WHITE PAPER

Contrast Security Product Applicability Guide for PCI DSS, PA-DSS, and PCI-SSF

Applicability to Assist Customers with PCI DSS Version 3.2.1 Deployments

Executive Overview

Contrast Security (Contrast) is a provider of a software security platform that pinpoints vulnerabilities in custom software, analyzes the security of open source libraries, and prevents software vulnerabilities from being exploited. Contrast's patented instrumentation-based approach enables highly accurate security testing and protection to be embedded across an entire application portfolio easily, without requiring security experts to run scans, tailor rules, or triage the results.

Contrast has engaged Coalfire, a respected Payment Card Industry (PCI) and Payment Application (PA) Qualified Security Assessor Company (QSAC), to conduct an independent technical evaluation of Contrast's integrated application security platform consisting of two products, Contrast Assess and Contrast Protect. Contrast Assess automates security expertise to find security vulnerabilities during development and Quality Assurance (QA). Contrast Protect guards against exploits in QA and production. The purpose of this product applicability guide is to identify alignment of Contrast Assess and Contrast Protect capabilities to the Payment Card Industry Data Security Standard version 3.2.1 (PCI DSS v3.2.1) to assist payment entities wishing to use this solution in a manner that supports compliance with PCI DSS v.3.2.1, in addition to payment application vendors, developers, security, and operations teams to address requirements of the Payment Application Data Security Standard (PA-DSS v3.2) and the new Payment Card Industry Software Security Framework Secure Software Requirements and Assessment Procedures version 1.0 also referred to as the PCI Secure Software Standard and the Secure Software Lifecycle (Secure SLC) Requirements and Assessment Procedures v1.0.

Contrast Assess and Contrast Protect are provided in a single technology that integrates continuous application security, vulnerability assessment, software composition analysis, and attack monitoring and protection as detailed below.

- **Contrast Assess** provides "interactive application security testing" (IAST) to detect vulnerabilities in both custom code and open source libraries. Contrast Assess can seamlessly detect vulnerabilities very early and throughout the software development lifecycle (SDLC).
- **Contrast Protect** provides "runtime application self-protection" (RASP). Contrast Protect offers continuous monitoring and runtime exploit prevention in production.

This product applicability guide may be useful to any payment entity and payment application vendor who need continuous automated security testing and security protection for their application. Contrast Assess and Contrast Protect can provide continuous automated security testing and protection to improve application security throughout the software's lifecycle. This paper discusses relevant PCI DSS v3.2.1, PA-DSS v3.2, and PCI Software Security Framework v1.0 requirements that may be addressable or supported by capabilities provided by Contrast Assess and Contrast Protect. It is understood that the payment entity must address every PCI DSS v.3.2.1 requirement. Likewise, payment software vendors wishing to validate payment software under PA-DSS v3.2 or the PCI Software Security Framework v1.0 would be required to adhere to all requirements of the applicable standard.

COALFIRE OPINION

Coalfire has determined that Contrast Assess and Contrast Protect can be a valuable tool for helping organizations identify, classify, and address vulnerabilities and protect their software throughout the SDLC. Contrast Assess can be useful to assist with the development of secure code by identifying issues earlier in the lifecycle and offering remediation paths. Contrast Protect allows supported software to be protected with greater fidelity than what is offered by traditional software security approaches alone. Contrast Assess and Contrast Protect may be used to replace some of the traditional approaches to assessing and protecting applications.

Introducing Contrast Security

Contrast's integrated application security platform allows software to be self-assessing and self-protecting through the entire lifecycle. Contrast Assess helps development teams following Waterfall, Agile, or DevOps to ship secure code faster and protect applications from attacks, while providing visibility to unlock threat intelligence. Figure 1 illustrates how the Contrast platform's sensors are integrated with the application and its libraries to identify vulnerabilities in the application through all layers.

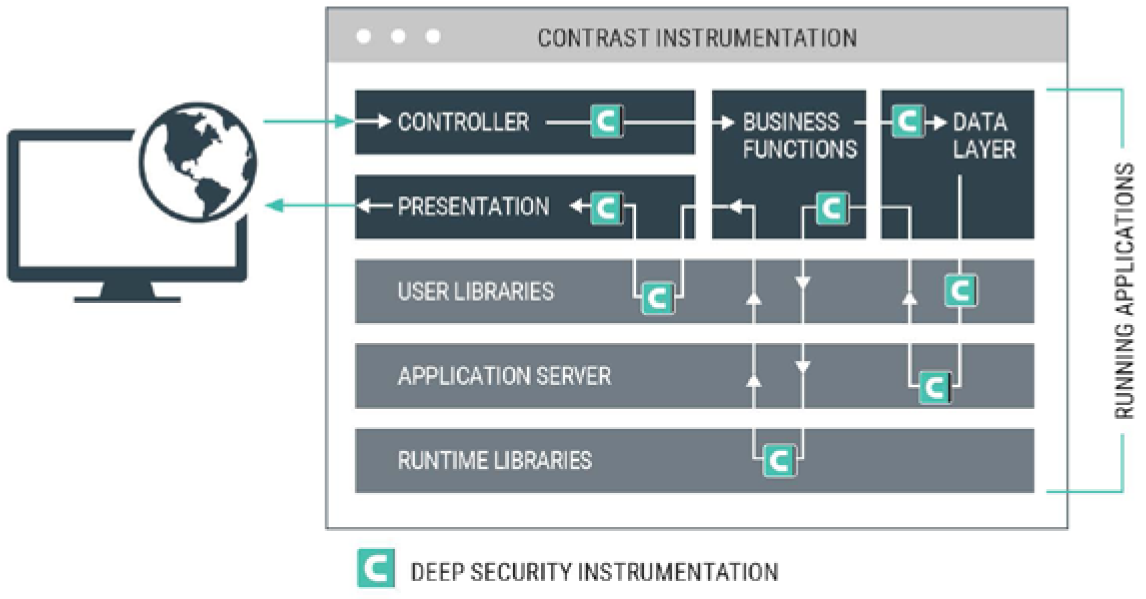


Figure 1 - Integrated Security through Contrast Instrumentation

To monitor the application, the Contrast Security platform language-specific agents use the appropriate mechanism to instrument each runtime. Because the Contrast Security platform directly measures the security of running applications, all the complexity of different development processes, build pipelines, repositories, and other details of how software is built, tested, and deployed disappear. The Contrast Security platform does not require access to the source code and can run wherever the application runs. Even obfuscated code can be easily assessed and protected. The Contrast Security platform instrumentation simply takes a snapshot of the application state inside security critical methods whenever they are invoked. The stream of events from these sensors creates a trace that can be compared against a set of allowed and disallowed application behaviors. This approach ensures both accuracy and that Contrast Assess and Contrast Protect cannot ever be bypassed.

Contrast sensors continually assess the code for vulnerabilities and do not require a specified scan-period, human expert, or security focused testing as is common for static and dynamic scanning tools. Therefore, users cannot accidentally scan the wrong code or partial code because the code is running. The code is assessed as the application is exercised in runtime. In this way, vulnerabilities can be detected much earlier in the process, during functional testing (unit testing, integration testing, and so forth), without the need of security expertise, rather than at specific lifecycle gateways dedicated to security testing. The Contrast Security platform sensors "Assess" continuously during development and test and "Protect" when the code is deployed to production.

This continuous assessment of the code does not require interaction by the developers or security team members to detect and report on vulnerabilities. When Contrast Assess is in place, all usage becomes a security test because the Contrast Assess sensors are a part of the application. Even though the team may not be looking for security flaws during functional test cycles, Contrast Assess will report on detected vulnerabilities and security issues. Typically, early application code testing is primarily concerned with achieving functional and performance targets, rather than identifying or satisfying security targets. Including Contrast Assess as part of the application allows for security to be incorporated much earlier in the lifecycle, where identified security flaws can be remediated sooner rather than later. Contrast Assess will identify vulnerabilities in the entirety of the software application – in both the custom application code and open source libraries.

When security is incorporated into the application and applications become self-assessing and self-protecting, the fidelity of findings is improved. The Contrast Security platform sensors see everything that an application does and is not concerned with what an application does not do. External testing and monitoring tools, without context, will not understand the application's actual attack or usage types. Dynamic application security testing (DAST) may attempt a variety of attacks of a type that are not relevant to the application being scanned. A Web Application Firewall (WAF) may protect applications against exploits that are not applicable to the application, such as a SQL injection attack against an application that uses NoSQL. The resulting outcomes are often a flurry of information, wasted effort, and possibly broken applications. This abundance of information can make it difficult to prioritize issues for remediation. Security teams are also often tasked with continuous tuning of the tooling to better align with their systems to attempt to produce more reliable results.

Contrast Assess and Contrast Protect integration into the application lends itself to improved visibility and understanding into what the application is doing. This awareness allows for findings of vulnerabilities and detection of attacks to be evaluated based on context. Figure 2 illustrates how the Contrast Security platform is aware of the applications layers, the ingress and egress points of the application, and the flow of data through the application. It also illustrates awareness of methods being used by the application. From the front end to the back end, Contrast has a better awareness of the totality of the application, because it is part of the application.

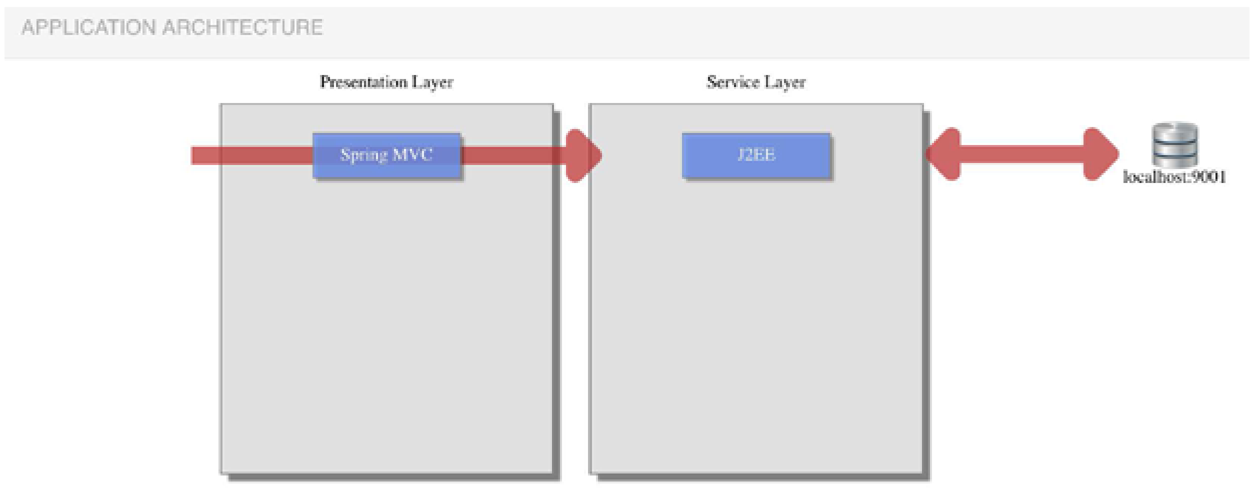


Figure 2 – Contrast Visibility into the Application Data Flows

The visibility to the application includes information for how the application communicates or uses the backend databases. This is especially important for understanding context, as it does not assume that all applications are alike. There are multiple possibilities for what could be used to support the back end of an application, ranging from traditional SQL databases to file-based or NoSQL databases. Understanding these data connections and the type of databases supported is important to understanding if a vulnerability could truly be exploited by an attacker. As an example, on the front end, a SQL injection vulnerability may be identified or traffic passing through a WAF may be identified as a SQL injection attack; however, if the application uses NoSQL on the back end, the attempted injections could never succeed.

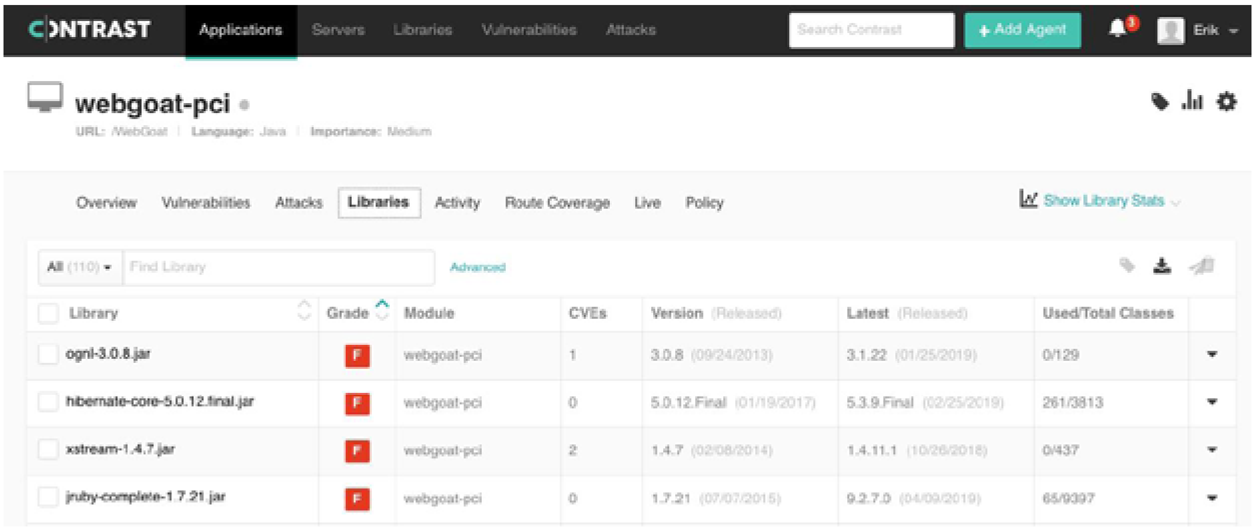
Contrast Assess and Contrast Protect can observe the activities of the application continuously and in real-time. Alternatively, dynamic code analysis only sees a vulnerability if a response or result comes back to the client. Similarly, with a WAF, an attack attempt is only identified based on incoming data. However, the WAF is not aware of what is occurring with the application on the back end.

Contrast Protect is well equipped to discover and effectively block attacks because it is a part of the deployed application. For instance, Contrast Protect knows the proper method of decoding to be able to better identify things like deserialization attacks. If the security solution is watching the front door, rather than participating as part of the application code, it would not necessarily be able to understand the language that would be used to deserialize the payload or how the results of this payload would or would not impact the application. Even if the WAF was educated about the language that is used to deserialize the payload, it still may not be able to make the right determination because it lacks sufficient information to understand the hierarchy needed to differentiate between a legitimate deserialization and an attack.

Open source library code often makes up a larger percentage of an organization's total application code, as much as 79% on average, and 96% of applications include some form of open source software. The use of open source software speeds up application development and helps propel compelling business applications to market faster. However, the use of open source software also adds the potential of greater risk for an organization's applications. This requires better management, monitoring, and visibility of open source software use to promote more secure applications.

Contrast Assess can view and perform software composition analysis of all third-party libraries used by the application. The inventory of libraries includes an understanding of how the library is used by the application, what classes from the library are used, and the version of the library that is used. This inventory of libraries is compared against known vulnerabilities to display common vulnerabilities and exposures (CVEs) that exist for the library and provide guidance for libraries that should be upgraded and to what version they should be upgraded. The contextual awareness of how the library is used helps to grade or classify the vulnerability according to the exploitability of the vulnerability as it pertains to the running application.

Figure 3 depicts an example of how Contrast Assess identifies open source libraries in use by an application. It provides a grade for each library according to the risk the library presents. The list of libraries includes CVEs that exist for each library, the current version of the library that is used, and the latest version of the library that is available. Also, Contrast Assess has the ability to assess what classes from the library are used by the application. This information is also useful in determining the degree of exploitability the library presents to the application.



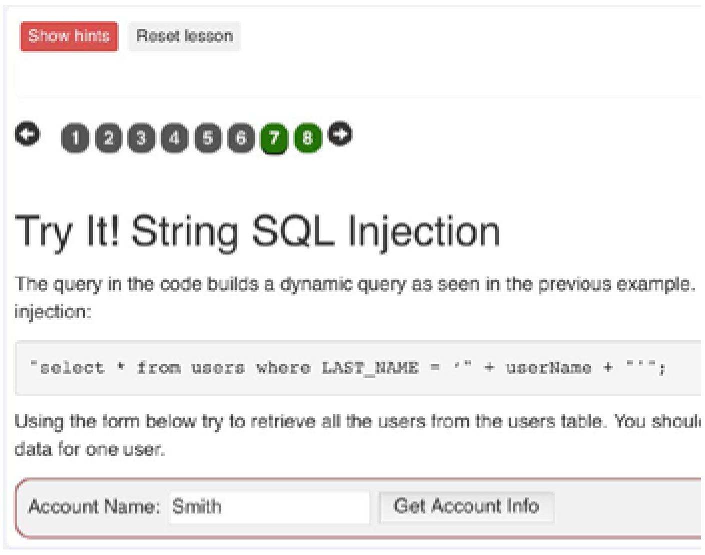
Library	Grade	Module	CVEs	Version (Released)	Latest (Released)	Used/Total Classes
ognl-3.0.8.jar	F	webgoat-pci	1	3.0.8 (09/24/2013)	3.1.22 (01/25/2019)	0/129
hibernate-core-5.0.12.final.jar	F	webgoat-pci	0	5.0.12.Final (01/19/2017)	5.3.9.Final (02/25/2019)	261/3813
xstream-1.4.7.jar	F	webgoat-pci	2	1.4.7 (02/08/2014)	1.4.11.1 (10/26/2018)	0/437
jruby-complete-1.7.21.jar	F	webgoat-pci	0	1.7.21 (07/07/2015)	9.2.7.0 (04/09/2019)	65/9397

Figure 3 – Contrast Open Source Software Detection

The following illustrates the detection of a vulnerability through normal functional testing of an application. In this example, WebGoat from the Open Web Application Security Project (OWASP) was used for demonstrating Contrast Assess and Contrast Protect capabilities. The WebGoat project provides a deliberately insecure web application maintained by OWASP designed to teach web application security lessons. A preceding awareness of vulnerabilities in WebGoat allows for testing to be performed against expected results.

Contrast Assess and Contrast Protect were integrated with WebGoat and testing was performed to determine whether Contrast Assess could detect and report on the vulnerability and whether with protection policies in place Contrast Protect could prevent an attack from succeeding. This example uses SQL injection vulnerability and attacks for demonstrating capabilities. The Contrast Security Platform ships with additional built-in protection rules including Bean Classloader Overwriting, Command Injection, Cross Site Request Forgery (CSRF), Cross-Site Scripting (XSS), Expression Language Injection, Method Tampering, Padding Oracle, Path Traversal/Local File Include, Regular Expression DoS, SQL and NoSQL Injection, Untrusted Deserialization, and XML External Entity Processing. To better understand the extent of capabilities available from Contrast, please go to www.contrastsecurity.com.

The WebGoat application provides helpful guidance to show a user how to attack the application. For the first test, Coalfire observed the capabilities of Contrast Assess to detect vulnerabilities when normal functional testing was performed without actively attacking the application. In Figure 4, on WebGoat, the user enters in an account name using the regular expression to get account information where the account last name is Smith.



Try It! String SQL Injection

The query in the code builds a dynamic query as seen in the previous example.
injection:

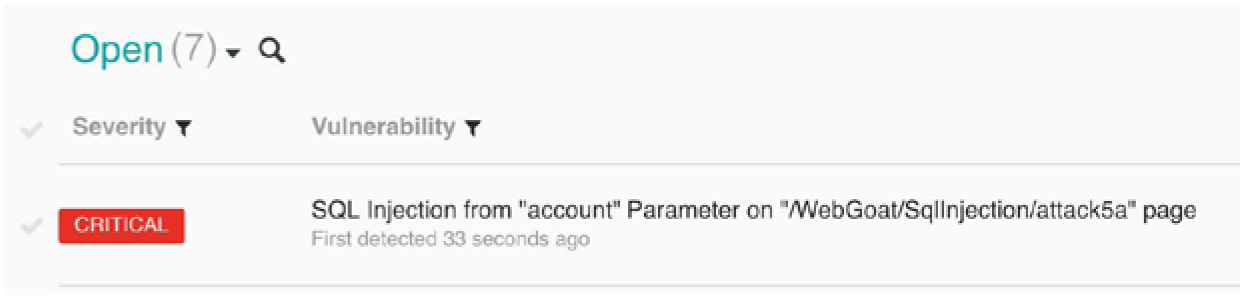
```
"select * from users where LAST_NAME = '" + userName + "'";
```

Using the form below try to retrieve all the users from the users table. You shoul
data for one user.

Account Name: Smith   Get Account Info

Figure 4 – Using the Application Normally

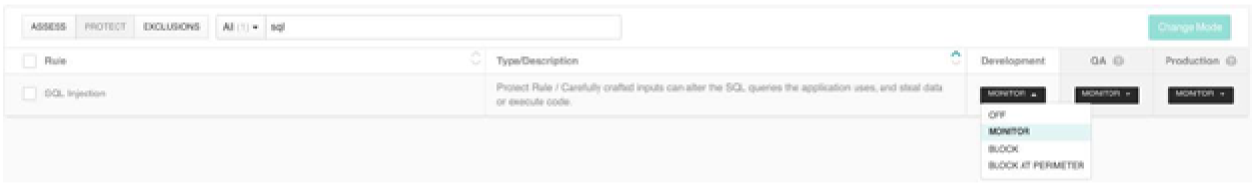
Contrast Assess reports the vulnerability that is present in the application, as shown in Figure 5. The detection of the vulnerability did not require an actual attempt to exploit the vulnerability; rather, the vulnerability was discovered as it would have been should functional testing of the application occurred. Contrast Assess assigns a severity to the vulnerability, which is useful for prioritizing and coordinating a response to correct the issue.



Severity	Vulnerability
CRITICAL	SQL Injection from "account" Parameter on "/WebGoat/SqlInjection/attack5a" page First detected 33 seconds ago

Figure 5 – Contrast Assess identifies a SQL Injection Vulnerability

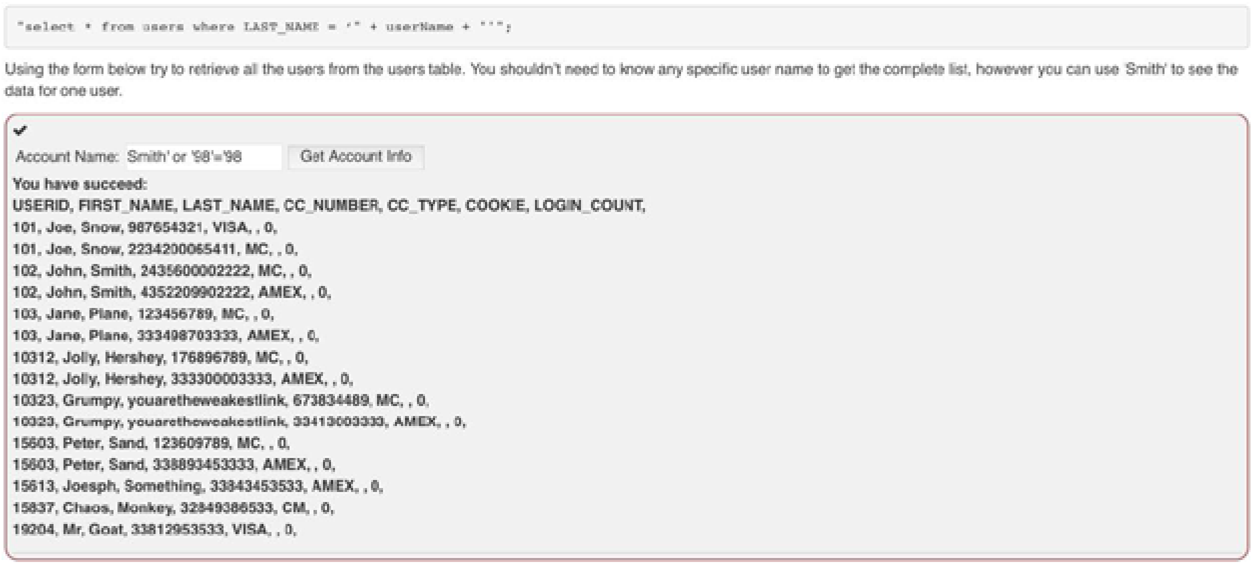
The security professional or user of Contrast TeamServer can determine how to handle different vulnerabilities across the various environments (development, QA, or production), as shown in Figure 6. A protect rule can be established for each environment with options including: Off, Monitor, Block, and Block at Perimeter. Depending on the severity of vulnerability, the user may choose to block the attack for production while continuing to monitor the issue in development or QA. This provides an option for an immediate response to detected vulnerabilities for the production environment, until the development team can correct the code, retest, and redeploy.



Rule	Type/Description	Development	QA	Production
SQL Injection	Protect Rule / Carefully crafted inputs can alter the SQL queries the application uses, and steal data or execute code.	MONITOR	MONITOR	MONITOR

Figure 6 – Options for Detected Vulnerabilities

With the protect rule set to "Monitor", the attempt to perform the SQL injection is successful, as shown in Figure 7. In this example, the request string included SQL Injection with Account Name = 'Smith' or '98' = '98'. With WebGoat, this subsequently resulted in a dump of all of the accounts in the database, including credit card primary account number.



```
"select * from users where LAST_NAME = '" + userName + "'";
```

Using the form below try to retrieve all the users from the users table. You shouldn't need to know any specific user name to get the complete list, however you can use 'Smith' to see the data for one user.

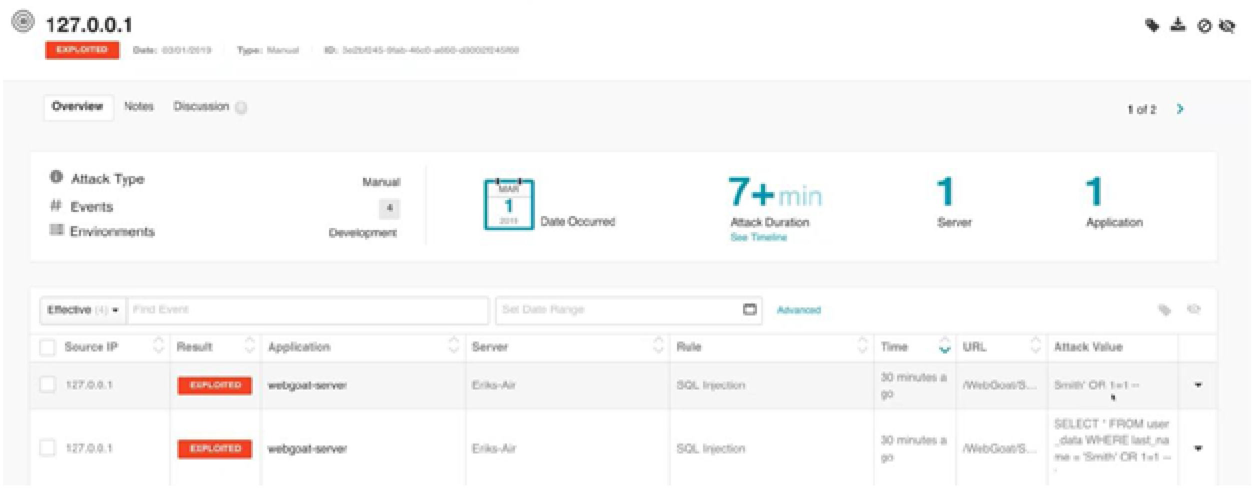
✓

Account Name: Smith' or '98'='98

```
You have succeed:
USERID, FIRST_NAME, LAST_NAME, CC_NUMBER, CC_TYPE, COOKIE, LOGIN_COUNT,
101, Joe, Snow, 987654321, VISA, , 0,
101, Joe, Snow, 2234200065411, MC, , 0,
102, John, Smith, 2435600002222, MC, , 0,
102, John, Smith, 4352209902222, AMEX, , 0,
103, Jane, Plane, 123456789, MC, , 0,
103, Jane, Plane, 333498703333, AMEX, , 0,
10312, Jolly, Hershey, 176896789, MC, , 0,
10312, Jolly, Hershey, 333300003333, AMEX, , 0,
10323, Grumpy, youaretheweakestlink, 673834489, MC, , 0,
10323, Grumpy, youaretheweakestlink, 33413003333, AMEX, , 0,
15603, Peter, Sand, 123609789, MC, , 0,
15603, Peter, Sand, 338893453333, AMEX, , 0,
15613, Joesph, Something, 33843453533, AMEX, , 0,
15837, Chaos, Monkey, 32849386533, CM, , 0,
19204, Mr, Goat, 33812953533, VISA, , 0,
```

Figure 7 - Successful SQL Injection Attack Results

Contrast Protect identifies that the application was successfully exploited, as shown in Figure 8. Contrast TeamServer shows when the attack occurred (date and time), the result of the attack, the type of attack that occurred, the frequency of the attack, the URL that was compromised, and the attack value. Additional useful information from TeamServer shows the source IP from where the attack was launched and the server where the application resided when the attack was successfully executed.



127.0.0.1

EXPLOITED | Date: 03/01/2019 | Type: Manual | ID: 3e2bf045-9fab-46c0-a660-d3002f245f66

Overview | Notes | Discussion — 1 of 2

Attack Type	Manual
Events	4
Environments	Development

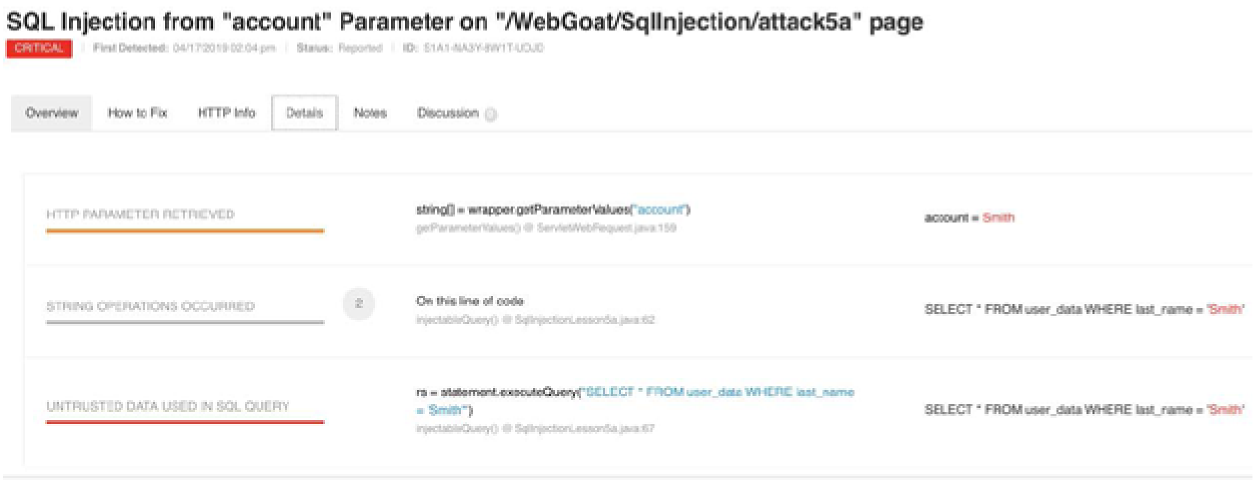
MAR 1 2019 Date Occurred — 7+min Attack Duration (See Timeline) — 1 Server — 1 Application

Effective (4) | Find Event | Set Date Range | Advanced

Source IP	Result	Application	Server	Rule	Time	URL	Attack Value
127.0.0.1	EXPLOITED	webgoat-server	Eriks-Air	SQL Injection	30 minutes ago	/WebGoat/S...	Smith' OR 1=1 --
127.0.0.1	EXPLOITED	webgoat-server	Eriks-Air	SQL Injection	30 minutes ago	/WebGoat/S...	SELECT * FROM user_data WHERE last_name = 'Smith' OR 1=1 --

Figure 8 – Contrast Review of Attacks

The Contrast Security platform provides useful insights into the vulnerability or the attack to help to determine exactly what happened with the application at the time of attack. Details on the vulnerability including the HTTP parameters used to discover the vulnerability, the string operation that was used, and the untrusted data used in the SQL query are displayed in Contrast TeamServer, as shown in Figure 9.



SQL Injection from "account" Parameter on "/WebGoat/SqlInjection/attack5a" page

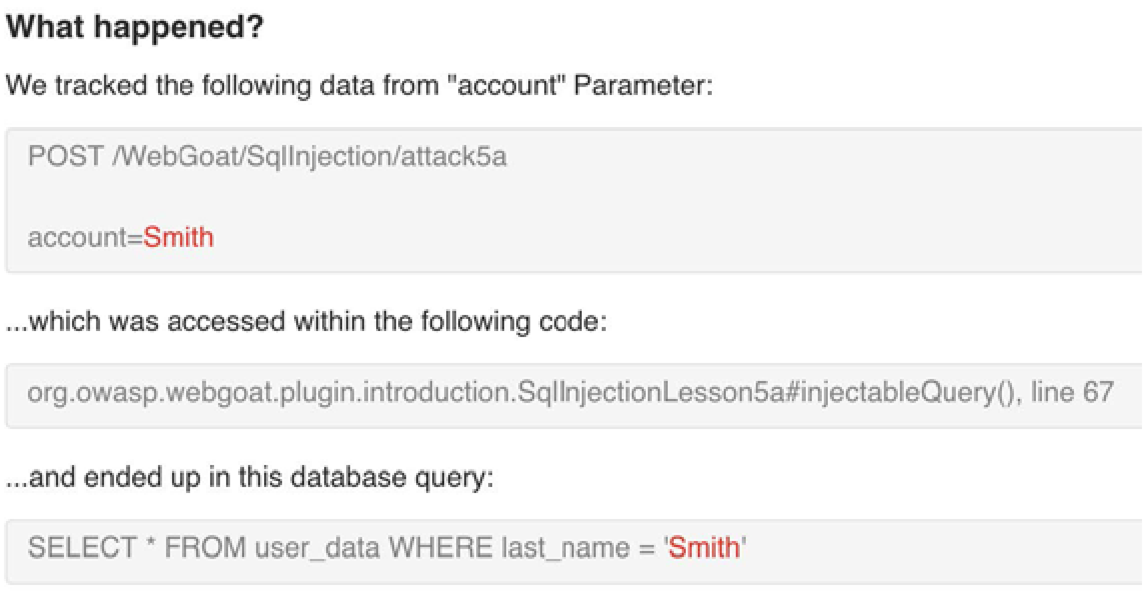
CRITICAL | First Detected: 04/17/2019 02:04 pm | Status: Reported | ID: S1A1-NA3Y-8W1T-UOJD

Overview | How to Fix | HTTP Info | Details | Notes | Discussion

HTTP PARAMETER RETRIEVED	string[] = wrapper.getParameterValues("account") getParameterValues() @ ServletWebRequest.java:159	account = Smith
STRING OPERATIONS OCCURRED (2)	On this line of code injectableQuery() @ SqlInjectionLesson5a.java:62	SELECT * FROM user_data WHERE last_name = 'Smith'
UNTRUSTED DATA USED IN SQL QUERY	rs = statement.executeQuery("SELECT * FROM user_data WHERE last_name = Smith'") injectableQuery() @ SqlInjectionLesson5a.java:67	SELECT * FROM user_data WHERE last_name = 'Smith'

Figure 9 – Attack Details

Details about where the vulnerability exists in the code are also provided, as shown in Figure 10. This illustrates what data could be passed with the query, where the code was accessed, and what query was used as a part of normal application testing when the vulnerability was discovered.



What happened?

We tracked the following data from "account" Parameter:

```
POST /WebGoat/SqlInjection/attack5a

account=Smith
```

...which was accessed within the following code:

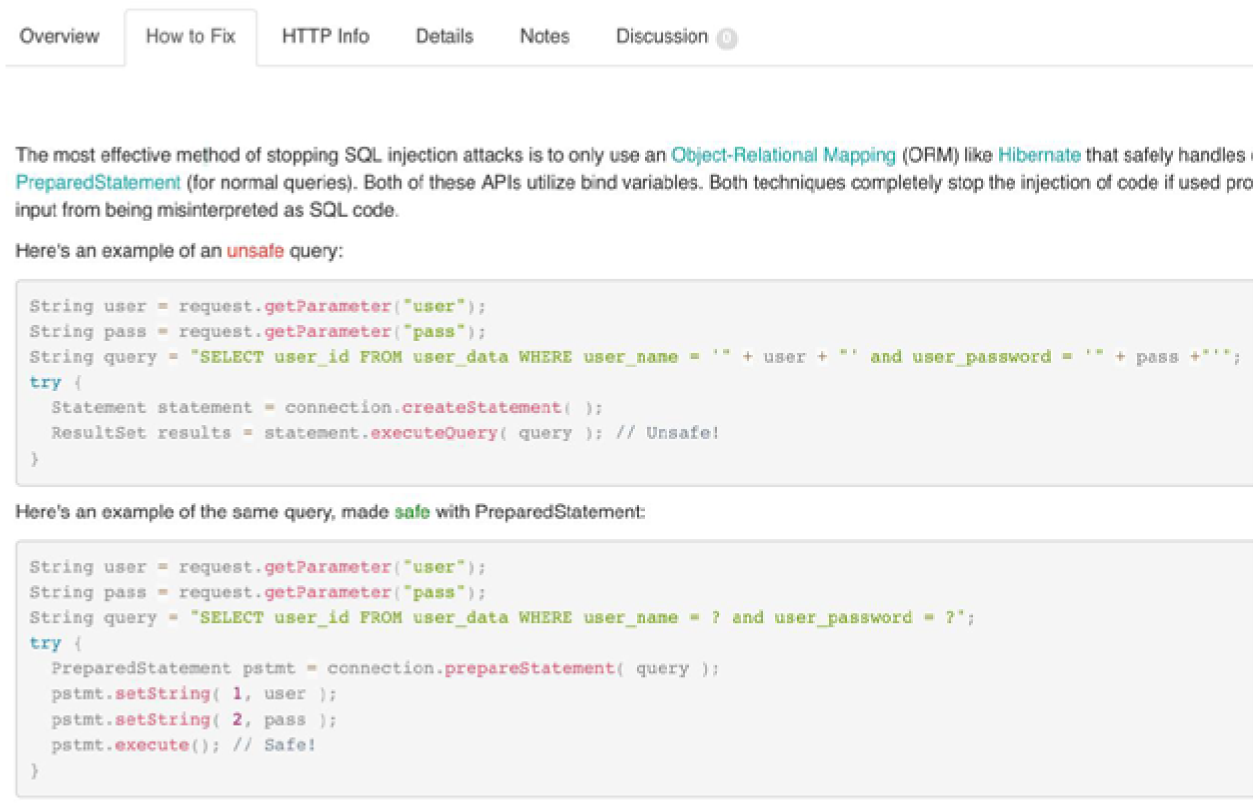
```
org.owasp.webgoat.plugin.introduction.SqlInjectionLesson5a#injectableQuery(), line 67
```

...and ended up in this database query:

```
SELECT * FROM user_data WHERE last_name = 'Smith'
```

Figure 10 – Additional Vulnerability Detail

Additionally, Contrast TeamServer supplies information about the vulnerability to educate the reviewer with methods for correcting the code to prevent the attack. Contrast TeamServer provides an example of unsafe code and how the code can be improved to be made "safe," as shown in Figure 11.



The most effective method of stopping SQL injection attacks is to only use an Object-Relational Mapping (ORM) like Hibernate that safely handles
PreparedStatement (for normal queries). Both of these APIs utilize bind variables. Both techniques completely stop the injection of code if used pro
input from being misinterpreted as SQL code.

Here's an example of an unsafe query:

```
String user = request.getParameter("user");
String pass = request.getParameter("pass");
String query = "SELECT user_id FROM user_data WHERE user_name = '" + user + "' and user_password = '" + pass +"'";
try {
  Statement statement = connection.createStatement( );
  ResultSet results = statement.executeQuery( query ); // Unsafe!
}
```

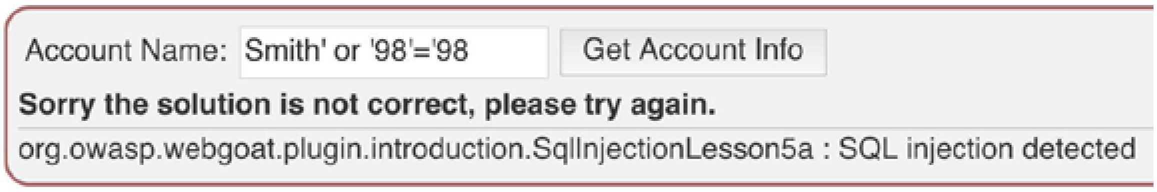
Here's an example of the same query, made safe with PreparedStatement:

```
String user = request.getParameter("user");
String pass = request.getParameter("pass");
String query = "SELECT user_id FROM user_data WHERE user_name = ? and user_password = ?";
try {
  PreparedStatement pstmt = connection.prepareStatement( query );
  pstmt.setString( 1, user );
  pstmt.setString( 2, pass );
  pstmt.execute(); // Safe!
}
```

Figure 11 – How to Improve the Code to Fix the Vulnerability

The Contrast Security platform provides options from Contrast TeamServer for handling a discovered vulnerability. These include adding exclusions to instruct Contrast Protect to leave the vector alone, create a virtual patch to mitigate the issue, block the IP of the attacker, configure a protection rule to address future attacks, or suppress the event from reporting. A virtual patch is a custom patch that can be created to mitigate vulnerabilities. Typically, virtual patches are used to address zero-day vulnerabilities for which a vendor or source patch is not yet available. Virtual patches are an excellent approach to handling zero-day vulnerabilities whereby the risk can be mitigated in the meantime between discovery and source patching and ideally before an exploit can be weaponized against the vulnerability.

For this effort, a protection rule was enabled from one of the available default rules, as shown in the drop-down menu in Figure 6 on page 9, to block the attack. The application did not have to be restarted for this rule to go into effect. Figure 12 shows the subsequent attack attempt from the perspective of WebGoat after the protect rule was enabled. The SQL injection attempt was blocked. WebGoat returns an error that the solution is not correct, and the attacker is unsuccessful. The nature of WebGoat as a teaching and testing platform reveals more details about the error such as the type of attack used.



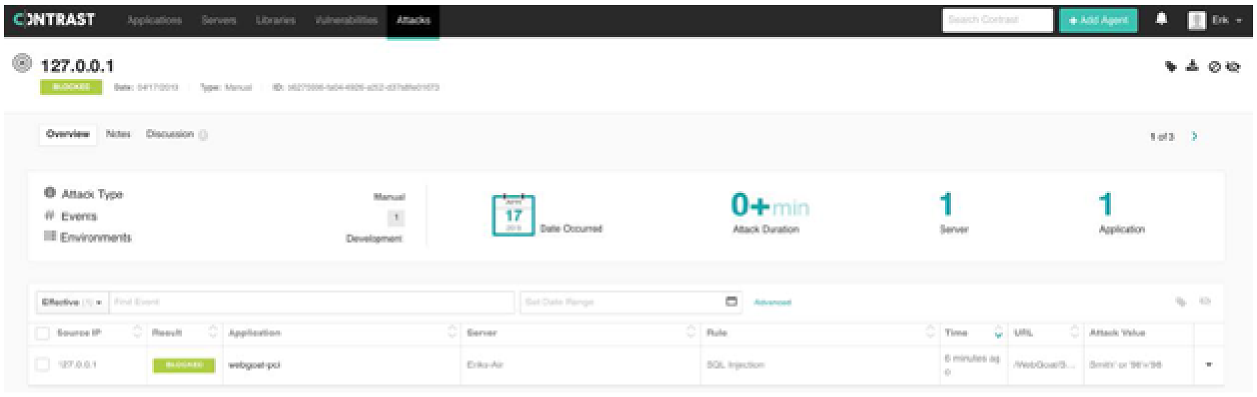
Smith' or '98'='98

Sorry the solution is not correct, please try again.

org.owasp.webgoat.plugin.introduction.SqlInjectionLesson5a : SQL injection detected

Figure 12 – Results of SQL Injection Attempt with Protect Rule in Place

Contrast TeamServer reports that the attack was blocked, as shown in Figure 13.



127.0.0.1

BLOCKED

Attack Type	Manual
Events	1
Environments	Development

17 Date Occurred

0+min Attack Duration

1 Server

1 Application

Effective (1)

Source IP	Result	Application	Server	Rule	Time	URL	Attack Value
127.0.0.1	BLOCKED	webgoat-pci	Eriks-Air	SQL Injection	6 minutes ago	/WebGoat/S...	Smith' or '98'='98

Figure 13 – Contrast TeamServer Showing SQL Injection Attempt Blocked

The nature of Contrast Protect to be integrated with the application supports context awareness for attack methods to more accurately identify the effectiveness of attackers to exploit the application. The data points collected from Contrast Protect can be integrated with external SIEM solutions, such as Splunk, to support application security dashboards to help security operations understand the threats to the organizations' applications. Figure 14 illustrates a Splunk dashboard generated from information provided by Contrast Protect. A key value of Contrast Protect data for the Security Operations Center (SOC) is the ability to differentiate which attacks are ineffective and decrease the noise of false positives.

The graphs in the dashboard further reveal a breakdown of attack attempts against applications both by application and by outcome. Where attackers used methods that were not supported by vulnerabilities within the application, the attacks were ruled ineffective. Where Contrast Protect was able to mitigate attacks due to weaknesses in application code, the attack was recorded as blocked. The graph illustrates successful exploits, as well as where an attacker simply probed the application for vulnerabilities. The insights provided by Contrast Assess and Contrast Protect allow security teams to focus more strategically on areas of legitimate concern.

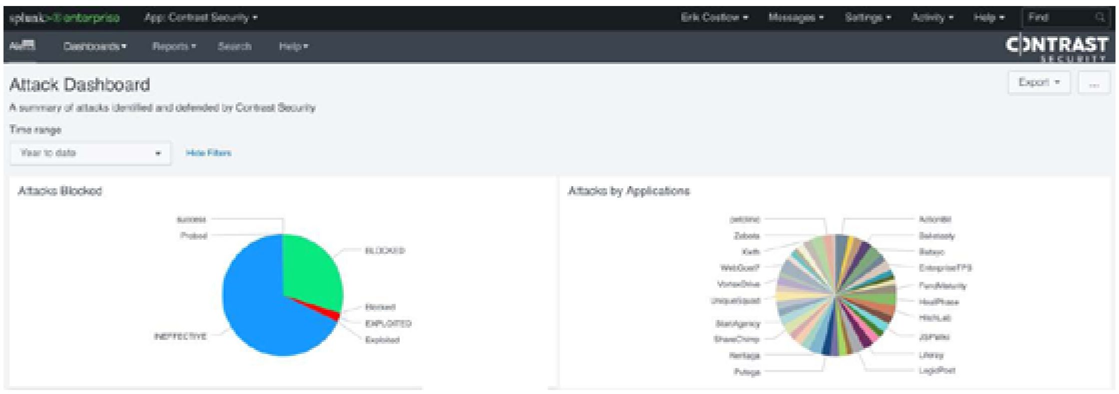


Figure 14 – Example Attack Dashboard

Scope and Approach for Review

The understanding of Contrast Assess and Contrast Protect and their combined capabilities to support application security was gained through review of product specifications, installation guides, configuration guides, administration guides, and integration documentation provided by Contrast and made generally available from Contrast's public-facing web site. Coalfire further conducted interviews and engaged in live product demonstrations with Contrast personnel. Live product demonstrations made use of the Contrast Team Server provided as a SaaS solution. The Contrast agent security sensors were integrated into the OWASP WebGoat Project to demonstrate capabilities against known vulnerabilities. Coalfire also tested the application following guidance from Contrast.

Coalfire's review of Contrast Assess and Contrast Protect began with a general alignment of the applicability of the solution against high-level PCI DSS, PA-DSS, and PCI Software Security Framework requirements and objectives. This was further narrowed down to specific requirements that were considered applicable to Contrast Assess and/or Contrast Protect. An assessment of capability for the reviewed technology to address the applicable requirement was then conducted and documented. This analysis primarily focused on what a PCI DSS QSA might review when following the PCI DSS testing procedures during an assessment of applicable requirements.

SCOPE OF TECHNOLOGY AND STANDARDS TO REVIEW

Coalfire was tasked by Contrast to review their integrated application security platform made up of Contrast Assess and Contrast Protect. The primary focus of the review included components, features, and functionality of Contrast Assess and Contrast Protect. The review did not include an assessment of the security of a SaaS or on-premises implementation of the solution regarding its proximity with a cardholder data environment (CDE), either as a CDE system or a connected to and security impacting system, and applicable control requirements that would need to be considered for the security of the CDE.

For this review, Coalfire included requirements from PCI DSS Requirements and Security Assessment Procedures Version 3.2.1 May 2018 publication, PCI Payment Application Data Security Standard Version 3.2 May 2016 publication, and the PCI Software Security Framework Secure Software Requirements and Assessment Procedures Version 1.0 January 2019 publication. These publications are generally available for download from https://www.

pcisecuritystandards.org. For broader understanding of the requirements and their applicability to technical solution implementation, Coalfire also reviewed supporting documentation provided by the PCI Security Standards Council (SSC) including the approach to assessment guidance, a sample report on compliance form, a sample report on validation form, and self-assessment questionnaires. Applied understanding of PCI DSS requirements, PA-DSS requirements, and PCI Software Security Framework requirements and recommendations was also made available from PCI DSS guidance documentation on relevant topics such as scoping and segmentation, best practices for maintaining PCI DSS compliance, risk assessment guidance, and blog posts regarding the frameworks from members of the PCI SSC.

COALFIRE EVALUATION METHODOLOGY

Coalfire initially examined the PCI requirements from each of the applicable frameworks and identified each requirement as procedural (organizational) or technical (implementation). Qualification of a requirement as procedural or technical was based on a review of the requirement narrative, testing procedures, and guidance.

"Non-technical" procedural requirements that included definition and documentation of policies, procedures, and standards were not considered directly applicable to the technical solution. Likewise, "non-technical" requirements including operational procedures that describe manual processes were not assessed against the technology's capabilities, though consideration was made for whether the technology could help to automate some of the manual processes should the technology be adopted by the payment entity as part of their documented and formalized processes. Examples of this type of "non-technical" requirement include maintenance of visitor logs, verification of individual's identity as part of an access granting processes, performance of asset inventories, or generation of network topology diagrams or flow diagrams to define standards.

Technical requirements were assessed to determine the applicability of the solution and/or solution components to address the outcomes that were defined by the requirement. In other words, could the solution produce the evidence necessary to satisfy an assessment.

Contrast Security Platform Applicability to PCI DSS

The following tables outline the applicability of Contrast Assess and Contrast Protect to PCI security frameworks. The following tables highlight the requirements from each framework where the Contrast integrated application security platform is applicable to support outcomes in alignment with each listed requirement. The capabilities of Contrast Assess and Contrast Protect minimally provide support for organizational operational procedures around the secure development and deployment of software. Additional advantages may be gained by the user of the Contrast Security platform for achieving the desired security outcomes. These include the capability to continuously assess and protect the payment application from current and future vulnerabilities, the support for operational awareness for the degree of risk associated with vulnerabilities regarding the capability for an attacker to exploit given the usability of the software or software component, and the capability to address vulnerabilities earlier in the life cycle, rather than prior to deployment to production or the general availability release of software.

PCI DSS V3.2.1 COMPLIANCE APPLICABILITY DETAIL

The following table details compliance capabilities of Contrast Assess and Contrast Protect as applicable to PCI DSS v3.2.1.

ID	REQUIREMENT	CONTRAST SECURITY PLATFORM APPLICABILITY
2.4	Maintain an inventory of system components that are in scope for PCI DSS.	Contrast Assess can discover and maintain an inventory of system components that make up the applications it assesses. This includes microservices, APIs, open source software, custom code, libraries, component technologies, architecture, and back end connections. The architecture components and back end connections include all servers that host components of the applications that are in use, including databases. Contrast Assess generates simple diagrams that illustrate the application's major architectural components. This information may be useful to the payment entity's efforts for maintaining a comprehensive system inventory. The completeness of the payment entity's inventory of systems will likely be generated through multiple sources, of which Contrast Assess may be one source for inventory data.
3.4	Render PAN unreadable anywhere it is stored (including on portable digital media, backup media, and in logs) by using any of the following approaches: • One-way hashes based on strong cryptography (hash must be of the entire PAN) • Truncation (hashing cannot be used to replace the truncated segment of PAN) • Index tokens and pads (pads must be securely stored) • Strong cryptography with associated key-management processes and procedures Note: It is a relatively trivial effort for a malicious individual to reconstruct original PAN data if they have access to both the truncated and hashed version of a PAN. Where hashed and truncated versions of the same PAN are present in an entity's environment, additional controls must be in place to ensure that the hashed and truncated versions cannot be correlated	Contrast Assess can identify where weak cryptographic algorithms are used by the application as part of the method for encrypting data at rest. The solution does not enforce encryption and truncation, or ensure the secure storage of data. Nor does the solution provide key management. The solution can identify and be used to inform where the payment entity needs to remediate the use of weak cryptographic algorithms.

ID	REQUIREMENT	CONTRAST SECURITY PLATFORM APPLICABILITY
6.1	Establish a process to identify security vulnerabilities, using reputable outside sources for security vulnerability information, and assign a risk ranking (for example, as "high," "medium," or "low") to newly discovered security vulnerabilities. Note: Risk rankings should be based on industry best practices as well as consideration of potential impact. For example, criteria for ranking vulnerabilities may include consideration of the CVSS base score, and/or the classification by the vendor, and/or type of systems affected. Methods for evaluating vulnerabilities and assigning risk ratings will vary based on an organization's environment and risk-assessment strategy. Risk rankings should, at a minimum, identify all vulnerabilities considered to be a "high risk" to the environment. In addition to the risk ranking, vulnerabilities may be considered "critical" if they pose an imminent threat to the environment, impact critical systems, and/or would result in a potential compromise if not addressed. Examples of critical systems may include security systems, public-facing devices and systems, databases, and other systems that store, process, or transmit cardholder data.	Contrast solutions can be incorporated as part of the procedural methods that the entity uses to identify new vulnerabilities in the software that it is assessing and protecting. Risk rankings are provided by Contrast Assess and Contrast Protect on the basis of severity for the vulnerability relative to source information about the vulnerability as well as exposure of the vulnerability given the methods that are employed by the application. Contrast incorporates outside reputable sources for security vulnerability information to expand the knowledge base of vulnerabilities and coding weaknesses that can result in successful exploitation by an attacker. Additionally, Contrast actively utilizes bug bounty programs to leverage expertise from the wider security community.

ID	REQUIREMENT	CONTRAST SECURITY PLATFORM APPLICABILITY
6.2	Ensure that all system components and software are protected from known vulnerabilities by installing applicable vendor-supplied security patches. Install critical security patches within one month of release. Note: Critical security patches should be identified according to the risk ranking process defined in Requirement 6.1.	Contrast Protect may be used by payment entities in the case of zero-day vulnerabilities to create virtual patches that detect and protect against exploits that attempt to compromise the vulnerability. This is a stop gap measure to mitigate the threat of a zero-day exploit during the window of vulnerability between the time of the vulnerability discovery and when the software vendor can release a patch to remediate the vulnerability. This does not ultimately remove responsibility for the vendor to address vulnerabilities through the development and release of security patches, or the payment entity's responsibility to apply patches made available by the vendor in a timely manner.
6.3.2	Review custom code prior to release to production or customers in order to identify any potential coding vulnerability (using either manual or automated processes) to include at least the following: Code changes are reviewed by individuals other than the originating code author, and by individuals knowledgeable about code-review techniques and secure coding practices. Code reviews ensure code is developed according to secure coding guidelines Appropriate corrections are implemented prior to release. Code-review results are reviewed and approved by management prior to release. Note: This requirement for code reviews applies to all custom code (both internal and public-facing), as part of the system development life cycle. Code reviews can be conducted by knowledgeable internal personnel or third parties. Public-facing web applications are also subject to additional controls, to address ongoing threats and vulnerabilities after implementation, as defined at PCI DSS Requirement 6.6.	Contrast Assess provides an automated process for assessing custom application code for vulnerabilities. Applying Contrast's extensive set of rules to the application, Contrast Assess can continually identify and assess vulnerabilities in the application throughout the application lifecycle. Assessment occurs wherever Contrast Assess is integrated with the application. Given this reality, vulnerabilities are discoverable earlier and more accurately in the lifecycle during any automated or manual functional testing. Contrast Assess can also evaluate Open Source libraries that are part of the application, using both security rules and a database of CVEs. Organizations can address the discovered vulnerabilities through developer coding efforts. The Contrast Security Platform can be integrated with third-party issues tracking systems like Jira, Bugzilla, and others to automatically create tickets for discovered vulnerabilities. Remediation and compliance policies can be set with The Contrast Security Platform and automatically apply to applications. Behaviors can be set to determine what happens with the application when a vulnerability is discovered, for example, management review and approval of code changes would be required before the remediation of the vulnerability can be closed. This aids the feedback loop between security and development. Security professionals can review Contrast Assess results from assessing the software throughout the lifecycle and can verify the security state of the software prior to packaging and deployment to production. QSAs commonly will collect evidence for 6.3.2.b, where the vulnerability, remediation request, management review and approval, and completion of remediation are documented in Jira or other similar ticketing system.

ID	REQUIREMENT	CONTRAST SECURITY PLATFORM APPLICABILITY
6.5	Address common coding vulnerabilities in software-development processes as follows: • Train developers at least annually in up-to-date secure coding techniques, including how to avoid common coding vulnerabilities. • Develop applications based on secure coding guidelines. Note: The vulnerabilities listed at 6.5.1 through 6.5.10 were current with industry best practices when this version of PCI DSS was published. However, as industry best practices for vulnerability management are updated (for example, the OWASP Guide, SANS CWE Top 25, CERT Secure Coding, etc.), the current best practices must be used for these requirements.	When incorporated into the entity's processes, Contrast Assess can successfully identify vulnerabilities throughout the SDLC. Contrast Assess capabilities to detect vulnerabilities include the PCI DSS listed vulnerabilities as well as many others that are not listed. Contrast Assess assesses custom code, open source libraries, and methods of using the code to identify vulnerabilities within the context of the running application.
6.5.1	Injection flaws, particularly SQL injection. Also consider OS Command Injection, LDAP and XPath injection flaws as well as other injection flaws.	6.5.1 through 6.5.10 address specific vulnerabilities and flaws that were prevalent at the time of publication of PCI DSS v3.2.1; however, these are not an exhaustive list of all prevalent vulnerabilities that may be present in code. Contrast Assess assesses custom code, open source libraries, and methods of using the code to identify vulnerabilities within the context of the running application. Contrast Assess can accurately detect substantially more vulnerabilities than those outlined in 6.5.1 through 6.5.10. Contrast Assess can detect injection flaws that would allow an attacker to launch a SQL injection attack. Additionally, Contrast Assess can identify the context of the vulnerability to determine the degree to which the vulnerability can be successfully exploited, e.g., Contrast Assess understands the different types of databases, including NoSQL database like Couchbase, MongoDB, and so forth, and can detect their specific injection issues. Contrast Assess can identify these vulnerabilities with greater accuracy and thoroughness than Static Application Security Testing (SAST) tools.
6.5.3	Insecure cryptographic storage	Contrast Assess integration with the application allows for assessment and identification for the use of insecure cryptography used to securely store data.
6.5.4	Insecure communications	Contrast Assess integration with the application allows for assessment and identification for the use of missing or insecure cryptography used for securing communications.

ID	REQUIREMENT	CONTRAST SECURITY PLATFORM APPLICABILITY
6.5.5	Improper error handling	Contrast Assess can identify coding errors or weaknesses that would allow an attacker to gather information useful for attack garnered from poor error handling.
6.5.6	All “high risk” vulnerabilities identified in the vulnerability identification process (as defined in PCI DSS Requirement 6.1).	Contrast Assess can to identify and categorize vulnerabilities by risk according to associated Common Weakness Enumeration (CWE) data as well as within the realm of context relative to how the vulnerable code is used by applications.
6.5.7	Cross-site scripting (XSS) - Apply to Web Applications and Application Interfaces (internal or external)	Contrast Assess can identify XSS errors.
6.5.8	Improper access control (such as insecure direct object references, failure to restrict URL access, directory traversal, and failure to restrict user access to functions). Apply to Web Applications and Application Interfaces (internal or external)	Contrast Assess can identify improper access control vulnerabilities in applications.
6.5.9	Cross-site request forgery (CSRF) - Apply to Web Applications and Application Interfaces (internal or external)	Contrast Assess can identify CSRF vulnerabilities in applications.
6.5.10	Broken authentication and session management. Apply to Web Applications and Application Interfaces (internal or external)	Contrast Assess can identify broken authentication and session management vulnerabilities in applications.

ID	REQUIREMENT	CONTRAST SECURITY PLATFORM APPLICABILITY
6.6	For public-facing web applications, address new threats and vulnerabilities on an ongoing basis and ensure these applications are protected against known attacks by either of the following methods: Reviewing public-facing web applications via manual or automated application vulnerability security assessment tools or methods, at least annually and after any changes Note: This assessment is not the same as the vulnerability scans performed for Requirement 11.2. Installing an automated technical solution that detects and prevents web-based attacks (for example, a web-application firewall) in front of public-facing web applications, to continually check all traffic.	Contrast Assess and Contrast Protect can provide continuous assessment and protection for applications. The integration of the Contrast Security platform sensors within the application allows the application to be continuously self-assessing and self-protecting. It is recommended that only Contrast Protect be enabled in the production runtime environment. Contrast Assess meets the requirement for automated application security assessment by continuously analyzing the application for vulnerabilities. Every time a change is made to the application, Contrast Assess analyzes the application or API in both development and QA environments, and provides instant results to the development team. Further, Contrast Protect meets the automated technical protection requirement by continuously detecting and preventing both known attacks and zero-day attacks. Contrast Protect is a RASP solution. Contrast Protect can prevent the exploit of many classes of vulnerability, automatically protecting against most zero-day attacks. For other vulnerability classes, administrators can quickly create and deploy virtual patches within seconds, and Contrast Protect can standardize protection for zero days within hours. In addition, Contrast Lab offers instant guidance and quick product updates that can be effortlessly scaled to the entire application portfolio. Due to the accuracy and thoroughness of web application protection capabilities provided by Contrast Protect, this solution can be used in lieu of a WAF.
10	Track and monitor all access to network resources and cardholder data Logging mechanisms and the ability to track user activities are critical in preventing, detecting, or minimizing the impact of a data compromise. The presence of logs in all environments allows thorough tracking, alerting, and analysis when something does go wrong. Determining the cause of a compromise is very difficult, if not impossible, without system activity logs.	Contrast Assess and Contrast Protect can enhance application logging by adding log statements that can be used for forensic analysis. Contrast Assess and Contrast Protect augmented logging capabilities can be used to track individual access to sensitive data and resources.

PA-DSS V3.2 COMPLIANCE APPLICABILITY DETAIL

The following table details compliance capabilities of Contrast Assess and Contrast Protect as applicable to PA-DSS v3.2. These findings are specific to how payment application vendors can utilize Contrast Assess and/or Contrast Protect for addressing requirements relative to the development of payment applications for sale to payment entities.

ID	REQUIREMENT	CONTRAST SECURITY PLATFORM APPLICABILITY
4	Log payment application activity	Contrast Assess and Contrast Protect can enhance application logging by adding log statements that can be used for forensic analysis. Contrast Assess and Contrast Protect augmented logging capabilities can be used to track individual access to sensitive data and resources.
5.2	Develop all payment applications to prevent common coding vulnerabilities in software development processes. Note: The vulnerabilities listed in PA-DSS requirement 5.2.1 through 5.2.10 and in PCI DSS at 6.5.1 through 6.5.10 were current with industry best practices when this version of PA-DSS was published. However, as industry best practices for vulnerability management are updated (for example, the OWASP Top 10, SANS CWE Top 25, CERT Secure Coding, etc.), the current best practices must be used for these requirements. **Aligns with PCI DSS Requirement 6.5**	Contrast Assess is a unified Application Security Testing (*AST) solution that can continuously assess the software through the SDLC to help identify coding vulnerabilities in custom code and in open source libraries. Contrast Assess is able to contextually identify code flaws within the application and score and prioritize flaws based on the ability to actively exploit the vulnerability within the application. Because Contrast Assess is part of the application, it can deliver security results as fast as the code changes. This allows developers to find and fix vulnerabilities without requiring dedicated security experts and point in time security scanning to be performed. Contrast Assess makes recommendations for coding changes to address identified vulnerabilities that the developer can apply for remediation. To provide instant feedback to development teams, Contrast Assess integrates seamlessly into the SDLC and into the toolsets that development and operations teams are already using, including native integration into IDEs, ChatOps, ticketing systems, and CI/CD tools, and includes a RESTful API. Contrast Assess also automatically discovers third-party libraries and triggers alerts to known risks associated with libraries. Contrast Assess analyzes the libraries to discover new risks and provides critical versioning and usage information to help development teams remediate risks. Contrast Assess analyzes code and provides a complete inventory of all third-party libraries that are used in the application. Contrast Assess flags libraries used as part of the application code for policy violations. Contrast Assess reports on versioning and out of date libraries and identifies all known CVEs present in libraries. Contrast's founders and employees are the founders of OWASP and are the driving force behind many of the most successful OWASP projects.
5.2.1	Injection flaws, particularly SQL injection.	Contrast Assess can help developers identify injection flaws within the application and can make recommendations for code fixes to remediate the flaw.

ID	REQUIREMENT	CONTRAST SECURITY PLATFORM APPLICABILITY
5.2.3	IInsecure cryptographic storage	Contrast Assess can identify insecure cryptography used by the application and make recommendations for improving the strength of cryptography.
5.2.4	Insecure communications	Contrast Assess can detect insecure cryptography used by the application and make recommendations for improving the strength of cryptography to improve communication security.
5.2.5	Improper error handling	Contrast Assess can detect error handling issues where results of errors may reveal information that could be useful for an attacker to gain unauthorized access to the application.
5.2.6	IAll "high risk" vulnerabilities as identified in the vulnerability identification process at PA-DSS Requirement 7.1	Contrast Assess can detect and report on "high risk" vulnerabilities as identified in the vulnerability identification process at PA-DSS Requirement 7.1 Contrast Assess uses information from reputable sources and researchers to expand the knowledge base of vulnerabilities and coding weaknesses that can result in successful exploitation by an attacker.
5.2.7	Cross-site scripting (XSS)	Contrast Assess can detect XSS weaknesses in applications. Contrast Assess can identify data that flows, for example, from an encoded cookie, through a data bean, into a session store, into the JSF component, and finally into a browser, indicating an XSS weakness.
5.2.8	Improper access control such as insecure direct object references, failure to restrict URL access, and directory traversal.	Contrast Assess can detect coding errors that violate access control policies such as insecure direct object reference, failure to restrict URL access, and directory traversal.
5.2.9	Cross-site request forgery (CSRF)	Contrast Assess can detect CSRF vulnerabilities in applications.
5.210	Broken authentication and session management	Contrast Assess can detect broken authentication and session management vulnerabilities in code.
7.1	Software vendors must establish a process to identify and manage vulnerabilities, as follows: Note: Any underlying software or systems that are provided with or required by the payment application (for example, web servers, third-party libraries and programs) must be included in this process. Aligns with PCI DSS Requirement 6.1	Contrast Assess provides tools and workflow to identify and manage vulnerabilities across the SDLC. Contrast Assess evaluates both custom code and third-party libraries and programs that may be used as part of the payment applications architecture.

ID	REQUIREMENT	CONTRAST SECURITY PLATFORM APPLICABILITY
7.1.1	Identify new security vulnerabilities using reputable sources for obtaining security vulnerability information.	CThe Contrast Security platform stays up to date on the latest vulnerability and attack trends including Method Tampering attacks, non-Struts2 OGNL Injection attacks, Padding Oracle attacks, and so forth. Additionally, Contrast actively utilizes bug bounty programs to leverage expertise from the wider security community to improve the platforms awareness of new vulnerabilities and threats.
7.1.2	Assign a risk ranking to all identified vulnerabilities, including vulnerabilities involving any underlying software or systems provided with or required by the payment application. Note: Risk rankings should be based on industry best practices as well as consideration of potential impact. For example, criteria for ranking vulnerabilities may include consideration of the CVSS base score, and/or the classification by the vendor, and/or impact to application functionality. Risk rankings should, at a minimum, identify all vulnerabilities considered to be a "high risk" to the application. In addition to the risk ranking, vulnerabilities may be considered "critical" if they pose an imminent threat, impact critical application components, or would result in a potential compromise if not addressed.	Risk rankings are applied by the Contrast Security platform for all identified vulnerabilities involving the underlying software or systems. The risk rankings that are used are based on industry best practices as well as consideration of potential impact. Because Contrast Assess is part of the application, it can determine the degree of risk for the vulnerability within the context for how the code is used by the application. Risk rankings can be further customized for vulnerabilities to align with the customer's specific requirements and processes.
7.1.3	Test payment applications and updates for the presence of vulnerabilities prior to release	Contrast Assess continuously assesses the application for vulnerabilities. As it can be used in any phase of development or testing, Contrast Assess can identify and report on vulnerabilities. This allows developers, operations, and security personnel to have a level of confidence in the code being promoted to a release version.

Table 2 - Contrast Security Platform Capabilities Alignment of Applicability to PA-DSS v3.2 Requirements Findings

PCI SECURE SOFTWARE STANDARD COMPLIANCE APPLICABILITY DETAIL

The following table details compliance capabilities of Contrast Assess and Contrast protect as applicable to the PCI Secure Software Standard.

ID	REQUIREMENT	CONTRAST SECURITY PLATFORM APPLICABILITY
1.3	Critical assets represent the sensitive data, functions, and resources that have business value and require confidentiality, integrity, or resiliency protection. There are numerous analysis techniques that can be used to identify critical assets, including Mission Impact Analysis (MIA), Functional Dependency Network Analysis (FDNA), and Mission Threat Analysis. Additional information and techniques can be found in publications such as the appendixes of NIST Special Publication 800-160 or in other publications from industry standards bodies such as EMVCo, ISO or ANSI.	Contrast Assess can discover and maintain an inventory of system components that make up the applications it assesses. This includes microservices, APIs, open source code, custom code, libraries, component technologies, architecture, and back end connections. The architecture components and back end connections include all servers that host components of the applications that are in use, including databases. Contrast Assess generates simple diagrams that illustrate the application's major architectural components. This information may be useful to the payment entity's efforts for maintaining a comprehensive system inventory.
2.1	All functions exposed by the software are enabled by default only when and where it is a documented and justified part of the software architecture.	Contrast Assess essentially enumerates the application surface and provides route coverage. On supported frameworks, Contrast Assess can show all access and entry points exposed by the application. Information provided by Contrast Assess can be used to identify deviations when findings are compared with the vendor's documented software architecture and listing of justified entry and exit points. Contrast Assess and Contrast Protect can be used to identify whether the functions expose protocols, methods, or services that have publicly disclosed vulnerabilities as compared against public vulnerabilities databases. Contrast Assess can identify where the vulnerabilities exist, whether the vulnerability can be exploited, and offer remediation options for exposed vulnerabilities. Contrast Protect can protect the applications from attacks against exposed vulnerabilities.

ID	REQUIREMENT	CONTRAST SECURITY PLATFORM APPLICABILITY
3.3	The software protects the confidentiality and integrity of sensitive data (both transient and persistent) during retention. Noted: Security Objective: Software Protection Mechanisms included several specific software control objectives that are required to be implemented to protect sensitive data during storage, processing, and transmission. Those control objectives should be analyzed to determine their applicability to the types of sensitive data retained by the software.	Contrast Assess can identify where weak cryptographic algorithms are used by the application for the encryption of data at rest.
4.1	Attack scenarios applicable to the software are identified. Noted: This requirement is an extension of Control Objective 10.	Contrast Assess can identify the entry and egress points of the application including how data flows between the application endpoints, for instance, between the web front end of an application and a backend database. This information can be useful for the software vendor to identify relevant attack scenarios for the software. Moreover, this information can be used to determine the locations for appropriate application of authentication methods or trust model to secure the entry and egress points.
4.2	Software security controls are implemented to mitigate software attack.	Contrast Protect can be used to address or mitigate threats that may be identified by the vendor through their threat analysis. This assumes that the vendor will provide guidance for the use of Contrast Protect to their customer for deployment with the application with instructions for configuration of Contrast Protect for the implementation of threat mitigation.
7.1	Approved cryptographic algorithms and methods are used for securing critical assets. Approved cryptographic algorithms and methods are those recognized by industry accepted standards bodies – for example: NIST, ANSI, ISO, and EMVCo. Cryptographic algorithms and parameters that are known to be vulnerable are not used.	Where an application is using cryptography, Contrast Assess can identify and report on the use of weak cryptographic algorithms pertaining to the encryption of data at rest.

ID	REQUIREMENT	CONTRAST SECURITY PLATFORM APPLICABILITY
8.1	All access attempts and usage of critical assets is tracked and traceable to a unique individual. Noted: This Secure Software Standard recognizes that some execution environments cannot support the detailed logging requirements in other PCI standards. Therefore, the term “activity tracking” is used here to differentiate the expectations of this standard with regard to logging from similar requirements in other PCI standards.	As stated in the requirement description, several applications do not produce enough information in native logging methods to enable proper investigation capabilities through log analysis. Contrast Assess and Contrast Protect can enhance application logging by adding log statements that can be used for forensic analysis. Contrast Assess and Contrast Protect augmented logging capabilities can be used to track individual access to sensitive data and resources.
8.2	All activity is captured in sufficient and necessary detail to accurately describe what specific activities were performed, who performed them, the time they were performed, and which crucial assets were impacted.	The augmented logging capability of Contrast Assess and Contrast Protect can capture the enablement of privileged modes of operation by the application, the disabling of encryption of sensitive data, the decryption of sensitive data, the exporting of sensitive data to other systems or processes, the failed authentication attempts, the disabling or deleting of security controls, or the altering of security functionality within the application. The tracking methods record the unique identification of the person, system, or entity performing the access, a timestamp for each tracked event, and details on what critical assets were accessed. Contrast Assess and Contrast Protect can be configured to ensure that sensitive data is not directly recorded in the tracking data.
9.1	The software detects and alerts upon detection of anomalous behavior, such as changes in post-deployment configuration or obvious attack behavior.	By watching the application execute, Contrast Assess and Contrast Protect provides instant and accurate feedback about vulnerabilities as they occur within the code of the application. Contrast Assess and Contrast Protect provide the functionality of differentiating between normal activity and anomalous user behavior. This allows developers to focus on business logic when developing the application.

ID	REQUIREMENT	CONTRAST SECURITY PLATFORM APPLICABILITY
10.1	Software threats and vulnerabilities are identified, assessed, and addressed	Contrast Assess can assist the software vendor through its discovery of application vulnerabilities and assessment of the vulnerability's exposure to exploitation. Contrast Assess can identify and assess vulnerabilities more thoroughly and accurately because it is part of the applications methods. This identification and assessment of vulnerabilities is performed without the need of expert security knowledge; rather, Contrast Assess turns all usage into a security test to better minimize the opportunity of vulnerabilities being introduced or missed. This awareness of vulnerabilities that exist within the application and the degree to which the vulnerability can be exploited allows for the application vendor to address the vulnerability before the software is promoted to production.
10.2	Vulnerabilities in the software and third-party components are tested for and fixed prior to release.	When Contrast Assess is used during development and testing, it can help to identify security issues before the code is promoted. Unlike other systems, Contrast Assess turns all usage into a security test to better minimize the opportunity of vulnerabilities being introduced or missed. Contrast Assess also monitors for the introduction of vulnerabilities using software composition analysis. Within the Libraries tab of Contrast TeamServer, the Contrast Security platform tracks software composition analysis, reports on libraries that are used, and assigns a score based on the risk associated with the use of the particular library. Unlike techniques that simply monitor components, Contrast Assess understands how the component is used by the application and which classes are used from each library. This allows Contrast Assess to have improved granularity for determining exploitability of a weakness in a library. Contrast integrates with third-party development tools to support bug tracking and ticket creation for vulnerabilities to be addressed. Contrast Protect "fixes" vulnerabilities in both custom code and third-party libraries by preventing them from being exploited at runtime, without code changes. Unlike external systems that require tuning, Contrast Protect is in the application to identify attacks and avoids defending against threats that are not present.

Table 3 - Contrast Security Platform Capabilities Alignment of Applicability to PCI Secure Software Standard v1.0 Findings

PCI SECURE SLC APPLICABILITY DETAIL

The following table details the applicability of Contrast Assess and Contrast Protect to the Secure SLC Requirements and Assessment Procedures.

ID	REQUIREMENT	CONTRAST SECURITY PLATFORM APPLICABILITY
3.1	Critical assets are identified and classified.	Contrast Assess can discover and maintain an inventory of system components that make up the applications it assesses. This includes microservices, APIs, applications, lines of code, libraries, component technologies, architecture, and back end connections. The architecture components and back end connections include all servers that host components of the applications that are in use, including databases. Contrast Assess generates simple diagrams that illustrate the application's major architectural components. This information may be useful to the payment entity's efforts for maintaining a comprehensive system inventory.
3.2	Threats to the software and weaknesses within its design are continuously identified and assessed.	Contrast Assess and Contrast Protect can be used by the software vendor as part of their mature process for identification, assessment, and monitoring of software threats and design weaknesses. Contrast Assess and Contrast Protect continuously evaluate the software for vulnerabilities that can be exploited or design flaws that could result in data compromise. Contrast Assess accounts for software inputs/outputs, process/data flows, and decision points and supports the vendor's analysis of how these can be exploited by an attack. Contrast Assess is aware of the entire code base of the application where it is deployed, including how the use of third-party, open source, or shared components or libraries, APIs, services, and applications are used and provides insights about how or if these components can be leveraged for an attack. Contrast Assess provides a report of all design flaws and vulnerabilities in a recorded inventory. Contrast Assess is continuously monitoring the application and can account for changes to existing threats or design flaws, as well as the emergence of new threats and design flaws. Contrast Assess provides an inventory of all open source components used by the application and provides insights as to known vulnerabilities that exist for the open source components with guidance on how to mitigate the vulnerability.

ID	REQUIREMENT	CONTRAST SECURITY PLATFORM APPLICABILITY
4.1	Existing and emerging software vulnerabilities are detected in a timely manner.	When included as part of the vendor's documented mature processes for testing software, Contrast Assess can be used to test for the existence and emergence of vulnerabilities within software. For supported frameworks, Contrast Assess is an appropriate tool for detecting applicable vulnerabilities. Contrast Assess can be used to provide continuous monitoring for the application throughout the entire SDLC, including after release. Contrast Assess does not rely on point in time testing strategies for detection and identification of vulnerabilities either existing or emerging. Rather, it is continuously evaluating the security of the application through normal functional testing and operation. Contrast Assess can be used to assess the security of the entire code base for the application, including detecting vulnerabilities in third-party, open source, and shared components and libraries. Contrast Assess provides objective assessment of the application code without the need for highly skilled security personnel to qualify vulnerabilities. The continuous assessment of the application code base results in an inventory of identified vulnerabilities with guidance provided for remediation of vulnerabilities. The information about the vulnerability includes relevant and context-specific exploitability of the vulnerability based on the way that the application uses the code, library, third-party module, shared component, or open source code.

Conclusion

Coalfire has determined that Contrast Assess and Contrast Protect are valuable tools for helping organizations identify, classify, and address vulnerabilities and protect their software throughout the SDLC. Contrast Assess assists with the development of secure code by identifying issues earlier in the lifecycle and offering remediation paths. The capabilities of Contrast Protect allows the software to be protected with greater fidelity than what is offered by traditional software security approaches (for example, Web Application Firewalls).

Resources and References

https://www.owasp.org/index.php/Category:OWASP_WebGoat_Project

https://www.csoonline.com/article/2130877/the-biggest-data-breaches-of-the-21st-century.html

https://www.forbes.com/sites/thomasbrewster/2017/09/14/equifax-hack-the-result-of-patched-vulnerability/#1a649fb95cda

https://www.contrastsecurity.com

https://www.contrastsecurity.com/security-influencers/question-i-understand-sast-and-dast-and-how-to-use-them-but-what-is-iast-and-why-does-it-matter

Bibliography

PCI SSC. (2016, May). PA-DSS 3.2. Retrieved from Payment Card Industry Standards: www.pcisecuritystandards.org

PCI SSC. (2018, May). Payment Card Industry (PCI) Data Security Standard, v3.2.1. (P. SSC, Ed.) Retrieved from PCI Security Standards Council: www.pcisecuritystandards.org

PCI SSC. (2019, January). PCI Secure Software Standard v1.0. (P. SSC, Ed.) Retrieved from PCI Security Standards: www.pcisecuritystandards.org

PCI SSC. (2019, January 16). PCI Security Standards Blog Just Published New PCI Software Security Standards. (L. K. Gray, Editor, P. SSC, Producer, & PCI SSC) Retrieved from PCI Security Standards: https://blog.pcisecuritystandards.org/just-published-new-pci-software-security-standards

Contrast Security provides the industry's most modern and comprehensive Application Security Platform, removing security roadblocks inefficiencies and empowering enterprises to write and release secure application code faster. Embedding code analysis and attack prevention directly into software with instrumentation, the Contrast platform automatically detects vulnerabilities while developers write code, eliminates false positives, and provides context-specific how-to-fix guidance for easy and fast vulnerability remediation. Doing so enables application and development teams to collaborate more effectively and to innovate faster while accelerating digital transformation initiatives. This is why a growing number of the world's largest private and public sector organizations rely on Contrast to secure their applications in development and extend protection in production.

240 3rd Street
2nd Floor
Los Altos, CA 94022
Phone: 888.371.1333
Fax: 650.397.4133

Contrast SECURITY

contrastsecurity.com