers**