Clustering Functional Sections Using Finer Grained ACDC

A Case Study Document

By Pengxiang Zhu and Tianhang Liu

Team Member
Zhaoqi Zhu
Tianhang Liu
Pengxiang Zhu

Secure-related Architectural Decision Picked

The following document illustrates the process of recovering the security decision of introducing "CredentialHandler" for user credential management using the proposed method.

Background

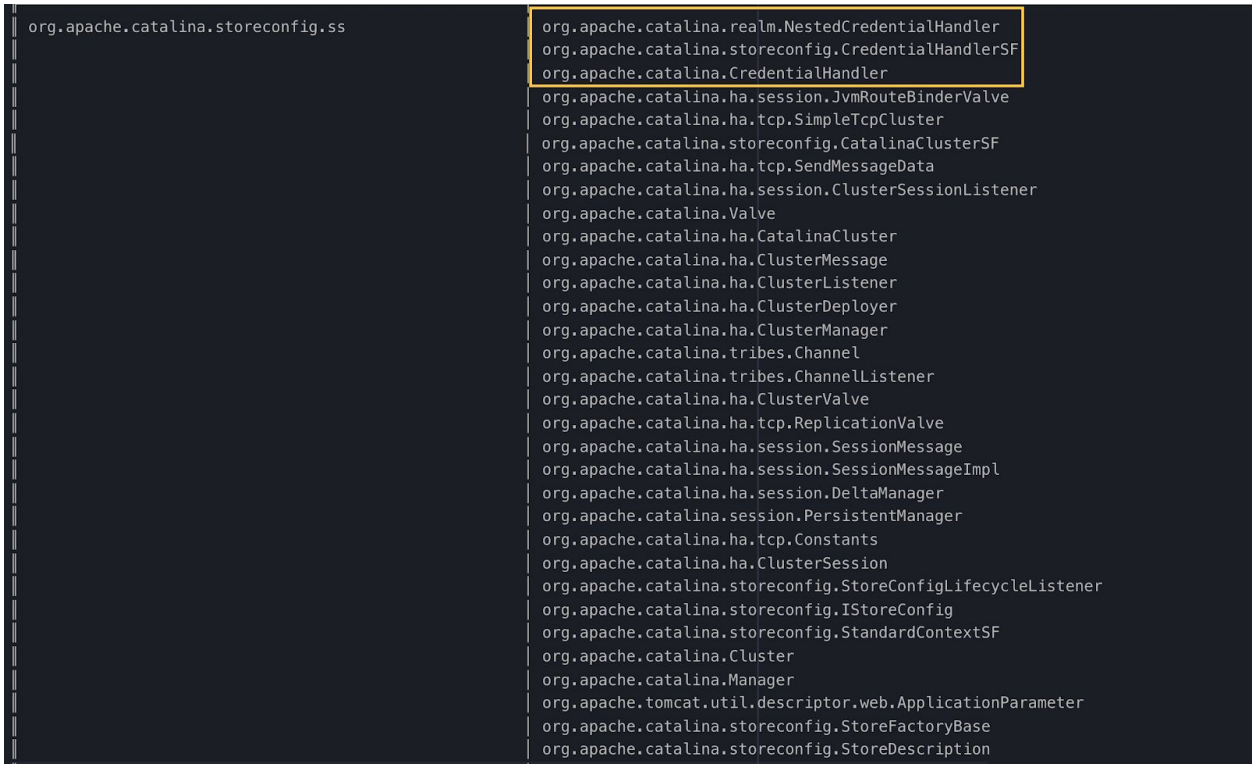
Realm component in Tomcat "represent a "database" of usernames, passwords, and roles (similar to Unix Groups)"[1], and it provides functions for user authentication. For versions before 8.0, Realm component handles user authentication within itself, including credential comparison functions, and this decision has been related with several vulnerabilities (CVE-2009-0580, CVE-2011-1184, and CVE-2016-0762). After version 8, Tomcat abstracts the credential comparison functions into a separate CredentialHandler component, which makes the user authentication a multi-component process. The CredentialHandler component is critical for the user authentication functions provided by Realm component that "a **CredentialHandler** element MUST be nested inside a Realm component"[1]; however, results from ACDC did not help us to identify this design decision.

Results from original ACDC

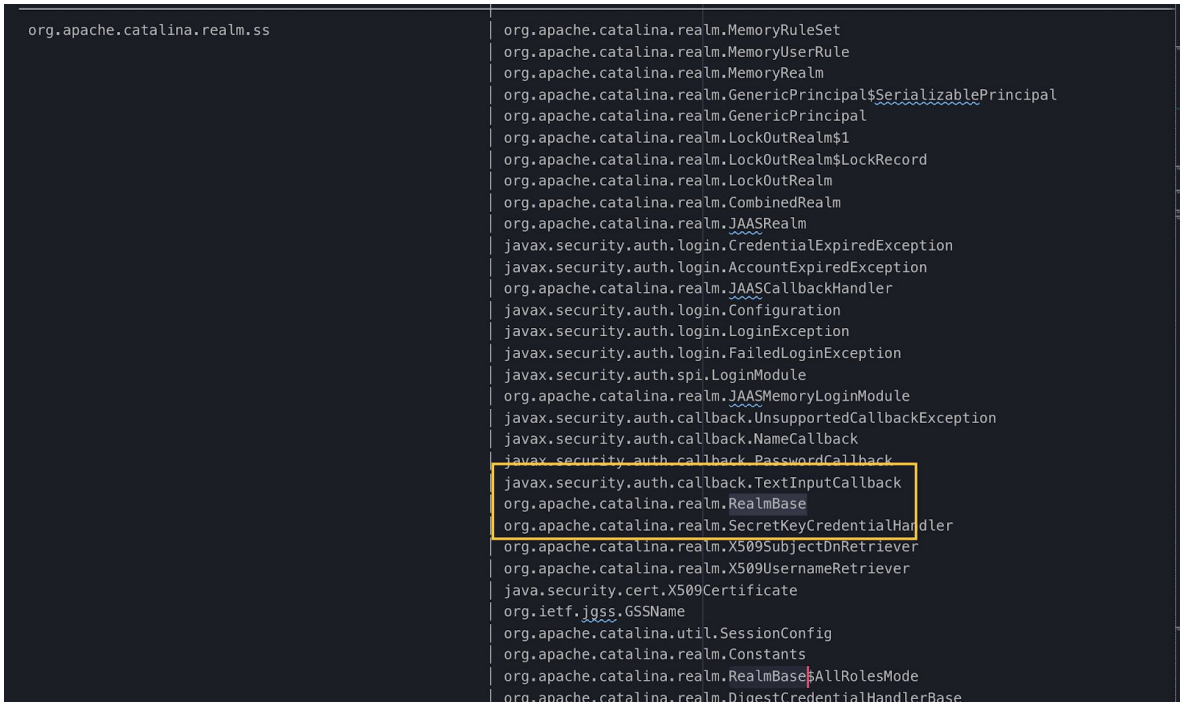
Our teams find out following problems from ACDC results:

1. While CredentialHandler and RealmBase (an implementation of Realm) are closely related for user authentication, they are distributed into different clusters, which prohibits us from recovering this design decision.

CredentialHandler is clustered to "org.apache.catalina.storeconfig.ss".

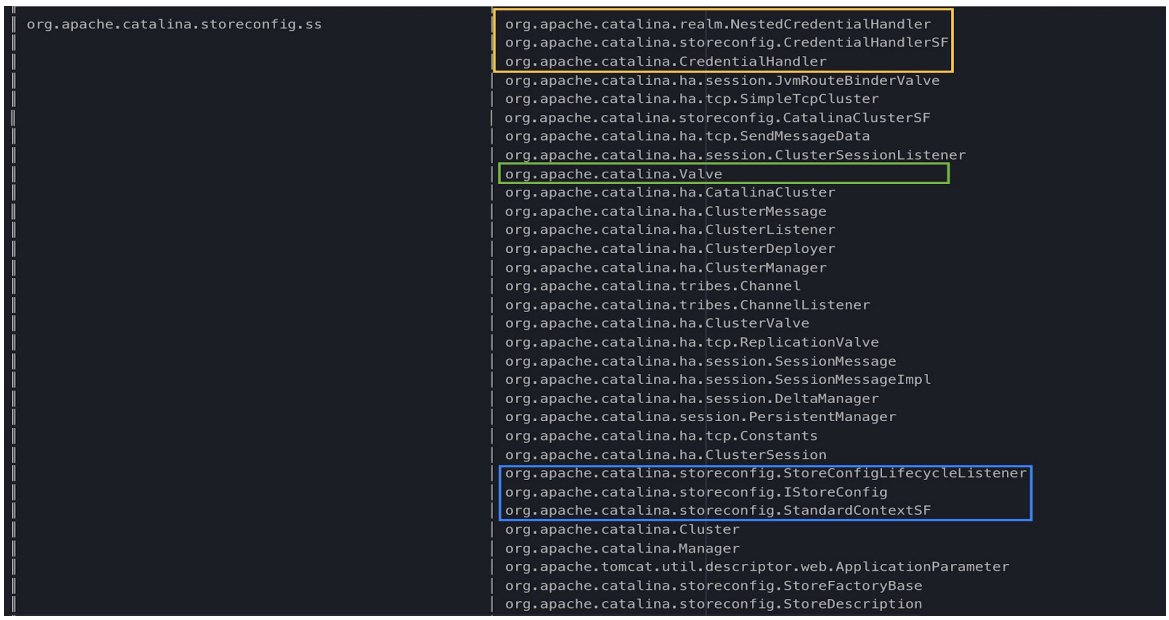


RealmBase is clustered to “org.apache.catalina.realm.ss”



2. ACDC is based on a structural approach and the basic units are files/classes[2], this makes it hard to understand the sub-class-level implementation of the system. For example, an ACDC component can have multiple concerns or “functional sections,” but ACDC does not identify or separate them in any ways. Different function an element provides may be buried by the general function of its cluster.

The highlighted elements form different functional sections.



From the information we gained from the analysis, we try to extend the ACDC recovery technique to recover design decisions that span multiple components, and please refer to rationale document for the full reasoning and description of the extension.

The process of discovering this design decision with our method

By analysing the existing implementation artifacts, we design 5 traits combined together to discover this specific security-related architectural decision - User Credential management.

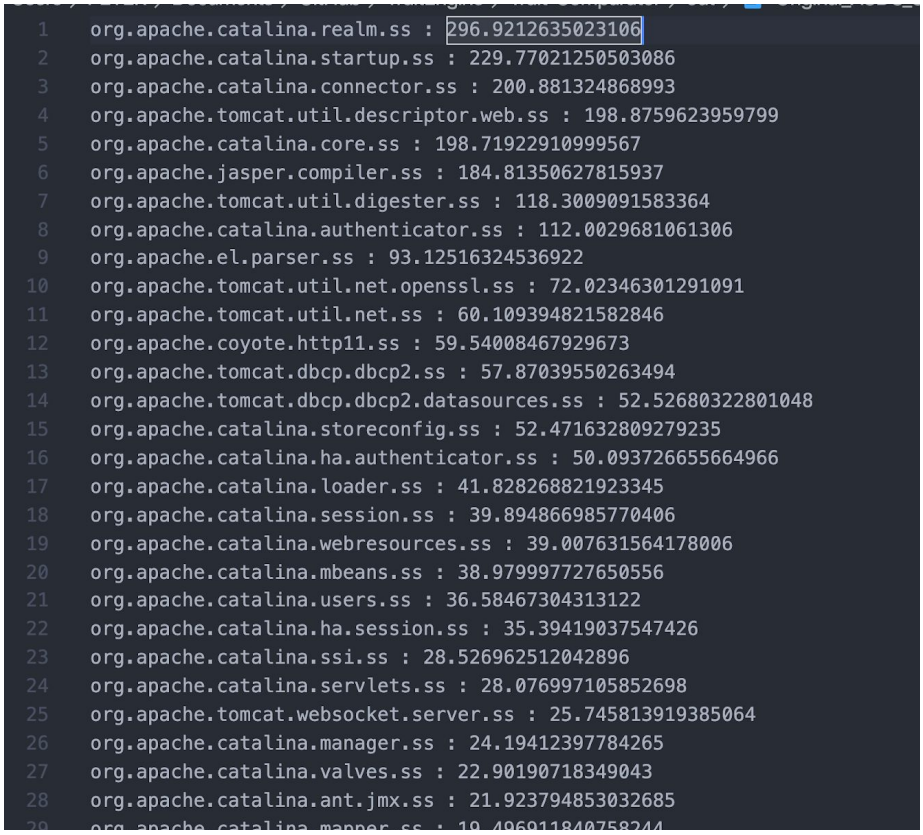
(1) Check if the function contains “authenticate”.
(2) Check if the function contains “compare”, “match”.
(3) Check if the function contains “password”, “credential”, “username”.
(4) Check if the function contains “digest”, “md”.
(5) Check if the function has return type of “java.security.principal”.

If a function matches a trait, that function will be added with 1 point and each function can obtain a maximum of 6 points in total.

The result of our method

After we go through the pipeline that is mentioned in the rational document, we acquire (1) a ranking of the original ACDC clusters and (2) a ranking of the functional sections with their corresponding original ACDC clusters. These two rankings will be shown below.

(1) Ranking of the original ACDC clusters with their score



The score of each cluster can be interpreted as the amount of code related to the user credential management. The increment rate of the score is linear, which means that if a cluster A's score is twice as cluster B's score, cluster A has twice as much user credential-related code as cluster B. In our result above, cluster "org.apache.catalina.realm.ss" has the highest score, which means that it is the most relevant cluster with respect to user credential management and contains the most amount of code implementation related to the security decision we picked.

(2) Ranking of the functional sections with their corresponding original ACDC clusters in Tomcat Version 8.5

The result above shows functional sections that is closely associated with the user credential management, ranked by their scores. It also shows the original ACDC clusters that each function section is mapped back to. For example,"org.apache.catalina.realm.RealmBase.ss11" can be mapped back to original ACDC clusters "org.apache.catalina.realm.ss", org.apache.catalina.storeconfig.ss", "org.apache.catalina. Authenticator.ss" because "org.apache.catalina.realm. RealmBase.ss11" contains functions that comes from these three clusters. In addition, by looking into "org.apache.catalina.realm.RealmBase.ss11", we can see the introduction of Credential Handler component in Tomcat Version 8.5, which does not exist in Tomcat Version 6.0. It will be discussed in detail in the following paragraph.

Function section of "org.apache.catalina.realm.RealmBase.ss11" in Tomcat Version 8.5

Function section of "org.apache.catalina.realm.RealmBase.ss9" in Tomcat Version 6.0

As you can see here, "org.apache.catalina.realm.RealmBase.ss11" in Tomcat Version 8.5 and org.apache.catalina.realm.RealmBase.ss9" in Tomcat Version 6.0 are considered as the same

function section with respect to different versions because it is obvious to notice that two function sections contains many of the same functions like authenticate(), getPassword() and getDigestCharset(). However, in version 8.5, it reveals the introduction of CredentialHandler component by having functions like CredentialHandler.matches() and NestedCredentialHandler.matches(). Besides, in version 6.0, some digest-related functions are removed such as digest() and MD5Encoder.encode() As we take a look into the actual code implementation, it turns out that in version 8.5, all these message digest functions are extracted into a new component - Credential Handler. In this case, by looking at the functional sections of “org.apache.catalina.realm.RealmBase.ss11” and “org.apache.catalina.realm.RealmBase.ss9”, we are able to successfully identify the security decision of introducing `CredentialHandler` for user credential management.

Reference:

[1] Y. Shapira, J. Arcand, and F. Hanik, “Apache Tomcat 8 Architecture,” Apache Tomcat, 7Oct-2019. [Online]

[2] V. Tzerpos and R. Holt, “ACCD: an algorithm for comprehension-driven clustering,” *Proceedings Seventh Working Conference on Reverse Engineering*.