Effective Security Assessments and Testing

David Culbreth, Adan Guadarrama, Ayad Barsoum*
St. Mary's University, San Antonio, TX (USA)
Emails: aguadarrama@mail.stmarytx.edu, dculbreth@mail.stmarytx.edu, abarsoum@stmarytx.edu

Abstract — Companies providing technology-driven services are held to the high standard of full availability, integrity, and confidentiality. Achieving even near-perfect availability is an increasingly daunting task, even for these companies with seemingly limitless resources. In order to approach this very challenging goal, strategies must be implemented to ensure that changes and improvements to the provided services do not leave the currently functioning environment vulnerable to attacks or introduce new issues. Systems and processes must be evaluated to ensure their efficient and effective operation. Administrative security controls must be audited to ensure the proper implementation of policies and procedures. A failure to properly evaluate the programs and procedures leaves an organization at risk for a data incident or an attack on the organization's assets. This paper covers some of the most important elements of security assessments and testing.

Keywords: Audit, Test Strategy, Change Control

I. Introduction

In recent years, the abuse and misuse of technical resources and information has become a prevalent problem. While technical capabilities have exponentially grown since the invention of the computer, the ability of companies to properly manage these resources has not kept up the pace. It is now a daily occurrence to hear reports about where to not save credit card information, how identity theft can come from phone calls pretending to be banks, and how emails can be used to send/receive false information. As a result of this, security awareness has become increasingly important to companies [6].

In 2013, Target suffered a data breach leaking millions of customers' personal and financial information [7]. However, this data was not leaked due to compromised internal systems. The data breach happened because of a third-party vendor that takes care of the ventilation. As a result of the data breach, thousands of shoppers at target were now at risk of becoming victims of identity theft. The settlement from this data breach resulted in Target having to pay millions of dollars to settle lawsuits that started coming their way. If Target would have successfully audited their third-party vendor, they may have saved themselves all the millions they lost in the settlement.

II. Establishing Product and System Definitions

Documenting the proper behavior of an application is a necessary step for secure and reliable development in every environment. This documentation should include the system behavior and configurations, preferably as a process diagram or as human-readable documentation. The inclusion of testing from the beginning of any development has also proven to be a useful supplement to the user documentation. Using strategies like *test driven development* [4] or *behavior driven development* [5] provide the opportunity for the project management to define their requirements in a formalized and reproducible fashion, such that the functionality, once developed, is known to work to the specifications and requirements initially produced. At IBM, a development team was able to reduce their defect rate by 50 percent by implementing test-driven development strategies in their retail store solutions [1]. In addition to the dramatic decrease of defects, test-driven development produces a reusable test asset, which is an invaluable tool come the time for regression testing. Product and system definitions provide critical guidance to the developers and users during and after the development period. Test driven development enhances this documentation with functional proof that the product works as intended.

The process by which a change is deployed is just as important as the testing that happens before it. Similarly, the deployment process must also be thoroughly tested and documented, as deploying a change to an application or an infrastructure is perhaps the riskiest portion of the process, as it adds an element of variation to an environment that is likely already functioning. Every step of the deployment process needs to be crafted with care and scrutiny. The tools used to develop the code must be reviewed to ensure proper code is generated. The repositories that store and track the source code must be vetted to ensure the code is not at risk for being leaked to prying eyes. Finally, when the package is ultimately deployed, the configuration must be deployed

through a system known and tested to reliably deliver the correct results [2]. Mature change management will consider many aspects of each individual process. Each change should document its intended effect, and the tests run to ensure that the change executes its purpose with no unintended side-effects. Additionally, the precise actions to deploy the change will be vetted, checking whether this kind of change has failed before, and that there are no other conflicting changes happening at the same time. This list can contain a seemingly endless set of records to document, but a final important piece is the backout plan: what to do when the change goes south, despite your best efforts. Ensuring a functional environment is most frequently more important than implementing the change to it.

III. Auditing Technical Controls

Even once an application or process has been deemed secure or bug-free enough to release, the system as a whole must also be tested, including its surrounding physical environment, its users, its technical assets and processes, and the interaction between all of these. A system, facility, or application can appear to be secure, but if the users' actions leave the system vulnerable, then all is for naught. To combat this possibility, periodic audits of the security performance must be evaluated, but not every audit or test is equal.

First, white box testing [8-10] provides the testers with the internal implementations of the software and systems. White box testing is useful for finding errors in hidden code by removing extra lines of code and maximizing code coverage. However, it is expensive to implement white box testing, and the nature of the tests can leave many code branches untested. The relevant techniques for white box testing involve control flow testing, branch testing, basis path testing, data flow testing, and loop testing. White box testing is typically done by the developing team, or a QA team closely familiar with the developers' work.

Black box testing [8, 9, 11] takes the opposite approach, providing the testers with no knowledge of the system except the bounds of the test itself. Black box testing is useful for efficiently testing large segments of code from a simple perspective and developing test cases very efficiently. Black box testing strategies include equivalence partitioning, boundary value analysis, fuzzing, cause-effect graphing, orthogonal array testing, all pair testing, and state transition testing [3, 12].

Grey box testing finds a middle ground between black and white, providing general knowledge of the internal operations of the system. Grey box testing carries many of the benefits of both black and white testing, allowing for many specific elements to be tested knowing the general purpose of the algorithms under scrutiny, while still writing minimal amounts of efficient tests. The techniques used here include Architecture models, unified model diagrams, and finite state machines.

While black, white, and grey box testing can efficiently determine the effectiveness of the code, internal software testing does not cover every aspect of the vulnerabilities that a CyberSystem can bare.

Think of this for a second, a company created a website application that takes user credit card information to purchase a product. How does the user know that once the credit card information entered for the purchase is safe? Unfortunately, that is the case when it comes to users feeling unsafe when it comes to their information because of the rise of data breaches like Target. For this reason, and many other reasons unmentioned, penetration testing is very important when it comes to Cyber protection. If me as an attacker can find a way to mess with the database from the web application produced, that will be identified as an SQL Injection attack [13-16].

Vulnerability testing is conducted as a series of steps that have one general purpose: Identify a way to get in the system, and how to address each vulnerability that is identified. As mentioned earlier, with an SQL Injection attack those types of threats are identified and addressed by changing the way that information is taken in to the system. As soon as this threat is fixed, it is retested through the penetration testing phase. This process gets repeated as many times as needed to help ensure that the safest product being developed is out there.

Once these tests are completed, we move into integration testing [17-19]. The purpose of the integration testing is to see how the application works with a majority of components integrated together. From there, we can see how the application interacts altogether with the network and from there address any issues that may arise.

IV. Testing Admin Controls

Another aspect that needs to be tested besides code and application practices on a network is the administrative perspective. If left unchecked and untested, administrative misconfiguration could provide an attacker a back door into the network expose it, do what he wants to a network, and the attack can almost go unnoticed due to the lack of administrative scrutiny. This can potentially be very dangerous, as this creates the opportunity for an attack as serious as the Target Data breach. While the development

teams are responsible for ensuring the secure internal processes are created, it is up to the administrative leaders to ensure that the policies surrounding protecting the systems are implemented properly.

The first such administrative control are user permissions. When initially hired or introduced to the system, users should be granted access only to the necessary systems. As the user progresses through his position, his assigned accesses must change and grow with his function within the corporation. When these users are transferred between departments or let go, this change should trigger another audit to ensure that the permissions are still appropriate for that user's position. These audits protect both the user and the company, as the user cannot negatively affect systems he should not be using, and his responsibility is lessened to solely protecting the systems for which he is responsible, and no more. Beyond the triggers on these boundary conditions, the user permissions should be audited at a regular interval, to be sure that his permissions were not inadvertently expended or constrained beyond the necessary limit. This rigorous inspection of the permissions that each user has is beneficial for everyone: auditors, users, and administrators, as it simplifies the already complex world of user permission sets.

The next thing to consider is how the company backs up data when unexpected events happen. For example, consider a Service Desk Analyst, a call comes in and the issue is identified as a network connectivity issue. As he is trying to fix the issue, he overhears another call that has a similar issue like the one he is dealing with currently. After further analysis into the issue, a pattern emerges and look up in the call que that there are over 50 calls waiting to be addressed. This normally means that there is a massive network outage and all of those calls are being affected by it.

To be prepared for this type of issue most network teams have a network share drive set up on their network. This means that if an excel spreadsheet is created inside a share network file folder, network teams that have access to that same folder can access the excel spreadsheet at any given time. In case of a network outage, you might not be able to access the network file at that precise moment in time. If you have the excel sheet open at the time, you run a risk of losing all unsaved work on that excel spreadsheet or the network may not recognize the changes you made and will have only the original contents of the file before any edits were made. When these types of issues arise, sometimes it is best practice to have a copy of that shared file on the desktop.

Each company must assess how to address different situations that may arise and handle them accordingly. For the above scenario, this may be a low-level occurrence, but it can be detrimental for programs that run on the network and depend on database connection. Especially in a bank environment, most applications need to read stored information. If they are working on the information during a network outage and that type of scenario is not addressed, you can possibly face all that data being lost in the first place.

V. Identifying Test Personnel

One of the most complexing issues companies face is who conducts the audits. When it comes to audits, different things need to be considered. There are recommendations to have audits be conducted by external, third-party, vendors. The biggest reason given is that with external audits there is no bias and that would make the audit be completely accurate for identifying threats.

Depending on how the company culture is, there can be a real upside to having companies put together a team and go through each aspect of the network together to confirm how safe and secure it really is. There is one negative aspect to this. Unfortunately, some companies have environments to where if a bad report comes out about their specific department, the employees are the ones who pay the price.

A logical thing to do as an employee would be what we are ethically bound to do, report the issue and address it. An internal employee may not report all issues because they may feel that this could jeopardize their career due to the fact that their boss has not created an environment where that employee can feel safe. Due to this, some companies are beginning to feel that going third party is the way to go to ensure that the most accurate report can be possible.

With external audits, yes it would be the more likely best way to get an unbiased report. However, external audits can frequently be very expensive, so these are usually only utilized in small organizations or locations that are at risk of bias on the part of the individuals that would perform the audit. One guide on how to think about it is, the more you spend is the more you can almost ensure that the system is going to be very secure. Sometimes the only issue is that when companies value money over cost vs trying to come up with the money to have a secure system, you leave yourself open to potentially costing yourself millions of dollars. Then, you would reinvest into trying to make the system more secure to rebuild the reputation.

VI. Audit Reporting

The tests and audits are not simply completed for the sake of running tests and audits. Each test or audit begins

with explicit intent, whether it be for routine health checks, or because the government found a serious legal issue. Communicating the results of the audit is perhaps as important as the audit itself, as an improperly communicated report may lead the management to take yet another inappropriate, or potentially damaging action.

Any audit report should begin with a proper executive summary, which gives a brief summary of why the audit is important, what was found, and what needs to be done about it. This section should include key talking points and vulnerabilities. When presented to the other portions of the company, the executive summary will provide the key talking points on why your recommendations are so important.

From the executive summary, the report should naturally continue into its background. The background will highlight specific pieces of legislation, particular issues that were experienced, and other reasons why this audit or test was performed. This context will be important for deciding whether and how quickly the recommendations are implemented into company policy and program.

The methodology will then describe the process taken to provide the results of the test. Whether it be through automated scripts, by compiling an excel document summarizing the current state, or by manually running through a checklist, the methodology is important for justifying why the results are valid. In this case, the results are just as important as the process taken to get them.

After describing the means of the test, the findings can finally be reported in full detail. This does not mean to dump the full data set acquired during testing onto the page, but a reasonable summary of the data with a brief analysis turns these results into truly useful information.

Finally, provide recommendations to the readers of the report. As the executor of the test or audit, the writer has typically seen one or more ways to improve the processes being tested. This is the opportunity to provide the advice rooted in technical expertise before the management tries to imagine some crazy solution that doesn't actually fix the problem. If possible, provide recommendations on ways to correct each vulnerability or error encountered during the audit. With no instruction on how to fix the problems, the creators of the problems are liable to commit the same atrocities yet again.

VII. Conclusion

When it comes to doing security assessments, nothing should be overlooked. They are the opportunity to lay everything bare and identify the issues. With the proper methods in place, you can ensure that all systems are up to date and offer the best protection possible. It would also be best to not overlook even the smallest of security flaws because the smallest components of a system can be exploited and used against you. If there are proper security protocols in place, you normally would never encounter a security breach or be at risk for having information stolen. Sometimes cost is a major factor for determining how much to spend on security, but if your goal is to create the most secure system, something has to give. It almost seems like a tough call to make at times, but it is a necessary one to make.

References

[1] Keith Jarvis, Jason Milletary. "Inside a Targeted Point-of-Sale Data Breach" Dell SecureWorks Counter Threat Unit Intelligence. January 2014

[2] E. Maximilien, Laurie Williams. "Assessing Test-Driven Development at IBM" IEEE. May 2003

[3] Mohd. Ehmer Khan, Farmeena Khan. A Comparative Study of White Box, Black Box and Grey Box Testing Techniques. IJACSA Vol. 3, No.6, 2012

[4] Astels, Dave. Test driven development: A practical guide. Prentice Hall Professional Technical Reference, 2003.

[5] Solis, Carlos, and Xiaofeng Wang. "A study of the characteristics of behaviour driven development." In 2011 37th EUROMICRO Conference on Software Engineering and Advanced Applications, pp. 383-387. IEEE, 2011.

[6] Siponen, Mikko T. "A conceptual foundation for organizational information security awareness." Information Management & Computer Security 8, no. 1 (2000): 31-41.

[7] Manworren, Nathan, Joshua Letwat, and Olivia Daily. "Why you should care about the Target data breach." Business Horizons 59, no. 3 (2016): 257-266.

[8] Nidhra, Srinivas, and Jagruthi Dondeti. "Black box and white box testing techniques-a literature review." International Journal of Embedded Systems and Applications (IJESA) 2, no. 2 (2012): 29-50.

[9] Khan, Mohd Ehmer, and Farmeena Khan. "A comparative study of white box, black box and grey box testing techniques." Int. J. Adv. Comput. Sci. Appl 3, no. 6 (2012).

[10] Khan, Mohd Ehmer. "Different approaches to white box testing technique for finding errors." International Journal of Software Engineering and Its Applications 5, no. 3 (2011): 1-14.

[11] Edwards, Stephen H. "A framework for practical, automated black-box testing of component-based software." Software Testing, Verification and Reliability 11, no. 2 (2001): 97-111.

[12] Gilfix, Michael A., and Rhys D. Ulerich. "Method for testing branch execution and state transition logic in session initiation protocol application modular components." U.S. Patent 7,499,405, issued March 3, 2009.

[13] Halfond, William G., Jeremy Viegas, and Alessandro Orso. "A classification of SQL-injection attacks and countermeasures." In Proceedings of the IEEE International Symposium on Secure Software Engineering, vol. 1, pp. 13-15. IEEE, 2006.

[14] Boyd, Stephen W., and Angelos D. Keromytis. "SQLrand: Preventing SQL injection attacks." In International Conference on Applied Cryptography and Network Security, pp. 292-302. Springer, Berlin, Heidelberg, 2004.
[15] Clarke-Salt, Justin. SQL injection attacks and defense. Elsevier, 2009.
[16] Wei, Kei, Muthusrinivasan Muthuprasanna, and Suraj Kothari. "Preventing SQL injection attacks in stored procedures." In Australian Software Engineering Conference (ASWEC'06), pp. 8-pp. IEEE, 2006.
[17] Jorgensen, Paul C., and Carl Erickson. "Object-oriented integration testing." Communications of the ACM 37, no. 9 (1994): 30-39.
[18] Hartmann, Jean, Claudio Imoberdorf, and Michael Meisinger. "UML-based integration testing." In ACM SIGSOFT Software Engineering Notes, vol. 25, no. 5, pp. 60-70. ACM, 2000.
[19] Wu, Ye, Mei-Hwa Chen, and Jeff Offutt. "UML-based integration testing for component-based software." In International Conference on COTS-Based Software Systems, pp. 251-260. Springer, Berlin, Heidelberg, 2003.
[20] Mangipudi, Prasad. "Method and apparatus for archiving data during unexpected power loss." U.S. Patent 7,954,006, issued May 31, 2011.

[21] Kwon, Min Cheol, Woon Hyug Jee, Dong Jun Shin, and K. I. M. Shine. "Nonvolatile memory system and related method of preserving stored data during power interruption." U.S. Patent 8,554,990, issued October 8, 2013.

David Culbreth is a graduate student at St. Mary's University, San Antonio, Texas enrolled in the MS Cybersecurity degree program. He received his B.Sc. in Computer Engineering and his B.A. in Mathematics from St. Mary's in 2018. David has been a Software Developer at USAA for one year, supporting the company in its needs for physical security applications.

Adan Guadarrama is a graduate student at St. Mary's University, San Antonio, Texas enrolled in the MS Cybersecurity degree program. He received his B.Sc. in Computer Science from St. Mary's in 2019. Adan has been a Help Desk Technician both at USAA and St Mary's University. Adan has focused these last few years studying the purposes of secure programming practices.

Ayad Barsoum is an Associate Professor in Computer Science Department at St.Mary's University, San Antonio, Texas. He is the Graduate Program Director of MS in Cybersecurity. Dr. Barsoum received his Ph.D. degree from the Department of Electrical and Computer Engineering at the University of Waterloo (UW), Ontario, Canada in 2013. He is a member of the Centre for Applied Cryptographic Research at UW.

He received his B.Sc. and M.Sc. degrees in Computer Science from Ain Shams University, Cairo, Egypt, in 2000 and 2004, respectively.

At the University of Waterloo, Barsoum has received the Graduate Research Studentship, the International Doctoral Award, and the University of Waterloo Graduate Scholarship. Dr. Barsoum has received "Amazon Web Services in Education Faculty Grant" for funding his research and teaching through using Amazon cloud infrastructure