International Journal of Emerging Trends in Computer Science and Information Technology
ISSN: 3050-9246 | https://doi.org/10.63282/3050-9246.IJETCSIT-V4I3P126
Eureka Vision Publication | Volume 4, Issue 3, 255-263, September - 2023

Original Article

Over-the-Air Update Validation for Connected-Vehicle Infotainment Systems

Krishnan Visweswaran
Independent Researcher, USA.

Abstract - Over-the-air (OTA) software updates are a vital way to keep and upgrade connected-vehicle infotainment systems without the need for physical service visits and have been developed to allow software updates, feature enhancements, security patches and bug fixes to be delivered remotely. Network connectivity issues, corrupted update packages, software incompatibility, interrupted upgrade processes, or failure to properly recover the system can impact the update and availability of the system. The paper suggests a systematic framework for OTA update validation for connected–vehicles infotainment systems that supports the evaluation of OTA update integrity, installation reliability, software compatibility, functional correctness, and recovery performance. The framework also includes update package authentication, pre-installed package compatibility testing, installation monitoring, post-installation functional testing and rollback verification in different network settings and failure scenarios. The evaluation methodology takes into account important metrics such as update success rate, installation time, failure detection rate, rollback success rate, recovery time, and functional test coverage. Comparative evaluation includes comparing the results with the traditional validation methods to look for improvements in failure detection, recovery effectiveness and overall update reliability. The proposed framework seeks to enable systematic and repeatable validation of automotive OTA updates, which will enhance the quality of their software, the operational resilience of the vehicle and the dependability of the connected-vehicle infotainment system.

Keywords - Over-the-Air (OTA), Connected Vehicles, Infotainment Systems, Software Integrity, Compatibility Testing, Failure Detection, Functional Testing, Update Reliability.

1. Introduction

1.1. Background of Connected-Vehicle Infotainment Systems

Today, connected-vehicle infotainment systems are sophisticated computers with navigation, multimedia entertainment, smartphone connectivity, voice control, internet access, and vehicle information systems. They are based on embedded software, communication networks and connections to various electronic control units (ECUs), enabling them to provide function reliability and responsiveness. [1] With the growing connectivity of vehicles, infotainment systems are undergoing regular software updates, security patches and additions. This means it is becoming a major challenge for the automotive industry to remain software reliable, compatible and available throughout a vehicle's life.

1.2. Role of Over-the-Air (OTA) Software Updates

Over the air (OTA) software updates will allow automakers to remotely and wirelessly provide and install software updates without the necessity of the vehicle being physically present at a service center. [2] OTA mechanisms provide for wireless communication to enhance features, fix bugs, provide security patches, and maintain software. But, secure package delivery, integrity checks, compatibility checks, controlled installation and post package update functional testing are needed for successful deployment. Proper validation of the OTA updates must be implemented for proper software updation, to maintain the functions of the infotainment, and to recover the updates in case of interruptions without affecting the system reliability.

1.3. Challenges in OTA Update Validation

There are a number of issues in the process of validating OTA updates such as: unstable network connectivity, incomplete downloads, corrupted packages, incompatible software versions, install failures, and rollback failures. Failure can also happen after installation due to the differences in the hardware configuration, software dependencies, and operating environments. Also, the various vehicle configurations need repeatable test scenarios with measurable evaluation criteria, to validate the updates. Therefore, a detailed validation process should evaluate package integrity, installation success, functionality, recovery when failures occur, and system availability under both normal and failure circumstances, in order to enhance the reliability of connected-vehicle infotainment updates.

2. Background and Related Work

Table 1. Comparison of Existing OTA Update Validation Techniques

Validation technique	Primary focus	Limitation
Package integrity verification	Detecting corrupted packages	Does not verify post-update functionality

Digital signature validation	Package authenticity	Does not assess software compatibility
Installation testing	Successful software deployment	May not identify functional regressions
Compatibility testing	Hardware and software dependencies	Requires diverse configurations
Rollback testing	Recovery after update failure	Does not cover all failure scenarios
Integrated validation	End-to-end update reliability	Requires comprehensive test coverage

2.1. OTA Update Architecture in Connected Vehicles

Over-the-air (OTA) update architecture of connected vehicles involves software update servers, wireless communication networks, vehicle connectivity modules, update management systems and electronic control units (ECUs). [3] The architecture makes it possible to remotely securely transmit, verify, install and monitor software packages. For automotive OTA, these mechanisms often include package authentication, software version management, software installation scheduling, and software recovery. For infotainment systems, the system architecture needs to facilitate efficient delivery of the updates, but must not compromise features of the existing system nor affect the compatibility of the vehicle's hardware and software requirements. The correct validation of these architectural elements is crucial to diagnose communication problems, installation discrepancies and recovery from updates failures in various operating conditions.

2.2. Existing OTA Update Validation Techniques

Existing techniques for validating OTA update include software integrity validation, compatibility checking, installation testing, functional testing, network interruption test, and rollback validation. [4] Package verification mechanisms provide cryptographic checks and authentication processes for determining the corruption or unauthorised use of update files, and compatibility testing is used to validate software versions, hardware configurations and system requirements. Installation testing involves checking the execution and completion of the update and checking the behaviour of the system to replace the software. Functional validation tests if the infotainment functions after installation, and failure injection recovery testing checks the behavior of the system during interrupted downloads, installation errors, and unsuccessful updates. The assessment of these techniques separately, however, may fall short of demonstrating the reliability of the evolution overall and it is important to find a way of validation that integrates them: a methodology that includes installation performance, failure detection, functional correctness and recovery effectiveness.

3. System Architecture and Problem Formulation

Table 2. Connected-Vehicle Infotainment System Components and Functions

Component	Primary function	Validation requirement
OTA update server	Distributes software packages	Verify package availability and version
Connectivity module	Transfers update packages	Assess network reliability
Update manager	Controls update execution	Verify workflow correctness
Package verification module	Checks integrity and authenticity	Detect invalid packages
Infotainment operating system	Supports software execution	Verify compatibility and stability
Recovery manager	Restores a working software version	Evaluate rollback and recovery

3.1. Architecture of the Connected-Vehicle Infotainment System

The connected-vehicle infotainment system consists of a central processing unit, operating system, multimedia applications, navigation service, communication interfaces, storage elements, and vehicle network interfaces linked to pertinent electronic control units (ECUs). [5] The connectivity module facilitates remote OTA update servers to communicate using cellular or wireless networks, and the update management component manages packages received, verified, installed and reported. The architecture must ensure software components and vehicle interfaces are coordinated to avoid disruption of critical infotainment operations during update operations. A thorough knowledge of these components and their interdependencies is essential to help identify potential failure points and define proper OTA update validation process.

3.2. OTA Update Delivery and Installation Workflow

The OTA update workflow is started if an update server publishes an update package, and the vehicle receives an update notification via the vehicle's connectivity interface. The package is downloaded, then checked for integrity and authenticity, compatibility with the software and preparation of the installation environment. After passing the pre-installation tests, the update manager runs installation and tracks its progress, completion and system behaviour. The updated software has been tested after installation to ensure that all necessary infotainment features are still present and are working properly. In cases of failure, the system should contain a proper recovery or rollback procedure to restore a known working software version, as determined by the software update architecture and supported. Each stage of the workflow needs to be validated to detect failures and validate the overall update reliability.

3.3. Failure Scenarios and System Requirements

One of the problems that need to be resolved in OTA update validation is that it needs to be able to detect if any of the following issues arise during the download, authentication or rollback process: interrupted download, corrupted update package, failed authentication, incompatible software versions, not enough storage, interrupted installation, or failed rollback. [6] Other scenarios include network connectivity that is unstable, forced shut down of the systems, conflicts with other systems, and infotainment system function failures after installation. The validation system needs to offer secure package verification, compatibility check, reliable installation tracking, fault detection, fault recovery and post update functional testing. Some of the critical factors are software integrity, minimised service disruption, ability to preserve vital system functions and safety ability to identify and recover from failed software updates. These requirements provide the foundation for creating measurable metrics and an experiment repeatable methodology for connected-vehicle infotainment systems.

4. Proposed OTA Update Validation Framework

4.1. Overall Validation Framework Architecture

The proposed OTA update validation framework provides a combined validation process by integrating the following functions into one: update package verification, compatibility check, monitoring of the update, post-OTA functional testing, recovery validation, security evaluation. [7] The framework downloads software update packages from the OTA server and tests the various phases of the software update process with preset scenarios and performance factors. Validation results are captured to determine failures, system reliability and to see if the update meets the requirements. This modular architecture will allow for repeatable tests in normal operating environments and simulated scenarios of failure, allowing for systematic testing of connected-vehicle infotainment updates.

4.2. Update Package Integrity and Authenticity Verification

Package integrity and authenticity verification that software downloaded is not altered, corrupted, or distributed by an unauthorized party. The suggested system runs cryptographic hash verification, digital signature validation, check of certificate trust, and consistency of package metadata before the installation of the package. [8] Test scenarios are corrupted files, invalid signatures, unauthorized packages, incomplete downloads and mismatched version information. The validation process keeps a log of packages that are accepted as valid packages or rejected as invalid packages and quantifies the effectiveness of identifying invalid packages for updates. The checks are meant to reduce the number of failures due to unsound or damaged packages, and to prevent unauthorized software installation.

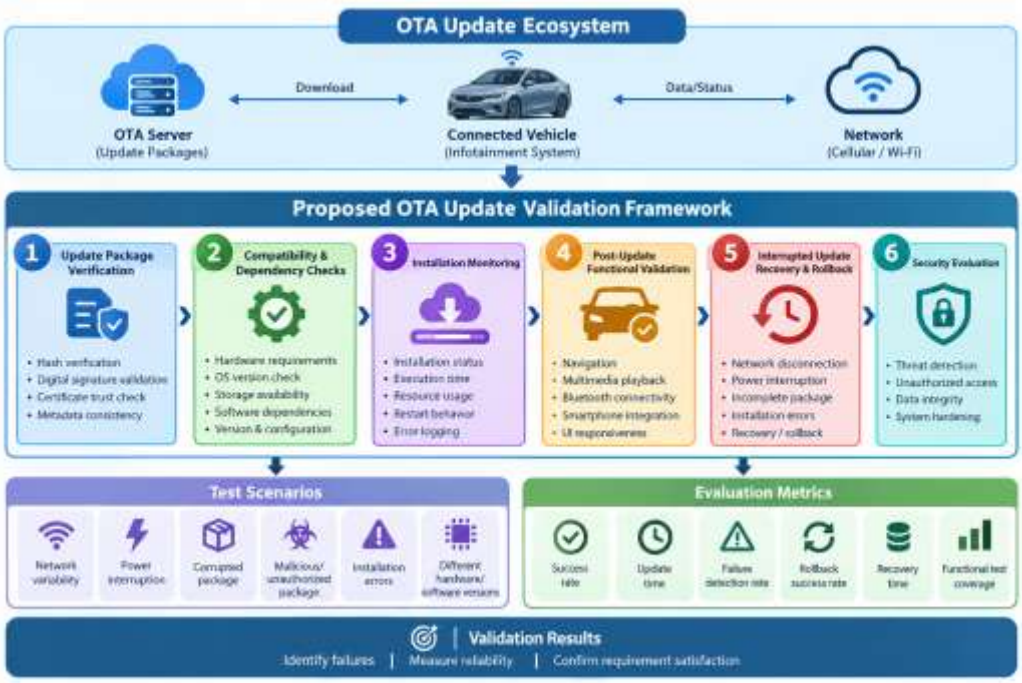


Figure 1. Proposed OTA Update Validation Framework for Connected-Vehicle Infotainment Systems

4.3. Pre-Update Compatibility and Dependency Checks

The pre-update compatibility and dependency checks ensure that the software package that is intended to be installed on the target infotainment platform is suitable for the platform. The framework covers hardware, operating system versions, storage, software dependencies, package versions and configuration constraints. Unsupported hardware, dependency missing, lack of storage space, incompatible software versions, and conflicting application components are examples of test cases. [9] The

validation module detects missing requirements and aborts the installation if there are any critical requirements that are missing. This approach minimizes installation failures and helps to ensure uniform software operation in all configurations of infotainment.

4.4. Installation and Post-Update Functional Validation

Functional Validation: This is used during installation and after updating to determine if the software package is successfully installed and if the infotainment system continues to function as expected. The framework tracks the status of installation, duration of the installation, use of system resources, restarting behavior and errors reported during the installation process. [10] Once installed, the essential functions are tested such as navigation, multimedia playback, Bluetooth, smartphone integration and the responsiveness of the user interface. Test results are compared to pre-defined acceptance criteria to verify that tests are failing as expected, installations are incomplete or behavior is unexpected. At this stage measurable evidence is obtained indicating the correctness of the installation and functional reliability after the update.

4.5. Interrupted Update Recovery and Rollback Validation

Interrupted Update Recovery and Rollback Validation test the system's ability to recover in the event of failure during package download, installation or system restart, and validate the rollback process. The framework uses controlled failure scenarios network disconnection, power interruption simulation, no package transfer and installation errors – in an appropriate test environment. Then it will delve into if the update process continues safely, whether it is able to boot from a valid recovery point, or if it can be restored to a working software version if supported by the architecture. Success in rollback, recovery time, data integrity and restoration of vital infotainment functions are key measurements. These tests are used to ascertain that the update mechanism does not leave the system in a software "inconsistent state" in the event of failure.

5. Experimental Design and Methodology

5.1. Test Environment and Infotainment Hardware Configuration

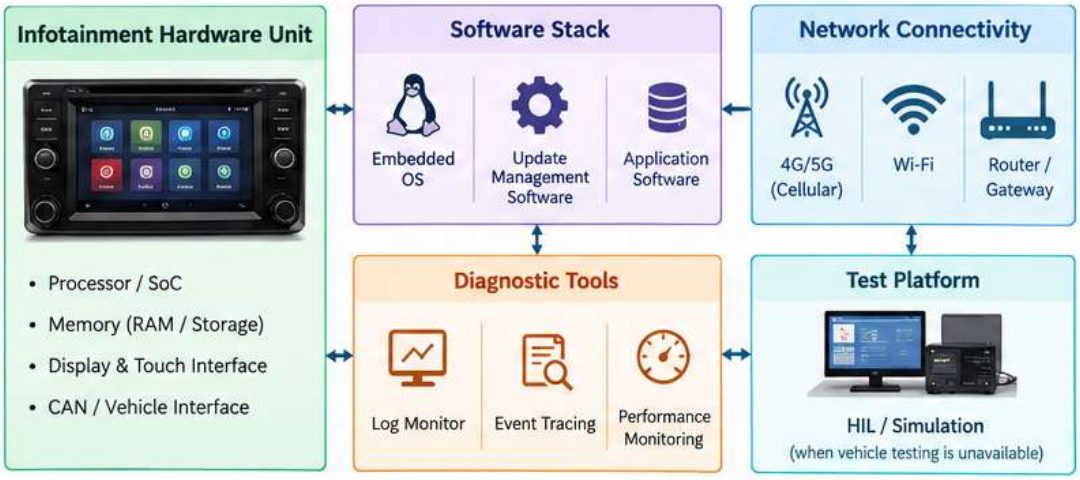


Figure 2. Connected-Vehicle Infotainment Test Bench Architecture

The experimental setup is set up to test OTA update validation in controlled connected-vehicle infotainment environment. [11] A test platform consists of an infotainment hardware unit or representative test bench, an embedded operating system, an update management software, a network connectivity interface, and diagnostic software to check the system's behavior. Before testing, the system configuration, storage size, software requirements and system resource usage are documented. When the vehicle testing is not possible, a hardware-in-the-loop test or a controlled, software simulation can be used to replicate failure scenarios and update operations. This setting allows packages to be verified, installed and tested repeatedly, as well as recovers after updating and tested.

5.2. OTA Update Test Scenarios and Experimental Conditions

Experimental scenarios are designed to evaluate OTA behavior in normal, degraded and failure scenarios. Successful packages downloaded, valid authentication, software installation is compatible, and successful post-update functional test are examples of normal conditions. [12] Some failure scenarios consist of corrupted packages, invalid digital signature, outdated versions, lack of disk space, interruption of download, installation failure, and failed recovery. The bandwidth of the network, latency, packet loss, and loss of connectivity can be controlled to test their impact on update performance. The scenarios are then run against a number of predefined test procedures and acceptance criteria, providing the ability to measure the success of the update, failure detection, installation time, and effectiveness of recovery.

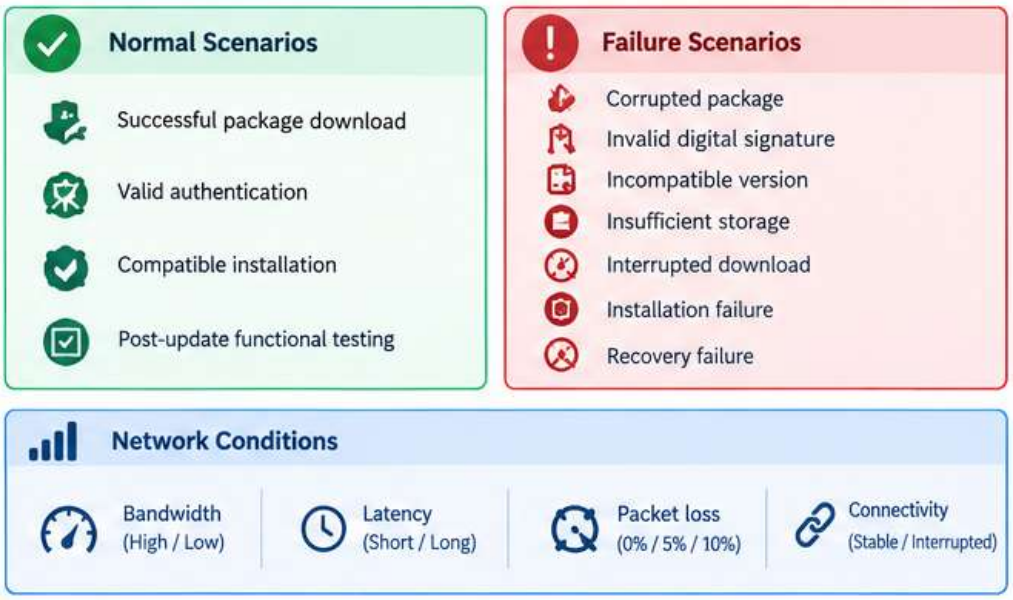


Figure 3. OTA Update Scenarios under Variable Network Conditions

5.3. Baseline Validation Methods and Proposed Approach

The proposed framework for validating OTA updates is assessed using a baseline approach, which reflects traditional approaches to validation, including simple package checking, checking of installation status, and limited testing of functionality after the update. [13] Equivalent test environments, software packages, network conditions and failure scenarios are used to ensure a fair comparison when evaluating both approaches. Success rate of updates, failure detection rate, success rate of rollbacks, recovery time, installation time and functional test coverage are all metrics used to evaluate the product. The results are analyzed to find out if the integrated framework offers improvements in failure identification, reliability in failure recovery and overall validation coverage. The comparative assessment is reliable and repeatable when the same procedure is used and repeated experiments are conducted.



Figure 4. Comparative Evaluation of Conventional and Proposed OTA Validation Methods

6. Evaluation Metrics

6.1. Update Installation Success Rate

The update installation success rate is the percentage of the OTA update attempts that are successfully installed and meet the pre-defined installation acceptance criteria. [14] It's defined as the number of successful installations divided by the number of update attempts plus multiplied by 100. Successful completion should include verification of the installation of the updated software and the system being brought to an expected operational state. This metric allows for comparing the reliability of updates on various software versions, hardware configurations and network conditions.

6.2. Update Completion Time and Network Transfer Latency

Update completion time includes the time it takes to download, verify, install and validate an OTA software package. Network transfer latency is the time it takes for the update to be transferred from the update server to the vehicle. The measurements are taken with different bandwidth, latency, packet loss and connection conditions to see how these factors affect the performance of the update. The timing information from the various experimental scenarios can be used to determine the communication delay, installation delay and improvements needed to make connected-vehicle infotainment information more efficient.

6.3. Package Integrity and Authentication Accuracy

Package Integrity and package authentication correctness checks whether the framework detects corrupted, modified or unauthorized software update packages for installation prior to installation. [15] The evaluation consists of valid and invalid test packages to assess correct Acceptance and Rejection decisions. The integrity verification success, authentication success, false acceptance rate, and false rejection rate are relevant metrics. These checks are based on cryptographic hash verification and digital signature validation. These calculations give an indication of the ability of the validation system to prevent that invalid or unauthentic software packages proceed to installation.

7. Results and Analysis

Table 3. Comparative OTA Update Validation Performance under Different Test Scenarios

Test scenario	Baseline success rate (%)	Proposed framework success rate (%)
Normal network	90	98
High network latency	82	95
Packet loss	75	92
Interrupted download	70	94
Installation interruption	65	90
Incompatible package detection	85	99
Rollback and recovery	72	96

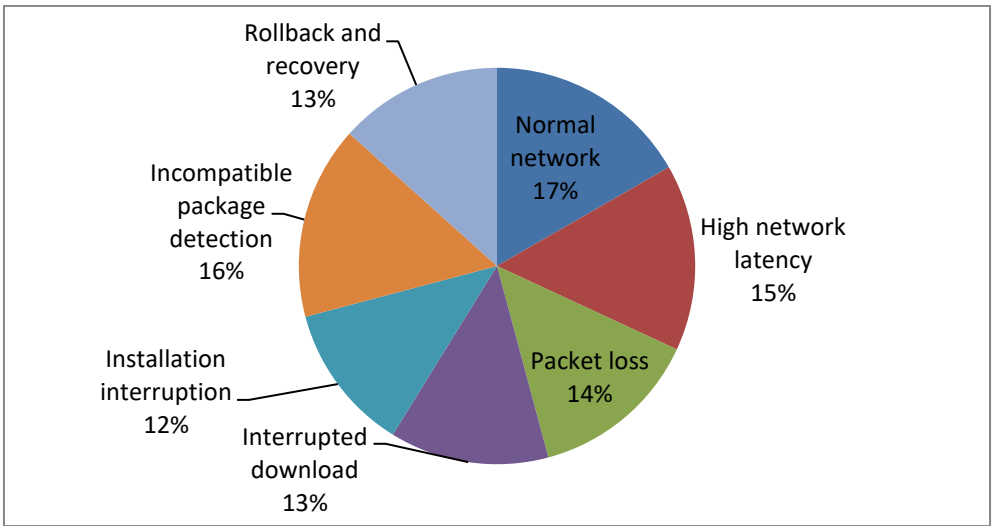


Figure 5. Graphical Comparative OTA Update Validation Performance under Different Test Scenarios

7.1. OTA Update Installation Performance

The installation performance for OTA updates is measured by the percentage of successful installations, total update completion time, and the functionality of the installed system, in various test scenarios. [16] The analysis takes into account successful installations, failed partial updates, installation failures, and unanticipated post-installation behavior. The performance measures are compared between software versions and hardware configurations to determine factors that impact on software updates. The results can show if the validation framework is able to consistently detect installation failures and confirm successful software deployment. Further experimental testing is needed to quantify the improvements in performance and determine the effectiveness of the proposed approach.

7.2. Validation Accuracy under Different Network Conditions

OTA update behavior is studied for various bandwidths, network latencies, packet losses, and connectivity interruptions to validate accuracy under various network conditions. The analysis assesses the completion of package downloads, the results of the verification of packages for updates, the success of the package installation, and the effectiveness of the framework in detecting communication-related failures. The performance of updates and reliability of validation are evaluated and compared between the results of network scenarios and stable conditions and degraded conditions. Special focus is paid to the partial downloads, interrupted transfers, and recovery after reconnection. Such results can be used to single out network conditions that need further validation or update management processes to be optimized.

7.3. Recovery Performance during Interrupted Updates

The recovery performance is tested by inserting controlled interruption in package download, installation and system restarting operations. The analysis compares how long it takes to recover, its success rate, restoration of vital infotainment functions and uniformity of software after recovery. [17] Test results are compared with each possible interruption point to

determine the situations where the recovery mechanisms are working well and when they are not. The assessment also checks if the framework is able to identify incomplete installations and also if it is able to check if the software was previously functioning and restore it if rollback is supported. The results show the robustness of the system to failures stemming from updates.

7.4. Detection of Corrupted and Incompatible Update Packages

Test packages with corrupted data, invalid digital signatures, unauthorized changes, incompatible software versions, and dependency requirements that are not met are used to test the detection capability of the proposed framework. Analysis indicates whether or not the package was properly accepted or rejected prior to installation. Detection rate, false acceptance rate and false rejection rate test to determine the effectiveness of the package verification and compatibility checking. The outcomes are useful in determining areas of vulnerability in integrity checks, authentication, and pre-installation assessment. If the software can be detected prior to installation, then any inappropriate package will not be able to move through the process, thereby lowering software failure risks following installation.

7.5. Comparative Evaluation of Validation Methods

The comparative evaluation compares the proposed integrated validation framework to an approach of using equivalent test environments, update packages, network, and failure scenarios as a baseline. [18] The comparison of performance is done by the success rate for updating, failure detection rate, roll back success rate, recovery time and functional test coverage of the updates. The analysis compares the performance of each method with respect to three aspects: the ability to detect failures on software updates, the ability to validate software functionality, and the ability to recover from a software installation that was interrupted. Outcomes will indicate if package verification, compatibility testing, monitoring installation and recovery validation is a measurable benefit over traditional testing practices. Improvements, if any, should be backed up by reported experimental data.

7.6. Statistical Analysis of Experimental Results

Repeated experimental trials are conducted and statistical analysis is used to evaluate the consistency and significance of the observed OTA validation results. The completion time for updates, the time to recover and other continuous measures of performance can be summarized using descriptive statistics such as a mean, standard deviation, minimum value, and maximum value. Validation methods can be compared, appropriate statistical analysis tests should be chosen based on data distribution and experimental design (paired t-test or Wilcoxon signed-rank test). Confidence intervals and effect sizes can offer further information on the size and the uncertainty of observed differences. This analysis is helpful to the objective interpretation of the results and to making the results more reproducible.

8. Discussion

8.1. Interpretation of Experimental Findings

The experimental results are analyzed to evaluate the capability of the proposed OTA update validation framework to enhance the update reliability, failure detection and recovery performance. Success rates of the installation, accuracy of the package verification, correctness of the post-update functionality, and recovery results under various operating conditions are being analyzed. Comparisons are made with baseline validation methods to identify benefits of multiple validation stages incorporated into a single process. The findings should also identify the most difficult test scenarios from the software update failure detection perspective, and the most important validation components. The analysis of these interpretations should be based on the experimental data gathered.

8.2. Reliability and Safety Implications for Connected Vehicles

To ensure the stability, security and availability of connected-vehicle infotainment systems, updates must be validated with reliability. Failure to verify packages, incompatible software installation into the vehicle, or a failure in recovery may affect the navigation, multimedia, smartphone connectivity, and other driver-focused features. The proposed approach mitigates these risks by verifying authenticity, assessing compatibility, monitoring installation and performing functional validation after updating. While infotainment validation helps to ensure vehicle dependability, it should not replace separate safety validation of safety critical automotive systems. Systematic OTA testing throughout the software development process can help to minimize failures associated with software updates and improve software maintenance.

8.3. Limitations and Threats to Validity

Validation approach proposed might be constrained by the variety of hardware configuration, software versions, network conditions and failure scenarios covered in the experiments. Simulations or lab test benches results may not be fully representative of the actual operating conditions of the vehicle, including long-term system behavior and interactions between various vehicle components. The quality of the test cases, reliability of measurement tools, and representativeness of the baseline method are also important for the accuracy of the evaluation. In addition, the results of a small number of infotainment platforms might not be representative of automotive architectures. Future work should examine a larger number of hardware configurations, more failure scenarios and real-world controlled experiments to augment the generalizability and usefulness of the results.

9. Conclusion and Future Work

9.1. Summary of the Proposed Validation Approach

This paper introduces a systematic framework for validating the integrity and authenticity of OTA software updates, the compatibility, reliability during software update, the correctness of software functionality after updates, and failure recovery for connected-vehicle infotainment systems. It combines the pre-installation checks, installation monitoring, functional testing and rollback validation into one overall evaluation process. The experimental procedure proposed is to measure the performance of the updates under the following conditions: normal operation, network variability, controlled failure scenarios. This integrated solution offers a structured basis to determine OTA update failure detection, review recovery strategies and enhance software deployment reliability in connected-vehicle infotainment.

9.2. Key Research Contributions and Findings

The main research work that was carried out is the creation of an integrated validation approach that considers several aspects of the OTA update lifecycle instead of just a successful installation. The framework is based on a unified experimental framework that includes security verification, software compatibility testing, functional evaluation, and recovery evaluation. The effectiveness can be measured by the success rates for installations, failure detection, package authentication, failure to roll back, recovery time, and functional test coverage. Comparative experimental analysis can be used to determine how much better the proposed approach is at achieving a higher degree of validation completeness and failure identification than baseline approaches. Only when experimental measurements are taken and analyzed, should specific improvements or results of research be reported.

9.3. Future Research Directions

The proposed framework can be expanded for future research by testing the framework with a large scale across various embedded OS and automotive software architectures and infotainment hardware platform. Further studies could include automatic generation of test cases, intelligent failure analysis, predicting future failure hazards after applying the update and ongoing tracking of system behavior after the update. Hardware-in-the-loop testing and controlled real-world vehicle testing integrate with the evaluation to boost reliability of updates under realistic operating conditions. Additional work can also consider secure multi-component updates, dependency-aware deployment techniques, and advanced recovery techniques for interrupted installs. The developments could help to create more comprehensive, scalable and reliable OTA update validation for next generation of connected vehicles.

References

[1] Halder, S., Ghosal, A., & Conti, M. (2020). Secure over-the-air software updates in connected vehicles: A survey. Computer Networks, 178, 107343.

[2] Mahmood, S., Nguyen, H. N., & Shaikh, S. A. (2022). Systematic threat assessment and security testing of automotive over-the-air (OTA) updates. Vehicular Communications, 35, 100468.

[3] Kirk, R., Nguyen, H. N., Bryans, J., Shaikh, S. A., & Wartnaby, C. (2023). A formal framework for security testing of automotive over-the-air update systems. Journal of Logical and Algebraic Methods in Programming, 130, 100812.

[4] Ghosal, A., Halder, S., & Conti, M. (2022). Secure over-the-air software update for connected vehicles. Computer Networks. https://doi.org/10.1016/j.comnet.2022.109394

[5] Bazzi, A., Shaout, A., & Ma, D. (2023). MT-SOTA: A merkle-tree-based approach for secure software updates over the air in automotive systems. Applied Sciences, 13(16), 9397.

[6] Lamssaggad, A., Benamar, N., Hafid, A. S., & Msahli, M. (2021). A survey on the current security landscape of intelligent transportation systems. IEEE Access, 9, 9180–9208. https://doi.org/10.1109/ACCESS.2021.3050038

[7] Luo, F., Zhang, X., Yang, Z., Jiang, Y., Wang, J., Wu, M., & Feng, W. (2022). Cybersecurity testing for automotive domain: A survey. Sensors, 22(23), 9211.

[8] Kuppusamy, T. K., DeLong, L. A., & Cappos, J. (2018). Uptane: Security and customizability of software updates for vehicles. IEEE vehicular technology magazine, 13(1), 66-73.

[9] Checkoway, S., McCoy, D., Kantor, B., Anderson, D., Shacham, H., Savage, S., ... & Kohno, T. (2011). Comprehensive experimental analyses of automotive attack surfaces. In 20th USENIX security symposium (USENIX Security 11).

[10] Koscher, K., Czeskis, A., Roesner, F., Patel, S., Kohno, T., Checkoway, S., ... & Savage, S. (2010, May). Experimental security analysis of a modern automobile. In 2010 IEEE symposium on security and privacy (pp. 447-462). IEEE.

[11] Miller, C., & Valasek, C. (2015). Remote explo itation of an unaltered passenger vehicle. Black Hat USA, 2015(S 91), 1-91.

[12] Regenscheid, A. (2017). Platform firmware resiliency guidelines (No. NIST Special Publication (SP) 800-193 (Draft)). National Institute of Standards and Technology.

[13] Cappos, J., Samuel, J., Baker, S., & Hartman, J. H. (2008, October). A look in the mirror: Attacks on package managers. In Proceedings of the 15th ACM conference on Computer and communications security (pp. 565-574).

[14] Samuel, J., Mathewson, N., Cappos, J., & Dingledine, R. (2010, October). Survivable key compromise in software update systems. In Proceedings of the 17th ACM conference on Computer and communications security (pp. 61-72).

[15] What is OTA in automotive? Over the air updates explained., Rambus, 2022. Online. https://www.rambus.com/blogs/ota-updates-explained/

[16] Fuchs, A., Krauß, C., & Repp, J. (2016, May). Advanced remote firmware upgrades using TPM 2.0. In IFIP international conference on ICT systems security and privacy protection (pp. 276-289). Cham: Springer International Publishing.
[17] Cooper, D. A., Polk, W. T., Regenscheid, A. R., & Souppaya, M. P. (2011). SP 800-147. BIOS Protection Guidelines.
[18] Shavit, M., Gryc, A., & Miucic, R. (2007, August). Firmware update over the air (FOTA) for automotive industry. In 14th Asia Pacific Automotive Engineering Conference.
[19] Nasr, A., Ghoneima, M., & Abdullah, B. A. (2022, March). Automotive software self reprogramming OTA. In 2022 13th International Conference on Electrical Engineering (ICEENG) (pp. 76-80). IEEE.
[20] Villegas, M. M., & Astudillo, H. (2020). OTA updates mechanisms: a taxonomy and techniques catalog. In XXI Simposio Argentino de Ingeniería de Software (ASSE 2020)-JAIIO 49 (Modalidad virtual).