**2023年全国大学生信息安全竞赛**

**作品报告**

**作品名称： 基于ABY3-VSSS的安全隐私密码管理系统**

**电子邮箱： 2574728338@qq.com**

**提交日期： 2023年6月12日**

填写说明

1. 所有参赛项目必须为一个基本完整的设计。作品报告书旨在能够清晰准确地阐述（或图示）该参赛队的参赛项目（或方案）。

2. 作品报告采用A4纸撰写。除标题外，所有内容必需为宋体、小四号字、1.5倍行距。

3. 作品报告中各项目说明文字部分仅供参考，作品报告书撰写完毕后，请删除所有说明文字。(本页不删除)

4. 作品报告模板里已经列的内容仅供参考，作者可以在此基础上增加内容或对文档结构进行微调。

5. 为保证网评的公平、公正，作品报告中应避免出现作者所在学校、院系和指导教师等泄露身份的信息。

**目 录**

[摘要 2](#_Toc971633848)

[第一章 作品概述 5](#_Toc603906430)

[1.1 研究背景 5](#_Toc1118474668)

[1.2 研究现状 7](#_Toc1082096305)

[1.3 作品简介 9](#_Toc1124765595)

[1.4 特色创新 10](#_Toc1263453531)

[1.5 研究意义 12](#_Toc155289202)

[1.6 应用前景 14](#_Toc72114699)

[第二章 作品设计与实现 16](#_Toc362236817)

[2.1 作品整体设计 16](#_Toc2077694686)

[2.2 系统详细模块设计 18](#_Toc1614474317)

[2.2.1 协议参与方介绍 18](#_Toc616766642)

[2.2.2 注册密码秘密共享模块 21](#_Toc877600421)

[2.2.3 秘密恢复登录模块 25](#_Toc324857957)

[2.2.4 Web分布式服务器验证模块 26](#_Toc1588517623)

[2.2.5 服务器数据泄露溯源模块 30](#_Toc714738363)

[2.2.6 服务器动态调整与自检模块 31](#_Toc522170454)

[2.3 安全协议方案介绍 33](#_Toc1560445779)

[2.3.1 ABY3与Cheetah 33](#_Toc2094389800)

[2.3.2 可变的秘密共享模式 35](#_Toc817197398)

[第三章 作品测试与分析 40](#_Toc401953320)

[3.1 安全性分析与证明 40](#_Toc329996814)

[3.2 原型测试说明 42](#_Toc1481332423)

[3.3 系统各流程UI展示 44](#_Toc834587234)

[3.3.1 系统UI展示 44](#_Toc1797870958)

[3.3.2 日志记录 45](#_Toc189240956)

[第四章 创新性说明 47](#_Toc231543236)

[4.1 用户端启用全新的密码管理方式 47](#_Toc825704609)

[4.2 服务器端集成多种秘密共享方案 47](#_Toc1559400854)

[4.3 系统安全鲁棒性扩展 48](#_Toc74593682)

[4.4 实现本方案应用SecretFlow框架 49](#_Toc2111119349)

[第五章 总结 50](#_Toc383551055)

[参考文献 51](#_Toc678500112)

# 摘要

**动机**

随着互联网技术的高速发展，每个人都在网络中传输和存储了大量敏感信息，如身份、生物、金融信息等。为保护用户隐私不受侵害，网站一般采用密码核对的方式进行用户身份验证。因此，对于每位用户的隐私安全而言，构建完备有效的密码管理系统具有十分重要的意义，不当的密码管理所导致的密码泄露将对用户的个人和财产安全造成严重威胁。然而，目前用户密码和个人信息泄露等严重侵害用户隐私的事件层出不穷。2013年，Adobe公司的1.5亿用户个人信息和加密密码遭到泄露；2018年，Facebook和Google发生了用户个人信息泄露事件；2021年，LinkedIn（领英）社交平台遭受黑客攻击，导致7亿用户密码被盗取。

经过调查与研究，我们发现造成上述用户密码及隐私泄露事件的主要原因与当前流行的用户密码管理体系紧密相关，用户与浏览器交互部分（Google公司事例）和网站服务器存储部分（Adobe公司，LinkedIn公司事例）都存在着一定的安全性隐患。

在用户与浏览器的交互过程中，大多数用户倾向于使用浏览器的自动保存密码功能。在当前互联网体系中，浏览器相较于用户占据着绝对的主导地位，浏览器会将用户保存的密码同步到云端，而用户却无从确定浏览器是否严格遵守隐私规定，或者留有侵犯用户隐私的后门。此外，将用户密码存储于浏览器云端也增加了用户密码遭受攻击的风险。

在与Web网站服务器交互时，为了进行用户登录认证，网站服务器需要存储用户密码的哈希值或加盐哈希值。然而，目前的服务器架构通常使用单台服务器存储用户的个人信息和密码。这样的存储方式存在着鲁棒性与可扩展性不足的问题：单个服务器被攻陷就会造成大量用户隐私泄露；单点故障则有可能直接导致网站登录功能崩溃。

基于目前主流的密码管理系统所存在的问题，为打破用户与浏览器之间的不对等关系；完善网站服务器端的密码存储方式，进而降低由密码管理系统设计不当而带来的安全风险，我们提出了**基于ABY3-VSSS的安全隐私密码管理系统**。

**功能**

与传统的密码管理技术不同，我们采用了**基于信息论安全的秘密共享方案**来搭建用户-浏览器密码管理系统与网站服务器端密码存储架构。对于用户-浏览器的密码管理，用户可以选择是否引入可信第三方，与用户和浏览器共同执行ABY3秘密共享方案；或是选择传统的用户-浏览器两方模型共同执行Cheetah秘密共享方案。我们的模型将根据用户选择的不同秘密共享方案将用户密码共享为秘密份额，并将秘密份额多方存储，在登录时自动恢复。在网站服务器端，我们通过集成不同的秘密共享方案，如：Shamir，Brickell，加性秘密共享和Semi2k-SPDZ等秘密共享方案，设计了可变的秘密共享模式——Variable Secret Sharing Schemas，简称VSSS，将用户密码的原文作为秘密信息进行秘密共享，在服务器端存储到分布式服务器中，登录验证时自动恢复。我们实现了性能模式和安全模式来适应不同的应用场景。

考虑到秘密共享方案都是基于分布式架构的，为了保证分布式系统的鲁棒性和可扩展性，我们相应地集成了秘密份额动态调整、数据泄露溯源以及被劫持参与方检查等功能模块，并对其进行了性能优化。

在我们的设计中，网站服务器端能够更加安全和鲁棒地存储密码，降低了浏览器云端存储所带来的攻击风险，同时面对“好奇”的浏览器，我们的设计也能完全保护真实密码信息的隐私不可见。我们为系统设计集成的安全检查，数据泄露溯源，动态调整等模块和多种模式策略的选择，进一步提升了系统的安全性与实用性。

**创新点**

1. **我们的系统打破了用户与浏览器之间的不对等关系：充分保护用户密码的隐私和机密性，**我们设计的系统打破了用户与浏览器之间的不对等地位，通过将注册的网站密码在用户、浏览器以及可能的可信第三方中进行多方秘密份额存储，使得“好奇”的浏览器无法查看到用户真实密码。
2. **我们的系统极大程度降低了浏览器云端存储所带来的攻击风险：**我们的系统会在用户使用浏览器自动存储功能时，仅交付给浏览器一份秘密份额，即使浏览器云端受到了攻击，攻击者获得的也只是秘密份额中的一份。
3. **分阶段多角度：**我们的系统设计主要分为两个阶段：用户与浏览器阶段和网站服务器端阶段。在用户与浏览器场景中，我们采用了ABY3协议与Cheetah协议，集成了三方模型与两方模型。在网站服务器端，我们集成了Shamir、Brickell、加性秘密共享和Semi2k-SPDZ等秘密共享方案，来应对不同场景的需求。
4. **多种可选模式：**在用户浏览器交互部分，我们提供了灵活的三方模型与更令人放心的两方模型。在服务器端的设计中，我们提供了效率更高的性能模式和安全性更强的安全模式，两种模式下，我们能够抵御目前流行的针对加密算法的攻击策略，避免单个服务器故障或被攻击的问题。
5. **我们的系统设计具备出色的鲁棒性和可扩展性。**我们的系统可扩展适应不同场景，追溯敏感数据的泄露，秘密份额的动态调整，还能提供实时的安全性检测和恢复机制，以应对可能的攻击和入侵，也能够避免单点故障等潜在问题。
6. **全新国产安全多方计算框架：我们的系统设计开发使用了最新的SecretFlow框架。**SecretFlow“隐语”框架是面向全球开发者正式开源的可信隐私计算框架，于2022年7月公布，是由蚂蚁集团历时6年自主研发的通用隐私计算技术框架，相对于PySyft等其余框架，具有便捷实用，与深度学习算子解耦度高等优点。

**实用性**

我们的系统具有良好的实用性，为用户提供了友好便捷的密码管理功能而无需理解系统的底层运作过程。我们的系统具备密码恢复和多种模式选择的特性，以满足用户与网站服务器的不同需求。我们的系统在安全性和应用性能方面取得了良好的平衡，既保护了用户数据的隐私和机密性，又提供了高效的密码管理，模拟用户所发出的恢复秘密份额请求在网络状态畅通的情况下可以在1秒内从系统中得到响应并完成密码验证工作，这使得系统在实际应用中具有很强实用性。我们的系统具有很好的扩展性，我们可以将该系统设计为浏览器插件适配到任意的浏览器之中。

**关键词：密码管理；秘密共享；混合模式；VSSS；隐私保护**

# 第一章 作品概述

## 研究背景

根据最新发布的《中国互联网发展状况统计报告》，我国网民数量已经达到了惊人的10.51亿人，总域名数也已经超过了3380万个。同时，使用台式电脑和笔记本的网民数量也高达了3.5亿人次。不可否认的是，更多用户将电子设备连接到互联网，意味着越来越多的人将自己的隐私数据接入了互联网，这对个人隐私来说具有极大的安全风险。2021年《中华人民共和国数据安全法》的颁布，也彰显了国家愈发高度重视数据安全。习近平总书记曾强调：“要加强关键信息基础设施安全保护，增强数据安全预警和溯源能力。”因此，如何保护用户个人隐私和数据安全已经成为了一个极为重要的课题。在涉及个人隐私信息保护的问题中，用户的密码扮演着非常重要的角色。密码可以说是保护用户隐私的“最后一道屏障”，使用密码可以有效地保障用户的个人隐私信息不被泄露。因此，密码安全问题也需要得到更加高度关注，对密码管理系统的设计也要与时俱进。

在用户访问互联网的过程中，需要创建和管理越来越多的在线账户，包括社交媒体、电子邮件和各种应用程序，在线账户需求日益增长；为了提高账户的安全性，密码要求也日益复杂，这些密码通常需要包含大写字母、小写字母、数字和特殊字符，并具有一定的长度要求。面对海量的用户名及其对应的复杂密码，用户很难记住所有密码。为了提高用户在数字化时代的体验感，能够快速、轻松地访问和使用各种在线服务，减少时间和精力来处理登录过程，浏览器自动存储和自动填充密码功能应时而生，大大提高了在线服务的效率和便利性。

在保证用户的便利性之外，安全性需求也需保障。针对安全性需求，前人的研究成果主要集中在以下方面，包括：提供用户是否保存密码的权限控制、提供注册过程中生成随机强密码的功能、对存储的密码数据进行加密、帮助用户辨识当前网站的安全性并进行安全警示等。

**然而，现有的用户--浏览器--网站服务器的密钥管理系统仍存在不可忽视的安全性问题和潜在高风险问题。**

第一方面，在用户浏览器阶段存在密码管理的安全隐患。我们将用户--浏览器场景建模如图1-1-1所示：

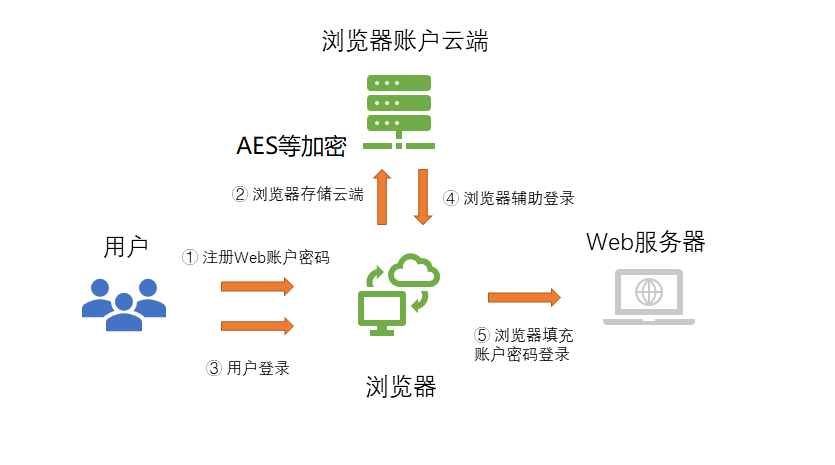


图 1-1-1：用户--浏览器建模图

在当前流行的场景中，当用户在浏览器上注册想要访问的网站账户与密码后，并选择了“自动保存密码”选项，浏览器会获取密码，并对其进行加密处理，以降低存储时被破解和截获的风险。之后，浏览器会将加密后的用户密码存储在浏览器账户的云端。接下来，当用户希望登录特定网站时，只需输入其用户名。此时，浏览器便会辅助用户从云端获取已加密的用户信息，并自动填充账户密码，从而实现自动登录。如果我们使用上述浏览器自动存储密码的便捷功能，尽管云端密码数据使用加密方式保存，仍有极高的可能性遭受攻击者的攻击，比如：穷举攻击、离线字典攻击以及差分攻击等等，导致用户账户被盗或信息泄露。

**在这个过程中，我们还将面临一个关键性问题，即我们必须完全信任浏览器。**让我们来看一下从Edge浏览器隐私政策中摘录的一段内容：“需要注意的是，Edge浏览器会自动保存和填充密码等个人信息，以提供便捷性。”并且，我们在更多的技术文档中发现：“Microsoft Edge相关数据同步，包括账号密码、收藏等信息，均会同步到云端服务器进行同步。”作为用户，我们必须意识到这意味着我们需要对浏览器保持完全信任，但是浏览器可能会收集、存储和使用我们的个人数据。所以，我们需要谨慎评估使用浏览器同步功能的利弊，也必须权衡方便的数据同步和个人信息安全之间的关系，并决定是否愿意将敏感数据存储在云端。我们同样应该密切关注相关服务提供商的隐私政策和安全措施，以确保我们的个人信息得到充分的保护。

也就是说，尽管我们解决了过多密码难以记忆等问题，但前提是我们必须完全信任Edge、Chrome等现行的浏览器。显然，浏览器一方占据着绝对的优势，我们无法确定他们是否拥有AES密钥或是否能够轻松获取我们的密码。同时，恶意软件或恶意网站很可能利用浏览器自动填充功能的漏洞或弱点，来获取用户的密码数据，从而进行钓鱼攻击、身份盗窃或其他形式的欺诈行为。

第二，在网站服务器一端，同样存在安全隐患。网站服务器将大量用户的密码明文或哈希值存储在单个服务器上，一旦攻击者发起攻击，仅需攻陷一个服务器就可截获大量的用户密码信息或者导致网站登录崩溃，对用户的隐私和安全性造成极大威胁，这种密码管理方式**缺乏一定的鲁棒性**。

现有的用户--浏览器--网站服务器的密钥管理系统，不论是在浏览器端还是在网站服务器端，针对密码的加密处理方式和存储管理密码的方案都较为单一，**缺乏一定的可扩展性**，难以应对密码管理的不断演变和增加的需求。因此，上述密钥管理系统存在的风险亟待解决，一种更为安全的密码管理系统的提出迫在眉睫。

针对当前浏览器自动存储和自动填充密码功能存在的问题，避免浏览器不诚实和Web网站服务器被攻击而造成的用户密码泄露，我们尝试通过在用户--浏览器--网站服务器的模型中运用**秘密共享方案**，设计一种更**具备安全性、普适性、鲁棒性、可扩展性**的密码管理系统。我们的最终目标是将我们的系统集成为**浏览器插件的形式**，在不同的浏览器中**普适地**使用。我们还扩展出多种功能，包括数据泄露溯源、劫持自检模块等等，以应对多方面的安全威胁。

## 研究现状

目前，如何管理大量的密码来保护在线账户的安全，降低安全风险，构建密码管理方案，已经成为信息安全领域的一个重要研究方向。针对用户密码的安全性保护，出现了一些在Web上提供远程密码管理服务的实体系统，这些系统旨在提供更安全和便捷的密码管理体验。例如：

1. LastPass：一款广泛使用的跨浏览器平台的在线密码管理器，使用强加密算法保护用户密码，并提供多因素身份验证等安全功能；
2. 1Password：它提供安全的密码存储和自动填充功能，用户可以通过主密码或生物识别登录来访问密码库；
3. KeePass：它提供跨平台支持和本地存储密码数据库的能力，采用强加密算法保护用户密码，并允许用户自定义密码库的设置。

这些密码管理系统通过采用加密算法、多因素身份验证和安全存储技术等手段，致力于保护用户密码的安全性。它们还提供便捷的密码填充功能，减少了用户记忆和输入密码的负担。此外，一些密码管理系统还提供额外的功能，如密码生成器、安全笔记和安全共享等，以提供更全面的密码管理解决方案。

然而，这些密码管理系统仍存在一定的安全风险，这些密码管理系统通常将用户的密码存储在云端服务器上，如果服务器发生故障或遭受攻击，可能导致用户密码的泄露或丢失，同时，密码管理系统的客户端应用程序也可能存在安全漏洞，可能被攻击者利用来窃取或篡改用户密码数据，使用加密算法的密码管理也可能成为下一个浏览器方。

所以，当今的研究趋势可以概括为如何为密码管理系统采取更为严格的安全措施，包括：采用更强大和安全的加密算法，以增加密码数据的保密性和完整性；增强身份验证方式，密码管理系统可以探索用户名和密码之外更多的身份验证方式；引入密码学创新，提供更高级别的密码管理和隐私保护功能；增强云端安全性，减少云端攻击和数据泄露的风险。

未来的研究将继续关注密码管理系统的安全性和隐私保护，通过创新的密码学算法、身份验证机制和安全设计，提供更可靠和安全的密码管理解决方案，以应对不断发展的安全威胁和攻击技术，实现最大化保护用户的密码和隐私信息。

在密码管理系统所应用的多种密码学方案中，多为加密算法，包括哈希函数、对称加密算法和非对称加密算法等，而忽视了另一种具备较高安全性和扩展性的密码学方案——**秘密共享方案**，它多用于**密钥分配管理、数据保护、访问控制和多方安全计算**等。**它提供了一种强大的安全机制，使得多个参与者可以共同保护和管理敏感信息，而不依赖于单个实体的信任**。该密码学方案于密码管理系统应用中的发展仍存在一定的空白，然而，它却是解决浏览器和网站服务器遭受攻击泄露用户密码的一种有效手段，它在密码管理系统等应用中可以用于分布式地安全存储和管理用户的密码和敏感信息。

我们的灵感来源于对**Password Protected Secret Share方案**的研究，对于使用密码保护的秘密共享方案对公私钥进行管理[1,2,3,4,5]（以下简称PPSS方案）的研究开始于2011年，PPSS方案是使用密码进行认证保护的秘密共享方案，通过使用口令密码来认证保护秘密信息sc进行分布式秘密共享的方案。我们从中收到启发，密码管理系统其实可以不使用口令来加强秘密共享方案，而是直接将密码作为秘密信息进行秘密共享保护。我们目前阅读了大量PPSS方案相关的文献，并进行实际调研，发现很多PPSS前沿技术停留在理论部分，没有真正应用落地，我们尝试将PPSS方案的内在思想进行落地实现。

总体而言，密码管理系统的研究现状积极向前。通过对密码学技术的运用，密码管理系统逐渐可以提供更安全、便捷和可靠的密码管理解决方案；然而，仍然存在一些研究挑战和运用空白。我们的作品探究基于秘密共享方案，最大程度地避免浏览器以及网站遭受攻击泄露密码情况，提供更加强大和可信赖的解决方案。

## 作品简介

在我们的作品中，我们考虑利用多种不同的秘密共享方案，设计一个更具备安全性、普适性、鲁棒性、可扩展性的密码管理系统。我们设计的基于ABY3-VSSS的安全隐私密码管理系统如图1-3-1所示。

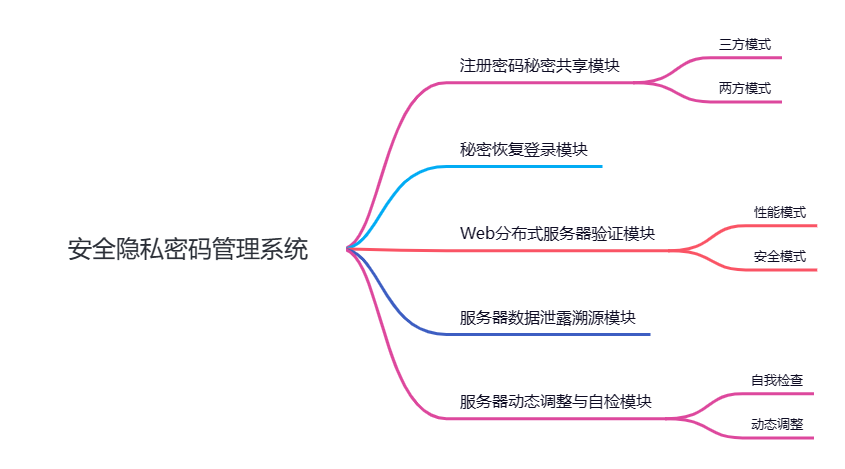


图1-3-1：系统架构思维导图

**我们的系统分为注册密码秘密共享模块、秘密恢复登录模块、Web分布式服务器验证模块、服务器数据泄露溯源模块、服务器动态调整与自检模块五个不同的模块**，实现了分阶段多角度的完整系统流程设计与实现。

注册密码秘密共享模块与秘密恢复登录模块应用于用户-浏览器的交互阶段。通过使用ABY3协议构建三方模型（或Cheetah协议构建两方模型），三方模型中，我们将密码分成三份秘密份额，分别存储在用户本机、浏览器和一个半诚实（诚实但好奇）的第三方中，只有获取其中的至少两份秘密份额，用户才可以复原出登录密码（两方模型中仅划分两份秘密份额，分存于用户本机与浏览器中）。这样的设计打破了用户与浏览器之间的不平等地位，让只拥有一份秘密份额的浏览器不具有在信息论层面上复原出原始密码的可行性，使得用户能够在真实密码对浏览器不可见的同时，兼容浏览器自动存储、自动登录等功能，不同于当前流行的使用AES等加密算法的密码管理系统，能够真正实现用户对自身密码的掌控（因为目前流行的密码管理系统类似于浏览器一方，同样掌控着加密密钥）。

Web分布式服务器验证模块应用于Web网站服务器端。通过运用可变的秘密共享模式，集成了Shamir、Brickell、加性秘密共享和Semi2k-SPDZ等可选的秘密共享方案，实现了网站服务器端对于用户密码原文的秘密份额的分布式服务器存储。我们将这些不同的秘密共享方案划分为性能模式与安全模式，使我们的系统可以扩展适用于不同的实际场景。

在作品设计中我们考虑到多方面安全威胁，还在Web网站服务器端集成了服务器数据泄露溯源、服务器动态调整与自检两个模块。服务器数据泄露溯源模块可以追溯数据泄露方；服务器动态调整与自检模块可以动态调整网站服务器端的服务器数目并定期更新秘密份额，自检模块使用优化后的组合方法检测应对可能出现的对分布式服务器劫持的情形，这两个模块进一步加强了实际应用场景中的系统安全性。在Web网站服务器端存储密码的过程中，不同于目前流行的加盐哈希技术，我们的系统实现了基于信息论的安全以及更好的鲁棒性与可扩展性。

综上所述，我们的作品一方面打破了用户与浏览器之间的不平等地位，实现了用户对自身账户密码的掌控，另一方面在网站服务器端提供了更安全的密码存储解决方案。系统还考虑了多方面的安全威胁，并引入了动态调整、数据泄露溯源和劫持自检等模块来增强系统的鲁棒性与安全性。

## 特色创新

主流的密码管理系统在用户浏览器交互阶段时，用户会使用浏览器的自动存储功能帮助存储密码，一旦浏览器存储密码的云服务器受到攻击、或浏览器主动获取用户存储的密码信息，那用户密码的隐私就会受到严重威胁。

在我们设计的方案中，使用ABY3协议或Cheetah协议将自己的密码分成三份或两份秘密份额，在浏览器中存储一份密码的秘密份额而不是完整密码，在用户的本地计算机存储一份秘密份额，第三份秘密份额可以存储在一个可信的第三方平台；只有获取两份秘密份额并进行复原，才有可能获取完整的密码。在这种方案设计中，只拥有一份秘密份额的浏览器没有获取用户密码完整的信息的可能，**使用秘密共享的密码管理方式为保证用户密码的安全性和隐私性提供了一个全新的实现思路。**

在服务器端，服务器端传统存储密码的方式是存储用户的密码明文、哈希值或加盐哈希值于单台服务器上，一旦这台服务器遭受攻击，攻击者就可截获大量的用户密码信息。我们的方案通过**在服务器上部署秘密共享的存储模式**，将每位用户的密码分成n份（n>=5）秘密份额，每台服务器仅存储用户密码的一份秘密份额，只有获取至少t（t>=3）份秘密份额后，网站服务器才能恢复出用户的原始秘密进行认证。使用多台服务器联合存储，攻击者至少需要攻破t台服务器的安全防御机制，获取一个用户密码的t份秘密份额才能恢复出用户的原始密码信息，从一台服务器到t台服务器，使用秘密共享的存储模式，极大地提高了攻击者的攻击难度。

此外，我们还**创造性地提出VSSS——可变的秘密共享模式**。即在服务器端集成多种可用的秘密共享方案，有些方案具有更高的效率、有些方案具有更强的安全属性，服务器可以根据需求，选择需要的方案进行部署。

虽然，我们作品中的密码管理方案相比于之前的管理方式更加的安全，但网站服务器仍然存在被攻击的风险。一旦某台服务器受到攻击，攻击者可能会发布或篡改所截获的秘密份额。为了快速锁定遭受攻击的服务器，我们**增加了数据泄露溯源模块**，在服务器存储每条秘密份额时连接上一个服务器编号哈希值，同时将其shuffle放入秘密份额之中，在混淆秘密份额的同时还能在数据泄露时可以快速确认是哪台服务器泄露数据。

如果某台服务器被攻击者劫持，攻击者可能会篡改存储的秘密份额信息，导致网站使用错误的秘密份额进行复原，进而导致复原失败或生成错误密码的情况。本作品还提出了一种优化后的组合方法，作为**服务器的自检模块**，用于帮助网站确认是否有服务器被绑架，并快速找出被绑架的服务器。

在作品开发实现的过程中，我们使用2022年7月公布，**由蚂蚁集团历时6年自主研发的通用隐私计算技术框架**[**SecretFlow**](https://github.com/secretflow/secretflow)**框架**。[SecretFlow](https://github.com/secretflow/secretflow)是最近发布的一个隐私保护数据分析和机器学习的统一框架，提供设将多方安全计算（MPC）、同态加密（HE）、可信执行环境（TEE）等隐私计算技术抽象为密文设备，将明文计算抽象为明文设备；使数据分析和机器学习工作流程能够表示为计算图；基于计算图的机器学习/数据分析的方法。我们使用这个框架主要利用框架中提供的秘密共享方案接口。

## 研究意义

在现实生活中，网络应用的安全保障尤其重要。在开发网络应用时，需要使用有效的安全协议（例如加密、两步验证和及时的安全更新）来防范攻击者入侵和数据泄露等安全风险。保护用户信息安全已成为网络应用开发的必要要求，而安全的网络访问体验也是确保用户隐私和避免数据丢失或被盗窃的关键所在。

最近的调查结果表明，有大量网民在多个应用中，包括网上银行和电子钱包等重要应用，都使用相同或者极其相似的密码，这种做法虽然能够减轻用户记忆密码的负担，但同时也会留下巨大的安全隐患。如果浏览器没有按照协议约定窃听了用户密码，或者用户本机被攻击者入侵，或者应用服务器被攻陷，攻击者就可以获取到用户的密码。在用户--浏览器--网站服务器安全密码隐私管理系统中，这种攻击会导致一个密码解锁多个应用的连锁反应，对用户造成难以估量的损失。

在2018年底，谷歌公司遭遇十万级别的Google+用户资料大规模外泄事件，导致该业务被关闭，并面临着相应的法律诉讼。而在2021年4月，国家信息安全漏洞平台(CNVD)曝光了谷歌浏览器存在的远程代码执行漏洞，该漏洞意味着攻击者可以利用未授权的情况下执行恶意代码，窃取用户的隐私。这一事件再次引起了人们对于隐私安全的担忧，同时也显现了即使是全球知名的浏览器公司，都很难抵御攻击者入侵以及源于企业内部的隐私外泄。因此，如果用户想将其密码安全交由浏览器进行保护，显然是不明智的。为了避免用户对浏览器存储密码的行为表示担忧，使用一种新型的密码隐私管理系统已经势在必行。

在当今的网络安全环境下，丰田和同花顺等知名企业都不可避免地受到攻击者入侵的威胁，导致用户密码泄露和隐私信息外泄。2019年4月1日，丰田公司表示攻击者入侵了其IT系统并访问了多家销售子公司的数据，其中包括了多达310万用户的个人信息。无独有偶，2020年4月2日，多地投资者的同花顺股票账户密码也集体泄露，恶意攻击者利用这些信息登录股民账户，并在清仓之后购买了济民制药使大量投资者遭受了巨大的损失。丰田和同花顺等知名企业均开发了应用来确保安全性，但这些应用也难以避免攻击者攻击和用户密码泄露等问题，**这清楚地表明了现存的用户--浏览器--网站服务器安全密码隐私管理系统存在诸多问题并需要进行改进。**只有采用更加细致和全面的技术手段，并加强用户的安全意识和管理，才能更好地保障用户密码的安全性及数据隐私的保护。

2018年国家网络宣传安全周期间，人民日报在微博上发布了一份个人信息保护指南，其中涵盖了网络安全的重要性、个人信息的定义和保护措施。根据指南的披露，大约54％的网民都曾经遇到过网络安全方面的问题，这提示了网络安全问题的普遍性和日益严重的态势。而在这些网络安全问题之中，个人信息泄露问题是极为严重的，据统计，大约28.5％的网民都曾经遭遇到过个人信息泄露的问题，这波及到了人们的隐私和个人权益，同时也对个人或者企业的安全稳定造成了巨大的影响。

2019年全国网民网络安全感满意度调查统计报告如图1-5-1所示（亿欧数据统计所得）：

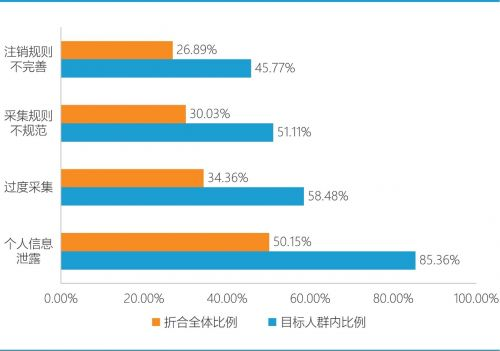


图1-5-1：2019年全国网民网络安全感满意度调查统计报告

上述讨论表明，无论是在国内还是全球范围内，用户--浏览器--网站服务器安全密码隐私管理系统存在着明显的设计缺陷。这些问题给攻击者提供了生存的空间，他们可能会通过利用这些密码隐私管理系统的弱点，成功地攻击网站和数据存储设备，导致大量用户的密码和隐私信息暴露。这不仅对个人隐私造成了威胁，而且对企业造成的经济损失也可能是无法预知的。

我们设计的基于ABY3-VSSS的安全隐私密码管理系统在很大程度上解决了当前用户--浏览器--网站服务器密码隐私管理系统中存在的不足。该系统在半诚实假设下保证了密码安全性，可以避免浏览器泄密以及应用服务器被攻陷时造成的用户密码泄露。相比于目前广泛应用的密码管理系统，我们的系统更具鲁棒性和可靠性。

这种安全隐私密码管理系统的另一个优点是它具有良好的普适性和可扩展性。用户可以选择使用密码份额三方存储而信任第三方，或者使用密码份额两方存储而不信任第三方。此外，根据浏览器自身的具体情况，还可以融合更多的秘密份额分配协议以提升系统的安全性。这为不同用户提供了更多的灵活性和自由选择，同时保护了其密码的安全和隐私。

总体来讲，我们的安全隐私密码管理系统是一个高效且安全可靠的密码管理工具，可以满足不同用户的需求，为密码隐私保护提供了强有力的支持，弥补了秘密共享方案在实际运用中的空白。我们将继续不断改进和优化这个系统，探索更好的密码隐私保护解决方案，以确保用户密码的安全和隐私。

## 应用前景

**我们设计的基于ABY3-VSSS的安全隐私密码管理系统应用了近年来密码学领域以及网络安全领域关于秘密共享的最新的研究成果**，可以充分保证用户的密码在互联网中存储的安全性，进一步保证了用户的个人隐私以及重要财产的安全

我们设计的ABY3-VSSS的安全隐私密码管理系统是一款具有**高安全性且高度用户友好**的密码管理系统。随着网络科技的不断发展，越来越多的用户倾向于使用轻量化、即插即用的浏览器插件。在全球知名的浏览器，如IE、谷歌等中，用户可以自定义安装插件，为使用网络服务提供更为便捷的操作方式。这也意味着安全隐私密码管理系统的部署环境极为丰富且容易。

据《中国移动互联网发展报告（2022）》显示，国内市场上检测到的应用数量已经超过了252万款，虽然其中并不是所有的应用都需要密码管理功能，如此之大的应用数量也体现了密码管理系统的大展拳脚之余还有巨大的应用前景。

为了更好地实现安全隐私密码管理系统的用户集成和应用拓展，我们可以**将可信第三方设计为浏览器插件的形式**完成对用户以及浏览器的交互。前台交互时，用户只需要从商城中下载安装浏览器插件，即可实现快速部署和自主管理自己的密码。而对于应用端而言，我们将安全隐私密码管理系统设计为接口的形式，其只需要调用对应接口即可获取相应密码，降低了使用成本和管理复杂度。

基于ABY3-VSSS的安全隐私密码管理系统在互联网安全与数据隐私保护领域具有广泛的应用前景。由于互联网信息安全威胁不断增加，越来越多的用户开始重视个人隐私信息的保护。因此，密码管理系统将得到更广泛的应用和推广。在电子商务、在线支付、社交网络等互联网应用领域，密码管理系统也扮演着重要的角色。近年来，随着国内电子商务、金融服务和社交网络等领域的高速发展，人们对密码安全的需求不断增加。在这些领域中，通过使用基于ABY3-VSSS安全隐私密码管理系统，可以有效保护用户的重要账号和财产安全，同时也加强了用户对隐私信息的掌控感和信任感。

此外，随着智能手机和智能家居等智能设备的广泛应用，密码管理系统的应用场景也将逐渐扩展。越来越多的智能设备和应用程序将以密码作为身份验证的一种标准方式，保障用户数据的安全性也将越来越得到重视。

总之，基于ABY3-VSSS的安全隐私密码管理系统的应用前景非常广泛，未来将得到更广泛的应用和推广。随着互联网信息化程度不断加深，用户对隐私信息安全的需求将会愈加强烈，密码管理系统将在此背景下发挥越来越重要的作用。

# 第二章 作品设计与实现

## 2.1 作品整体设计

考虑下面的场景，用户A在某网站注册了一个账户，在注册的过程中，用户A设置了账户名和密码，在注册完成后，用户A使用的浏览器会弹出提示框，询问A是否保存用户名和密码。追求便利的A让浏览器将密码存储起来，从而在下一次登录时，让浏览器自动填写网站的用户名和密码，通过浏览器的帮助，用户A加快登录速度、降低记忆成本。

但让浏览器帮助保存用户密码并不是一个可靠的密码管理方式，一方面浏览器存储密码的云服务器存在被攻击而导致密码泄露的风险，另一方面浏览器可能会出于某些目的主动获取用户存储的隐私密码。

除了浏览器，网站服务器也容易受到攻击。一旦攻击者获取了存储在网站服务器中的密码文件，如果密码是以明文形式存储的，那么他们就可以直接获取用户的隐私。即使密码是以加盐哈希的形式存储，攻击者仍然可以通过离线字典攻击来尝试破解其中的密码。

出于对上述传统密码管理系统所存在的安全性问题进行考虑，本作品的总体设计方法如图2-1-1所示：

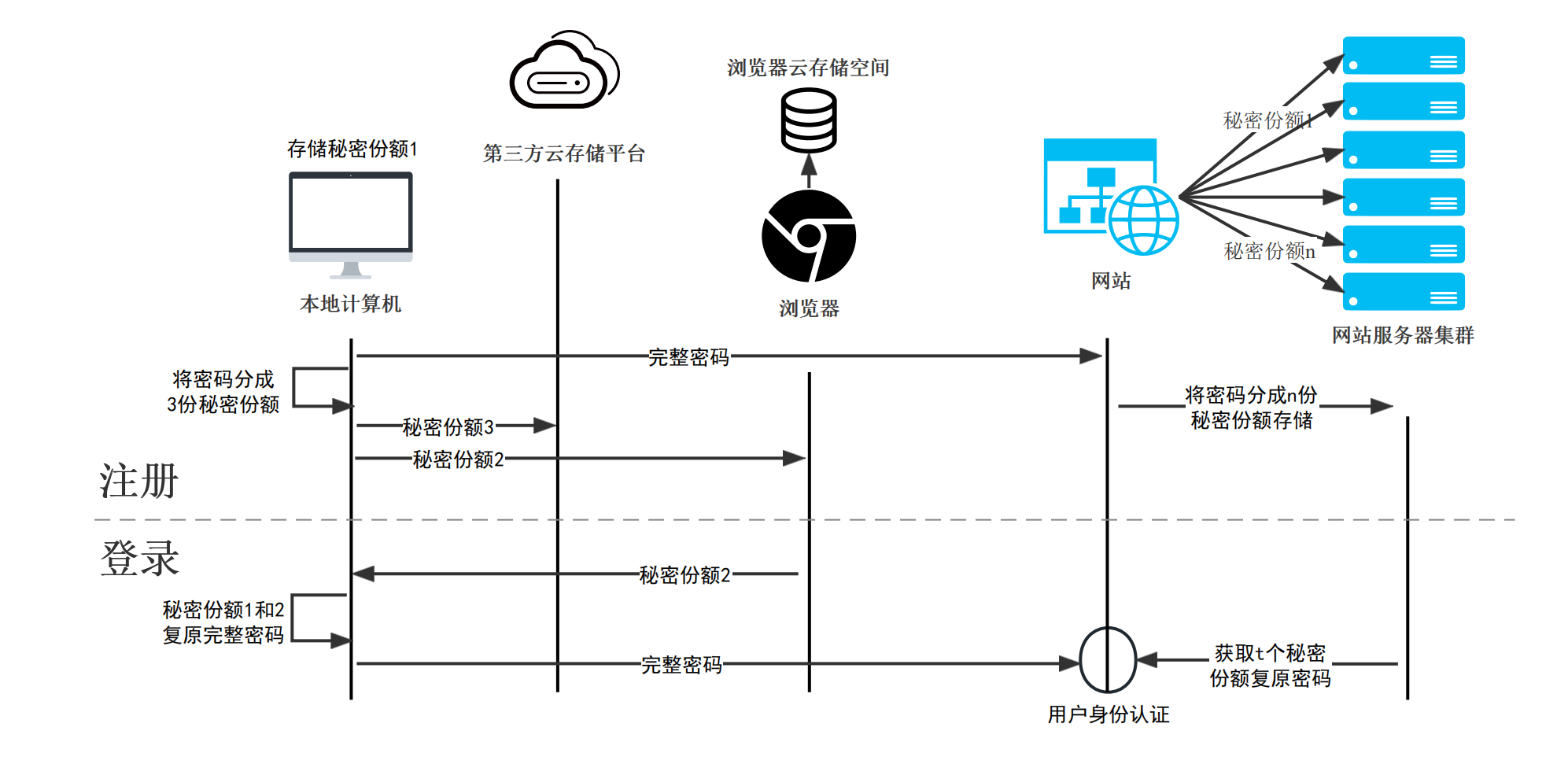


图2-1-1：系统总体设计图

**在用户-浏览器阶段，采用ABY3秘密共享协议。**将密码分成三份秘密份额，分别存储在用户本机、浏览器和一个半诚实（诚实但好奇）的第三方中，只有获取其中的至少两份秘密份额，用户才可以复原出登录密码，并将密码传输给Web服务器。其中半诚实第三方可以是我们的密码管理系统第三方，也可以是属于用户的其它物理设备。在我们的设计之中阐述的第三方为我们的密码管理系统。

如果用户对我们的系统云端作为半诚实第三方存储感到怀疑或不信任，我们也**提供了基于Cheetah协议的半诚实两方模型**。Cheetah协议会将用户个人隐私密码仅仅划分为两个秘密份额，分别存储在用户本机与浏览器之中，完全保证用户掌控自己的密码隐私。

相比于之前使用浏览器存储完整的隐私密码，使用浏览器存储密码的秘密份额可以保证攻击者或者浏览器在获取这份秘密份额后，不具有在信息论层面上复原出原始密码的可行性。剩余的两份秘密份额分别存储在用户的本地计算机和半诚实的第三方中，攻击者获取其中任何一方的秘密份额都无法复原出原始密码。如果本地计算机存储的秘密份额因为某些原因丢失，用户可以尝试从第三方获取存储的秘密份额，复原出原始密码；然后重新进行秘密份额的分配，并将秘密份额重新存储到本机计算机、浏览器和半诚实的第三方中。

**在网站服务器端，本作品使用可变的秘密共享模式（Variable Secret Sharing Schemas-VSSS）对网站服务器端存储的密码进行处理。**即网站将每条密码分成n份秘密份额，每份秘密份额存储在网站的一个服务器中，只有获取至少t份秘密份额后，网站服务器才能计算出用户的原始秘密进行认证。而且这个秘密共享方案具有可变性，目前常见的N方秘密共享方案有：Shamir方案、中国剩余定理（CRT）、Brickell和Additive Secret Share等，它们的秘密份额分配方式不同，对于不同的网站服务器，由于其不同的部署方式和计算性能，可以采用不同的秘密共享方案，以获取最佳的应用效果。当然我们也提供了安全性更强的Semi2k-SPDZ安全多方计算协议，该协议能够在协议运行过程中检测到恶意方并停止运行，以满足网站对更高安全性的要求。

采用秘密共享的方案，攻击者至少需要攻击t台服务器才有破解用户的原始密码的可能性，而且由于网站采用的秘密共享方式的不同，恢复的方法也不相同，这也极大程度上增加了攻击者恢复原密码的难度。

即使采用秘密共享方案存储密码信息，网站服务器仍然存在被攻击的风险。一旦服务器受到攻击，攻击者可能会发布或篡改所截获的秘密份额。为了解决这个问题，我们要求每台服务器在存储每条秘密份额时连接上一个服务器编号的哈希值，以便在数据泄露时进行数据溯源，确认是哪台服务器泄露数据。

如果某台服务器被攻击者劫持，攻击者可能会篡改存储的秘密份额信息，导致网站使用错误的秘密份额进行复原，进而导致复原失败或复原出错误密码的情况。本作品提出了一种优化后的组合方法，用于帮助网站在用户提出报警后确认是否有服务器被绑架，并快速找出被绑架的服务器。

使用本作品的密码管理系统，在注册过程中，用户A首先将完整密码传给网站，网站采用(t,n)的可变秘密共享方案将密码分成n份秘密份额，存储在n个不同的服务器中；然后用户A在本地计算机采用ABY3协议将完整密码分成三份秘密份额，将秘密份额①存储在本地计算机、秘密份额②存储在浏览器中，秘密份额③存储在第三方云存储平台。用户需要登录时，浏览器会将秘密份额②发送到本机计算机，和本地计算机中的秘密份额①一起复原出原始密码，并将原始密码发送给网站；网站从t个存储秘密份额的服务器获取信息并复原出网站端存储的用户密码，将用户发送来的密码和网站存储的密码进行对比，认证用户身份。

本作品从用户和浏览器的交互和网站服务器存储两个阶段利用秘密共享的方案，给出了一种更加安全、鲁棒性和扩展性都更加强大的密码管理系统。

## 2.2 系统详细模块设计

### 2.2.1 协议参与方介绍

在本节中，我们将重点介绍系统涉及的各个参与方。由于我们的协议可以分为两个阶段：用户-浏览器交互阶段以及网站服务器交互阶段，因此我们将按照协议的两个阶段进行分阶段介绍。

#### 2.2.1.1 用户-浏览器阶段

在用户-浏览器阶段，系统应用的主要场景是用户使用浏览器中的自动存储密码与自动登录功能，然而用户与浏览器之间的不对等地位引发了我们对于个人数据安全的担忧。作为用户，我们必须承认我们无法掌控浏览器方的操作，并且我们无法确切知悉他们对我们的个人信息和密码的处理方式，这样的不确定性带来了对于数据隐私和安全的不确定性。

面对我们并不完全信任的浏览器，我们采用了三方模型来构建我们的系统设计。我们的系统设计和模型创建基于半诚实条件，在这种条件下引入了半诚实第三方。我们能够证明在半诚实条件下，我们的设计对用户是安全的，第三方和浏览器无法在未经用户同意的情况下获取用户的密码信息。通过我们的系统设计，弥补了用户与浏览器之间的不对等关系，消除用户对于浏览器安全性的依赖，为用户提供一个可信赖的密码管理解决方案。

我们运用ABY3协议，将密码分成三份秘密份额，分别存储在用户本机、浏览器和可信第三方中，只有获取其中的至少两份秘密份额，才可以复原出登录密码，实现了第三方与浏览器在没有用户方的秘密份额的前提条件下，无法获得用户密码信息。在用户与浏览器阶段，我们的协议参与方基本上包括：用户，浏览器，第三方（推荐，可选）

图2-2-1-1为该阶段的建模设计原型：

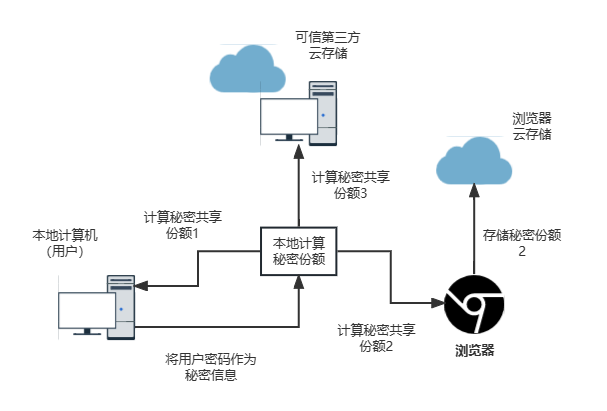


图2-2-1-1：用户--浏览器阶段设计原型

当然，我们理解用户可能无法完全信任半诚实第三方，担忧第三方可能会和浏览器联合窃取用户的秘密信息。因此，我们还提供了基于Cheetah两方安全计算协议的秘密共享方案。在这种情况下，我们仅在本地物理设备和浏览器中分别存储一份秘密份额。然而这样做会牺牲一定的鲁棒性和密码恢复功能，但同时也提高了该阶段的安全性，从而让用户对系统充分信任。

通过采用这种两方安全计算协议的秘密共享方案，用户的密码信息仅存在于用户本地和其所使用的浏览器中，减少了第三方参与的风险和可能的数据泄露渠道，确保了用户数据的隐私和安全。**如果用户希望在多台物理设备上都能享受我们的密码管理保护，我们仍建议用户选择三方模式运行，三方模式允许在缺少本地计算机秘密份额的情况下，从半诚实第三方获取秘密份额，并和浏览器中存储的秘密份额复原出原始密码。**在三方模式下，非本地计算机的物理设备仍使用我们的密码管理系统，保证密码的安全性。

#### 2.2.1.2 服务器登录阶段

在日常生活之中，用户在登录网站时需要在浏览器中输入网站的URL或通过搜索引擎进入目标网站。在登录页面，用户通常需要提供身份凭据，包括用户名、电子邮件或手机号码以及相应的密码，这些凭据用于验证用户的身份。根据身份验证的结果，网站服务器会向浏览器发送相应的认证结果。如果用户提供的凭据是有效且匹配的，服务器会发出成功认证的信号。一旦用户成功认证，网站会授予用户相应的访问权限。在成功登录后，用户通常会被重定向到登录后的页面，这可能是用户的个人资料页面、主页或其他指定的目标页面。

图2-2-1-2展示的为现实生活中服务器登录流程：

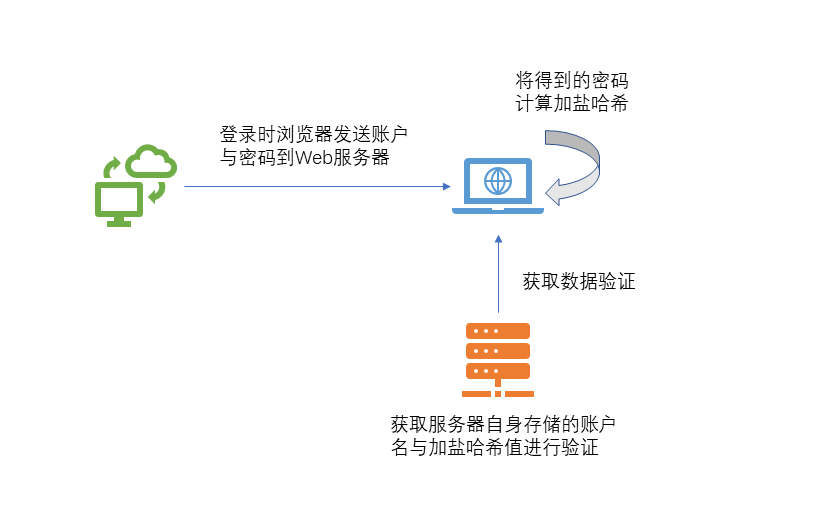


图2-2-1-2：服务器登录流程建模

在服务器登录流程中，涉及到用户、浏览器和网站服务器的参与。我们通过采用可变的秘密共享方案，在服务器端进行了创新性的设计，将用户密码存储为秘密共享的形式，而不是使用传统方法，例如：明文，哈希和加盐哈希等方法。

我们在该阶段的设计建模如图所示：

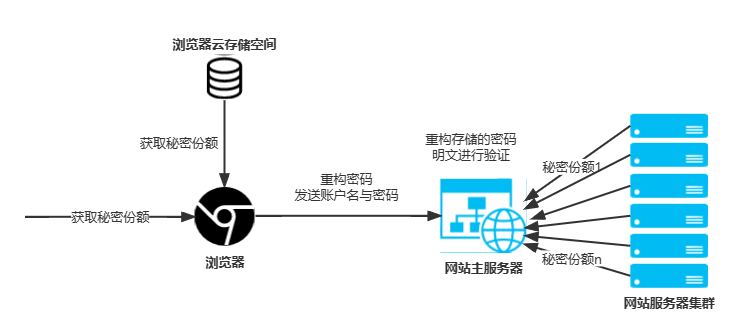


图2-2-1-3：服务器登录阶段设计原型

通过这种方式，我们极大地增强了服务器端秘密信息存储的鲁棒性，并且能够抵御目前绝大多数针对加密算法的攻击方式。这是因为秘密共享方案是基于信息论安全的原理，只有在获得足够的秘密份额之后，才能进行有效的攻击。在没有足够的门限秘密份额的情况下，攻击是无法成功的。

### 2.2.2 注册密码秘密共享模块

在我们的系统设计中，全面考虑了用户从注册到登录的整体流程。首先，该场景下用户的第一步是账户密码的注册过程，因此我们系统设计的第一个模块是注册密码模块。在这个模块中，我们实现了密码的秘密共享分布式存储，从而使完整密码对仅仅存储一份秘密份额的浏览器和半诚实第三方都是不可见的。这种安全设计保障了用户的密码隐私。

在注册密码秘密共享模块中，我们提供了两种模式：三方模式（用户、浏览器、半诚实第三方）和两方模式（用户、浏览器）。我们推荐用户选择三方模式，因为它提供了密码的可恢复性。然而，我们也理解一些用户可能对半诚实第三方存在疑虑，因此我们为这些用户提供了两方模式作为备选方案。无论选择哪种模式，我们都致力于保护用户的密码安全和隐私。

#### 2.2.2.1 三方模式

在本节中，我们将重点介绍基于ABY3协议的三方模式在注册密码秘密共享模块中的应用。在三方模式下，用户完成账户注册和密码设置后，在使用浏览器的自动密码存储功能之前，我们的系统将对用户注册的密码进行秘密份额划分。在三方模式中，密码将被划分为三个份额，并用秘密份额①、秘密份额②和秘密份额③来表示（为了方便叙述，我们将简称这些份额为份额①、份额②和份额③）。

在三方模式下，我们会将份额①存储到用户本机的特定文件夹之中，接着我们会将份额②作为交付给浏览器自动存储的“密码”，浏览器会将份额②存储到自己的云端之中，最后我们会将份额③存储到半诚实第三方的云端之中。我们的协议会在用户的本机以一种类似于浏览器插件的形式运行（可扩展为浏览器插件），即我们的三份秘密份额都是在用户本机运算得到的，那么在我们的假设之中，作为半诚实的第三方仅仅能获得秘密份额的一部分；而不像当前浏览器常常做的那样，直接存储密码的AES加密密文形式（显然AES密钥也是掌握在浏览器一方的手中的）。

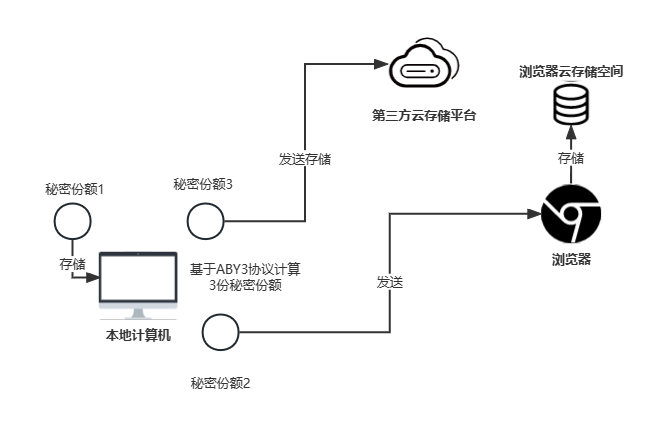


图2-2-2-1：三方模型

在这种情况下，作为半诚实的第三方参与方，如果想要恢复密钥，必须获取另一方的秘密份额。在这里，我们采用了ABY3协议并以(2, 3)门限方式应用，这意味着我们最多能够抵御一个恶意方的参与。由于我们的系统作为一个中间层进行处理，浏览器与我们的半诚实第三方串通的可能性很小，因此高效的ABY3协议已经具有很强的安全性。

**安全协议通常会带来计算时间和性能方面的损失。根据我们的调研，用户在注册账户密码时通常有足够的耐心，可以接受性能上的损失。**

#### 2.2.2.2 两方模式

在三方模式下，仍然存在半诚实第三方与浏览器之间串通的风险。如果用户对此持怀疑态度，我们的系统还提供了基于Cheetah安全协议的两方模式。在这种模式下，完全取消了半诚实第三方的参与，进一步增强了用户的安全保障。

在两方模式下，用户注册密码后，我们的系统会将密码划分为两个份额，并用秘密份额①与秘密份额②来表示（为了方便叙述，我们将简称这些份额为份额①和份额②）。

在两方模式下，我们将份额①存储在用户本机的特定文件夹中，并将份额②提交给浏览器自动存储。这样，我们确保真实密码对浏览器是不可见的，并且没有其他参与方介入。用户完全掌握主动权，且使用Cheetah安全协议带来的性能相对较高，提供了更好的用户体验。

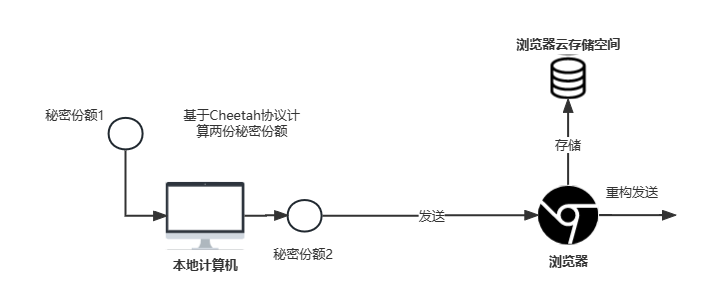


图2-2-2-2：两方模型

然而，相对于三方协议，两方协议在可扩展性和鲁棒性方面存在一些缺陷。在三方协议中，即使用户本机或浏览器云端遭受攻击者攻击，由于半诚实第三方存储的秘密份额，我们仍能够成功恢复并登录到网站服务器，这种特性为我们的系统在管理用户密码方面提供了可靠性。

我们还需要考虑到一个实际场景：如果用户不使用同一台物理设备进行登录应该如何处理，在这种情况下，用户无法获取存储在本地计算机中的秘密份额。对于这种场景，我们的两方模型是无法恢复出完整密码的，但是由于我们将系统分成了两个阶段，在其它物理设备上用户可以直接输入完整密码来登录网站服务器。但对于三方模型而言，如果用户使用了其它的物理设备进行登录，那么我们可以通过半诚实第三方的份额与浏览器的份额恢复密码成功登录。在其他物理设备上，使用三方模型恢复了密码原文信息后，用户还可以重新进行秘密份额的划分，并将一份秘密份额存储在当前物理设备，并将其他两份秘密份额更新到浏览器和半诚实的第三方中去。

图2-2-2-3展示为跨物理设备的登录流程：

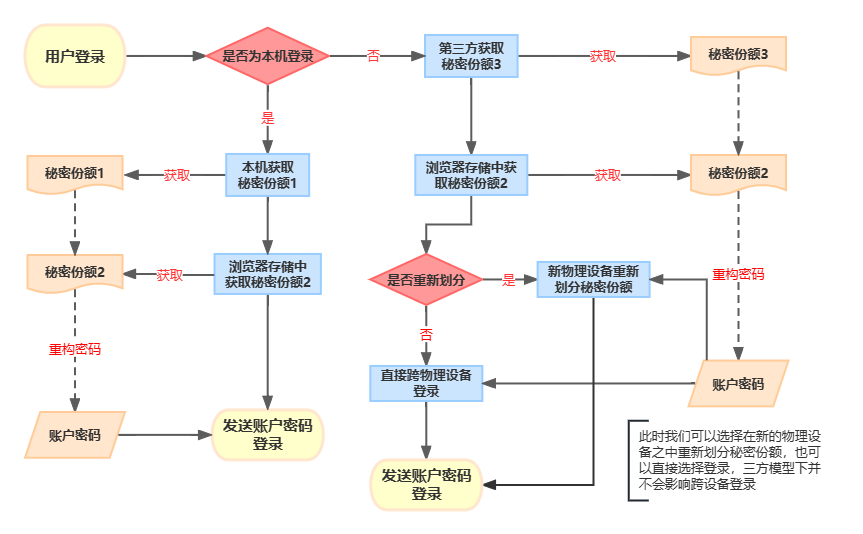


图2-2-2-3：跨物理设备登录与是否重新划分

至此，经过精心设计与实现，我们的注册密码秘密共享模块已经成功完成，为用户提供了安全可靠的密码管理方案。在这个模块中，我们融合了目前最先进的密码学技术和秘密共享方案，确保用户的密码在存储和传输过程中得到最大程度的保护。通过秘密份额的划分和合理的存储策略，我们实现了密码的分散存储，使得即使在面对恶意攻击或者系统故障的情况下，用户的密码仍然能够得到有效的保护和恢复。通过这个模块，我们为用户带来了更高层次的安全性和便利性，确保他们账户秘密的安全。

### 2.2.3 秘密恢复登录模块

在用户成功注册账户和密码后，下一步是使用浏览器作为媒介登录Web网站。此时，我们的密码已经不存在任何明文形式的存储，因此在登录过程中需要对密码这个秘密信息进行重构，然后通过浏览器完成网站登录操作。

在我们的注册密码秘密共享阶段，如果采用三方模型，我们会将密码分布式存储为三份秘密份额。在秘密恢复登录模块中，我们会向用户提出请求，并在获得用户的同意后，提取保存在用户本机的秘密份额。然后，我们将这个本机份额与浏览器方的秘密份额进行组合和重建，从而获得我们登录时所需的密码明文。

**值得注意的是，我们选择了在本机进行密码恢复和重构的策略，而不是在Web网站服务器端进行操作。这种分布式计算的方法可以减轻Web服务器的负担，并提高整体系统的效率和可靠性。**

通过将密码恢复的计算任务分配到用户本机和浏览器之间，我们实现了一种安全且高效的密码重建方式。用户的本机参与了密码恢复的过程，确保了密码的安全性和隐私性。同时，这也有助于提升整体系统的性能和用户体验。

下图展示的是三方模型中通过任意两方的组合进行密码的恢复与登录的描述，具体分为三种可能的排列组合方式，可以灵活运用应对不同场景。

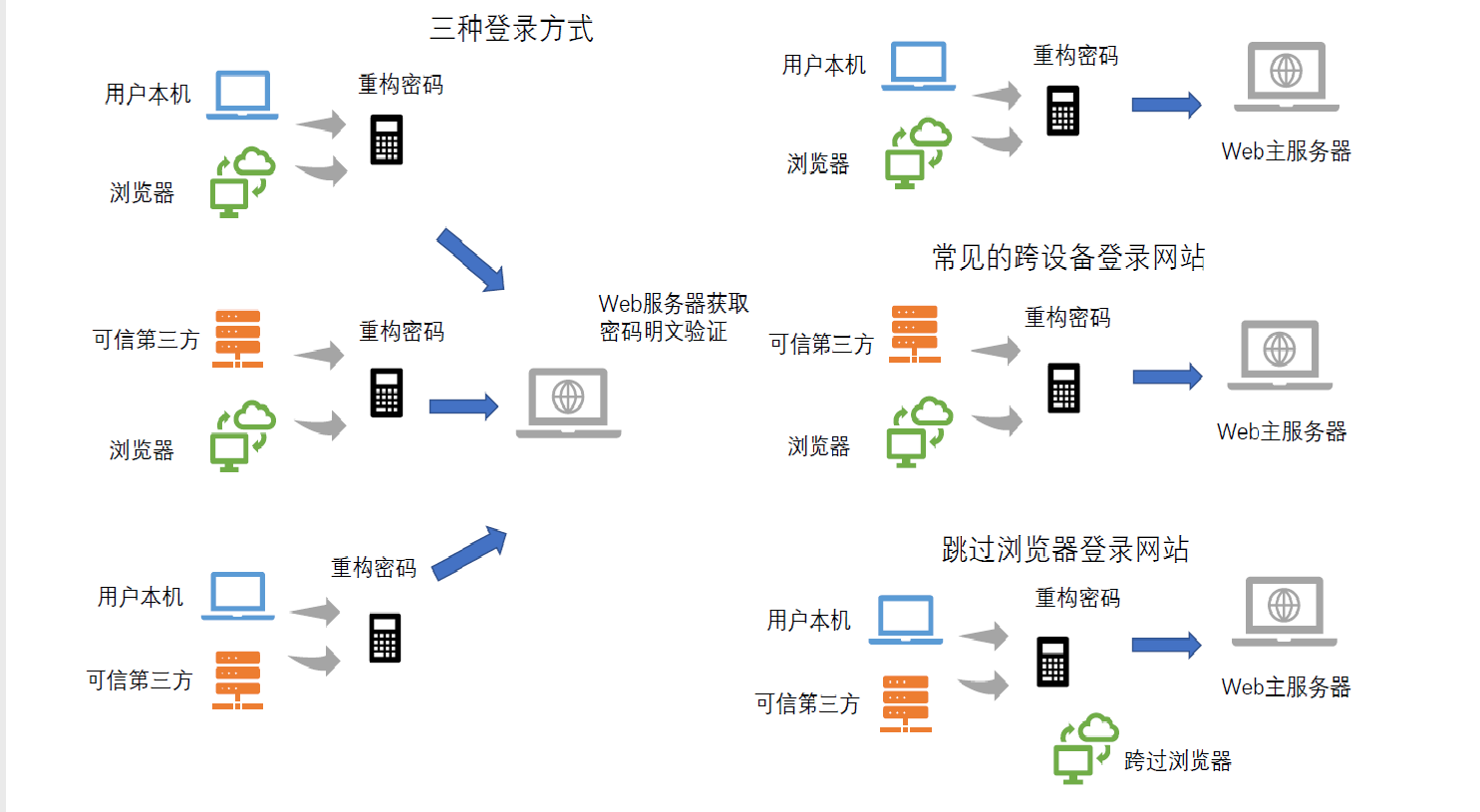


图2-2-3-1：任意两方的组合进行密码恢复

在我们的三方模型的秘密共享方案下，任意两方都能够实现密码的恢复，这为用户带来了极大的灵活性与可扩展性。**在没有秘密份额丢失的情况下，存在三种不同的恢复密码组合形式：**

1. 基本的组合模式为（用户，浏览器），该组合模式用于完成日常的网站登录。在这种组合模式下，半诚实第三方存储的秘密份额可以近似作为密码的备份，除了用户自身，其他两方无法获得我们的隐私密码信息。
2. 当我们采用（第三方，浏览器）这种组合模式时，我们能够实现跨物理设备的网站登录。例如，当我们在另一台不存储秘密份额的设备上进行登录时，我们可以通过该组合模式来完成网站的登录。我们还可以采取秘密份额重新划分的方式，在新设备上存储秘密份额。
3. 当我们采用（用户，第三方）的组合模式时，我们能够应对恶意浏览器的情况。半诚实第三方可以代替部分浏览器的功能，完成网站的登录过程。

这些不同的组合模式为用户提供了可选择性，以适应不同的登录场景和需求。无论是日常登录、跨设备登录还是应对恶意浏览器，我们的系统都提供了相应的组合模式来确保安全性和便利性。

在我们的两方模型的秘密共享下，我们认为系统具有更强的安全性，因为用户完全掌握了主动权。在这个阶段，如果没有同时获得用户和浏览器存储的秘密份额，就无法进行任何试图登录的操作；然而这种情况下我们无法处理本地设备的单点故障，并且当用户在其他物理设备上登录时，无法继续使用我们的系统进行密码管理。

在两方模型下，用户可以更加安全地管理自己的密码，因为只有在拥有自己的秘密份额时才能进行登录。这种方式确保了用户的隐私和安全，减少了半诚实第三方参与后可能存在的安全风险；但用户也需要保管好自己的秘密份额，并在使用其他设备时寻找其他适当的密码管理方法。

### 2.2.4 Web分布式服务器验证模块

在前面几个模块的设计和实现中，我们已经完成了用户与浏览器之间的整体流程。我们成功地实现了一种使用户密码对浏览器不可见的方法，并让用户完全掌握了密码管理的主动权。现在我们需要考虑的是整个系统设计的第二个阶段——Web服务器验证用户登录阶段。在这个阶段，我们将关注用户在登录时与Web服务器之间的交互和验证过程。

当用户访问需要身份验证的Web应用程序或网站时，他们会被重定向到登录页面。用户在登录页面上输入他们的用户名和密码，并将其提交给Web服务器。Web服务器接收到用户提交的凭据后，会对其进行验证。通常，服务器会将用户提供的密码使用哈希、加盐哈希的形式存储在服务器上。当获得用户提交的密码后，服务器会获取用户提供的密码，并进行相同的哈希计算。然后，它将计算得到的哈希值与存储的哈希值进行比对，如果相同则登录成功。如果凭据验证成功，服务器会创建一个会话（session），并为用户分配一个唯一的会话标识符。该会话标识符通常以cookie的形式返回给用户的浏览器，并用于后续的身份验证和访问控制。

我们在设计Web服务器验证模块时关注于创建一种新的模式，旨在使Web服务器能够更安全、更可靠地验证用户的凭证，同时保持对session和cookie功能的兼容性。我们的设计目标是提升系统的安全性，而不会对现有的session和cookie机制产生任何负面影响。

对于现代的Web网站而言，采用多台服务器来提供支持是很常见的做法。在我们的设计中，与目前流行的加盐哈希方法不同，我们采用了一种全新的方式来处理用户的密码。在用户注册账户并设置密码后，我们不会将密码与盐值组合进行多轮迭代哈希以生成固定长度的哈希值，并将该哈希值存储在单台服务器上。相反，我们的设计采用了(t, n)阈值秘密共享方案或NPC安全多方计算协议，将用户的密码原文作为秘密信息进行秘密份额的分割，并分布式地存储在Web网站的n台服务器上。这样的设计实现了基于信息论的安全性，并提供了良好的可靠性。

通过这种方式，即使某台服务器遭到攻击或数据泄露，攻击者也无法获得完整的密码信息。只有当足够数量的服务器参与计算，并共享其秘密份额，才能还原出用户的密码原文。这种分布式存储和计算的方式提供了更高的安全性和可靠性，即使少数几台服务器被劫持，导致了秘密份额的丢失，也能够确保用户密码的完整性和可恢复性，继续支持成功登录网站。

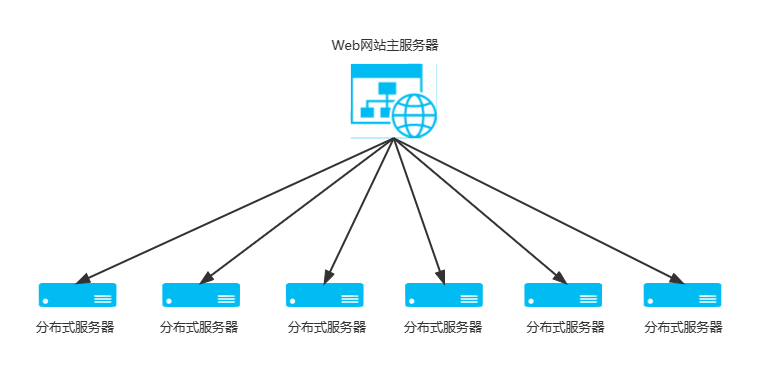


图2-2-4-1：Web服务器分布式集群架构

**在这种分布式存储的架构中，我们将用户密码信息的秘密份额分别存储在n台服务器上。在少于t台服务器参与恢复密码的情况下，无法获取用户真正的密码。**对于攻击者来说，要攻破用户密码，他们必须在非常短的时间内获取至少t台服务器的权限，因为我们设计了动态调整和自检机制，会动态检查每台服务器是否被劫持。

这个过程显然非常困难。在现代的服务器保护技术面前，想要实现针对一台服务器的漏洞发现->漏洞利用->提权->劫持->内网渗透已经十分困难。在短时间内劫持多台服务器几乎是不可能的，而且攻击者还必须应对入侵检测系统带来的麻烦。我们的系统架构在安全性方面考虑周全，有效地减少了攻击者成功攻击的风险，并在Web服务器端为用户的密码保护提供了高度的可靠性和保密性。

#### 2.2.4.1 性能模式

**为了满足不同条件和不同服务器场景的需求，我们在Web分布式服务器验证模块中提供了两种模式：性能模式和安全模式。**本节主要介绍服务器端的性能模式。

**使用性能模式需要满足一个前提假设：Web网站的大多数服务器都是诚实可信的，并且具有较高的安全防护级别。**因为在我们的性能模式中，仅支持简单的纠错和溯源机制（将在后续小节中介绍），无法应对大规模不可信方的出现。

我们的性能模式使用较简单的Shamir、Brickell秘密共享方案，分别基于一维方程和多维向量方程（也集成了Additive Secret Share秘密共享方案）。由于Shamir和Brickell秘密共享方案相对简单，所以其方案性能较好，秘密份额的划分和重构相对简单，不需要其他公钥认证等操作。相对而言，它缺乏许多安全机制，例如加密和认证，在面临恶劣环境时，无论是恶意参与者的攻击还是秘密份额的劫持和破解，都无法提供足够的防护。

在选择性能模式时，需要在性能和安全之间进行权衡，并确保服务器环境相对安全可信。对于需要更高安全级别的场景，我们建议选择安全模式，该模式提供了更多的安全机制和保护措施。

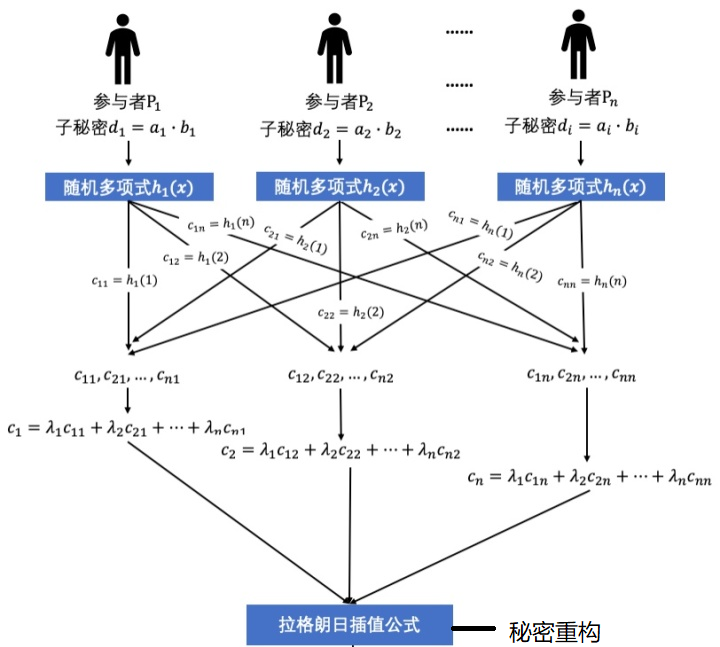


图2-2-4-2：Shamir方案原理

在实际场景中，可以合理假设Web网站的多台服务器大部分都是可信诚实的，这更符合现实情况，如果加入了更多类似于公钥认证等安全操作，会带来很大的性能损失，这样是得不偿失的。对于少数恶意参与方或被劫持的服务器，我们设计的纠错和溯源机制已经能够满足现实生活的需求。为了适应那些对安全性要求极高的场景，我们还提供了安全模式供选择。这样，网站可以根据具体需求和风险评估，选择性能模式或安全模式，以确保系统在各种环境下都能提供合适的保护措施。

#### 2.2.4.2 安全模式

**为了满足Web服务器端对更高安全性的要求，我们还提供了基于Semi2k-SPDZ协议的用户密码共享方案。**SPDZ协议在预处理阶段使用部分同态加密来计算消息授权验证码（MACs）和乘法三元组的分片数据，利用乘法三元组将乘法计算转化为线性组合，并通过MAC进行验证，以确保不受恶意敌方的干扰。如果协议被违反，会检测恶意攻击并停止协议的运行，以防止数据泄漏。Semi2k-SPDZ协议类似于SPDZ协议，是NPC安全协议，但需要一个可信第三方来生成离线随机数。目前，Semi2k-SPDZ协议默认使用可信第一方来实现。通过引入这种秘密共享方案，我们能够提供更高级别的安全保护，以满足Web服务器对于安全性的更严格要求。

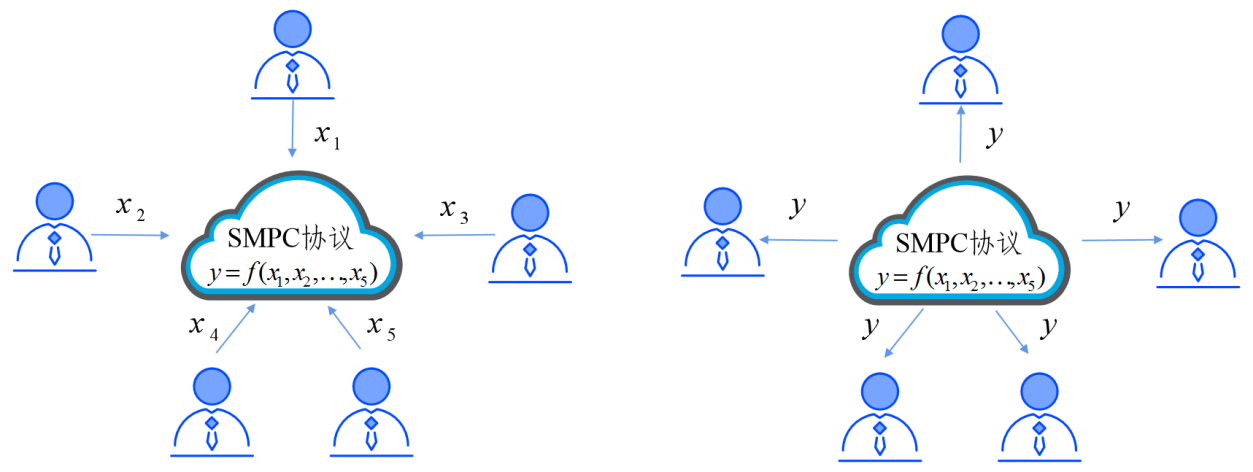


图2-2-4-3：Semi2k-SPDZ方案原理

Semi2k-SPDZ协议能够在协议运行过程中检测到恶意方并停止运行，以防止数据泄露。然而，要在协议中进行恶意方检测，需要使用我们优化后的组合方法，该方法适用于恶意方出现频率较低的情况。对于恶意方出现频率较高的情况，我们建议集成基于可检测恶意方的NPC协议，以进一步增强安全性。通过引入这些安全机制，我们能够有效应对潜在的恶意方，并提供更可靠的数据保护。我们目前也正在努力尝试搭建SPDZ这一MPC协议计算框架，集成部署部分经典协议。

### 2.2.5 服务器数据泄露溯源模块

为了提高系统的完整性并完善系统功能，我们进一步设计了数据泄露溯源模块。**该模块的主要目的是解决在发现分布式服务器中存储的秘密份额泄露时，如何确定是哪台服务器出现了问题导致数据泄露的情况。**通过这个模块，我们能够追溯数据泄露的源头，快速发现并处理故障服务器，从而保护用户的数据安全。

为了实现数据泄露溯源功能，我们采用了安全通信协议中的部分设计。在我们的设计中，服务器在计算用户密码的秘密份额时，会将每个秘密份额与对应存储服务器节点编号或节点哈希值进行连接，然后将它们一起发送给对应的服务器节点进行存储。这样做的目的是为了在数据泄露的情况下能够追溯到具体的服务器节点，从而确定哪个节点存在问题导致了数据泄露。具体形式如下：

**秘密份额 | 节点编号（节点哈希值）**

在这种存储方式下，处理过程是由我们系统内置的算法完成的，所以攻击者对于我们系统内部的秘密份额存储过程一无所知。即使攻击者获得了分布式存储的秘密份额，实际上他所获取的并不是真实的秘密份额信息，还需要进行一定的分割处理才能得到真正的秘密份额。对于较多的秘密份额来说，这并不是一件容易的事情。

由于我们在服务器端采用分布式存储，所以会涉及到秘密份额的分发过程。在我们的系统设计中，我们选择其中一台服务器作为主服务器。当我们注册密码后，首先会将密码的原文重定向发送到主服务器，主服务器会计算出 n 个秘密份额，并将这些秘密份额与额外的信息进行拼接，然后发送给其他 n-1 台服务器。在这种情况下，拼接节点编号或节点哈希值的算法仅部署在主服务器上，这符合 Semi2k-SPDZ 协议中一方作为可信方的要求。

由于拼接节点编号或节点哈希值的算法只存在于主服务器上，这意味着其他服务器也不存储真正的秘密份额，并且实际上他们并不知道自己存储的并非真正的秘密份额。这在受到劫持时能够给攻击者带来一定的混淆。这也意味着我们的主服务器非常重要，必须得到充分的安全防护。对主服务器的安全保护至关重要，以确保系统的完整性和安全性，当然我们的主服务器是可以切换的。

值得一提的是，我们在分发秘密份额时会从主服务器生成并分发 n 份秘密份额。而在恢复用户密码这一秘密信息时，仅需要使用 t 份秘密份额即可完成。为了选择这 t 份份额，我们采用了随机序列的方式来随机选择哪些服务器上的秘密份额。这样，我们的取秘密份额的模式是随机的，确保每次用户登录时所使用的恢复密码份额都是不同的。

### 2.2.6 服务器动态调整与自检模块

在实际生活中，服务器端存储秘密份额的服务器数量 n 是可能会动态调整的。Web网站的服务器往往需要根据需求进行扩增，因此我们在服务器端内置了Brickell秘密共享方案来应对这种情况。

Brickell方案是对Shamir方案的改进，它将一维方程扩展为多维向量方程，运用多项式插值方法。该方案在数据安全方面表现出较好的性能，能够在不泄露重建秘密的情况下，检查参与者的行为是否一致，并支持动态增减参与者，即使不知道最终参与者的个数。然而，由于其在加密和解密过程中的计算开销较大，相对于Shamir方案而言，效率会有所降低。

通过引入Brickell秘密共享方案，我们能够灵活应对服务器数量的动态变化，在保证数据安全性的同时提供更强的灵活性。这使得我们的系统能够适应不同规模和需求的Web网站，同时我们也能够使用Additive Secret Share等方案实现对秘密份额在其自身服务器上定时刷新，来达到秘密份额不是一成不变的效果。我们也能够选择定时收集所有秘密份额并重新划分，但是这会带来极大的性能开销。

除了集成了支持服务器端动态调整参与方数目的Brickell秘密共享方案，我们还设计了用于检查分布式服务器中恶意参与方的纠错自检模块。这个自检模块基于朴素的排列组合方法，并在此基础上进行了优化。

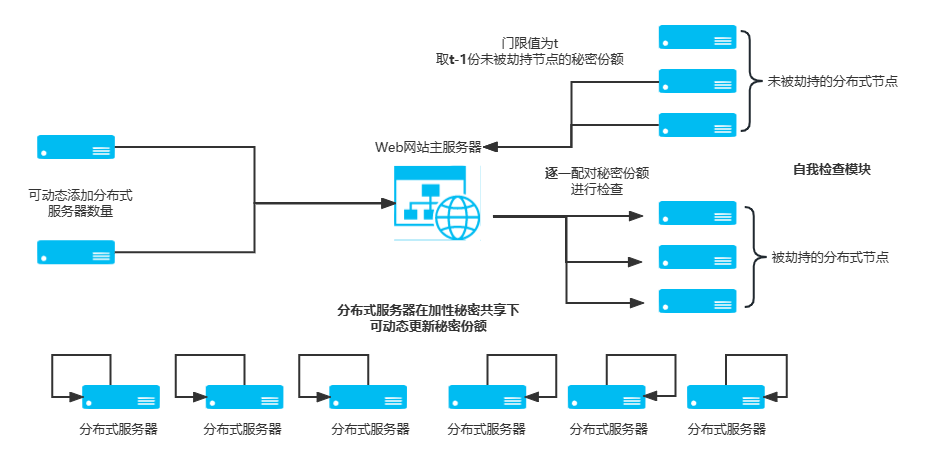


图2-2-6-1：动态调整与自我检查模块

我们不需要完成次排列组合的秘密恢复，我们只需要成功完成一次的秘密恢复即可，当系统完成了一次的秘密恢复后，从成功恢复的组合之中选取 t-1 个秘密份额，再将该 t-1个秘密份额分别与其余的 n-t个秘密份额逐一进行秘密恢复尝试，如果未能恢复成功，那么新加入的一方即为恶意参与方。

通过这种优化后的组合方案，我们在平均每次自检的时间性能上获得了不错的提升。然而，与传统的排列组合方式相比，即使是最坏情况下时间复杂度也没有太大差异。

当我们无法找到能够成功组合恢复的 t 个份额时，我们会向服务器发送预警并暂停所有功能服务，使秘密份额不再可用。在这段时间内，分布式服务器将自行检查自身情况，待服务器端发出重新启动密码管理系统的命令后，系统将恢复正常运行。这种预警和暂停措施加强了系统的安全性和可靠性。

## 2.3 安全协议方案介绍

### 2.3.1 ABY3与Cheetah

在用户-浏览器阶段的系统设计中，我们实现了基于ABY3协议的三方模型。这种模型可以确保数据的隐私和安全性，但如果用户对我半诚实第三方缺乏信任，我们也提供了基于Cheetah协议的两方模型。通过采用Cheetah协议，用户完全掌握密码管理的主动权，因为可以信任自己的设备。接下来，我们将详细介绍ABY3和Cheetah协议的主要内容、工作原理和特点。

#### 2.3.1.1 ABY3方案[7]

本小节将对ABY3（Arithmetic-Binary-Yao; Three-party with one malicious adversaries）进行简要介绍，以便理解设计细节和安全性分析。

安全多方计算中的算术共享（Arithmetic Sharing）、布尔共享（Boolean Sharing）、和姚氏混淆电路共享（Yao's Sharing）对于计算不同的算子，例如加法、乘法、比较等各有优势。如何设计更高效的算法、如何平衡不同技术之间的优势侧重从而发挥最好的性能，是一个值得研究的重要问题。

2015年，Demmler等人在论文《ABY–A Framework for Efficient Mixed-Protocol Secure Two-Party Computation》[6]中首次提出ABY协议，引入了算术共享、布尔共享和姚氏共享之间的有效转换，该框架仅实现具有半诚实安全性的两方计算。

ABY3协议是在ABY的基础上，由Mohasseld等人进行改进的协议，能够针对半诚实敌手和恶意敌手模型的安全三方隐私计算框架，协议框架中三个服务器是彼此对等的。在安全性上，每个计算参与方对于某个秘密都遵循一个(2,3)的门限方案：即3为参与方总数量，2为可重构秘密参与方的最少数量，ABY3协议能够抵抗最多一个恶意参与方而保证安全性，即在恶意行为发生的时候能够中止协议，在安全与效率之间取得了较好的平衡。

ABY3协议在具体算法实现上，综合采用算术协议（Arithmetic）、二进制协议（Binary）以及混淆电路协议（Yao，此处以姚期智院士的姓氏首字母代指其首创的混淆电路方法），在协议切换上有很多创新。对二进制协议和混淆电路协议的支持使得ABY3能够直接计算非线性函数，同时其独特的乘法算法采用截尾方法（truncation）而不需要依赖Beaver三元组。

#### 2.3.1.2 Cheetah方案[8]

本小节将对Cheetah协议进行简要介绍，以便理解设计细节和安全性分析。Cheetah协议来自于USENIX Security'2022的一篇文章《Cheetah: Lean and Fast Secure Two-Party Deep Neural Network Inference》, 作者是来自阿里安全双子座实验室的洪澄博士团队。

2PC-NN安全推理与实际应用之间仍存在较大性能差距，Cheetah协议通过对DNN, 基于格的同态加密、VOLE类型的不经意传输和秘密共享进行详细设计提出了一个2PC-NN模型，Cheetah协议相对于其它的方案开销小的多, 计算效率更快, 通信效率更高，目前是较为适合现实生活中应用的快速2PC半诚实协议。

Cheetah协议的创新：

1. 在协议中涉及的同态rotation操作是基于格的HE方案的性能瓶颈之一, 在Cheetah协议中通过构造映射巧妙地消除了同态rotation运算, 也并没有使用SIMD, 加快了同态运算的效率。
2. 基于HE的协议可以直接接受的秘密份额, 而不局限于, 避免了额外的计算和通信开销。
3. 在Cheetah协议中使用了基于VOLE类型的OT扩展协议来构造高效、精简的非线性计算协议, 如截断、比较协议等, 极大降低了安全推理的计算和通信开销。

### 2.3.2 可变的秘密共享模式

在服务器登录阶段，我们采用可变的秘密共享模式（VSSS）。对于VSSS模式，根据服务器不同的部署环境和需求，分为性能模式和安全模式。在性能模式中，提供了Shamir、中国剩余定理、Blakley、Brickell四种秘密共享方案；在安全模式中，提供了Semi2k-SPDZ这种秘密共享方案，下面一一介绍这些秘密共享方法的原理。

#### 2.3.2.1 Shamir方案[9]

Adi Shamir于1979年提出了基于拉格朗日(Lagrange)插值定理的(t,n)门限方案。该方案利用有限域上的n次随机多项式来分享秘密，被分享的秘密为多项式的零次系数，恢复秘密至少需要t个多项式上的点。

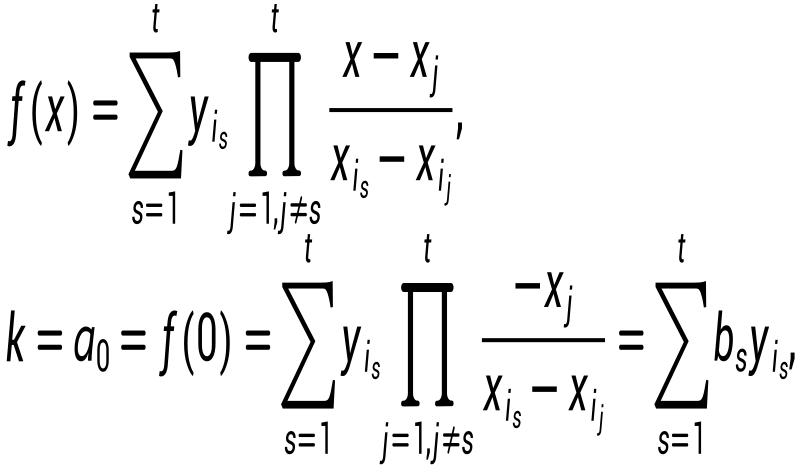
方案描述如下：设GF(q)是一个有限域,q为公开大素数,共享的密钥k∈GF(q),可信中心给n(n<q)个共享者Pi(1≤i≤n)分配共享的过程如下:

(1)秘密分发

可信中心随机选取多项式f(x)=at−1xt−1+…+a2x2+a1x+a0∈GF(q)[x],常数a0=k为要分享的秘密。可信中心在GF(q)中选择n个非零的互不相同的元素x1,x2,…,xn,计算yi=f(xi),1≤i≤n,将子密钥(xi,yi)分配给共享者Pi(xi是公开的,yi为Pi的秘密共享)。

（2） 秘密重构

给定t个共享(1≤s≤t),从Lagrange多项式重构的



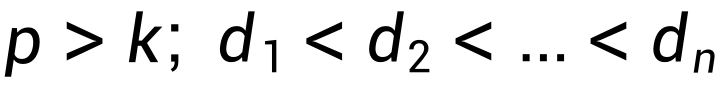
其中,(Lagrange插值系数),运算都是GF(q)上的运算。

以上是Shamir方案的基本实现原理，Shamir方案认为秘密的分发者和参与者都是诚实的。然而在具体实践中，参与者可能会故意或是由于一些非主观因素（如网络传输错误）提供了错误的份额, 导致无法正确地恢复秘密。基于此，在文献[10,11,12,13]中提出了可验证秘密共享方案，这些方案基于离散对数、椭圆曲线加密等不同的数学原理，可抵抗不同数目的恶意的参与者。

#### 2.3.2.2 中国剩余定理方案[14]

Asmuth和Bloom于1980年提出了一个基于中国剩余定理(t,n)-门限方案。该方案中，成员的共享是由秘密S得出的数y对于不同模数m1,m2,…,mn的剩余。

令p,d1,d2,…,dn是满足下列条件的一组正整数:



对所有的i,gcd(p,di)=1;对i≠j,gcd(di,dj)=1;d1d2⋯dt > pdn−t+2dn−t+3⋯dn

令N=d1d2⋯dt是t个最小整数之积，则N/p大于任意t−1个di。令r是区间:

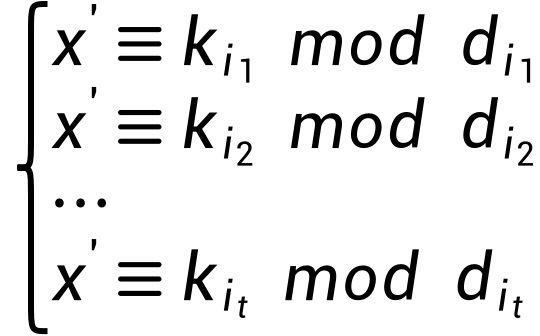
中的一个随机整数,并公布p,r。

(1)秘密分发

将k划分为n个共享,计算k=k+rp,则k′∈[0,N-1]。n个共享为ki=k′moddi, i=1,2,…,n,将子密钥(di,ki)分配给共享者Pi(di是公开的,ki为Pi的秘密共享）。

(2)秘密重构

若给定t个共享,则由中国剩余定理可知,同余方程组:



关于模在内有唯一解x,因为,推出。最后计算出,即。

#### 2.3.2.3 Blakley方案[15]

Blakley于1979年提出了基于高斯消元法的(t,n)门限方案。该方案利用多维空间中的点来建立门限方案，将共享的秘密看成t维空间中的一个点，每个子秘密为包含这个点的t-1维超平面的方程，任意t个t- 1维超平面的交点刚好确定共享的秘密，而t-1个子秘密(即t-1个t-1维超平面)仅能确定其交线，因而得不到共享秘密的任何信息。

(1)秘密分发

生成k-1个随机数{x1,x2,⋯,xk-2,y}mod P,构造平面交点{s=x0,x1,x2,⋯,xk-2,y}

为N个参与方的每一方随机生成{a0,a1,⋯,ak-2},并根据平面公式y=a0\*s + a1\*x1 + a2\*x2 +⋯+ ak-2\*xk-2 + c，计算c=y-(a0\*s+a1\*x1+⋯),将{a0,a1,⋯,ak-2,c}发送给各个参与方。

(2)秘密重构

求解如下矩阵，即可得到平面交点坐标{s=x0,x1,x2,⋯,xk-2,y}

#### 2.3.2.4 Brickell方案[16]

Brickell于1989年提出了基于差分插值定理的(t,n)门限方案。该方案是Shamir秘密共享方案的推广，由一维方程转向多维向量。

(1)秘密分发

选择一个正整数n，表示参与方的数量。确定d与pi，其中pi是d维向量；并确定分配规则，例如{(p1,p2,p3),(p1,p4)}，代表第一、二、三个人可以解密，称为一组规则向量，第一、四个人也能解密，称为一组规则向量。

确定n个向量，并保证一组秘密份额可以线性表示成（1,0,0,⋯）的形式。这些向量所有人都知道。保证在上述同一组规则中的向量可以线性构成（1,0,0,⋯），而不在上述规则里的向量构不成（1,0,0,⋯）。

确定秘密K，和a1,a2,⋯,ad，且a1=K；计算yi=a\*pi，其中a=(a1,a2,⋯,ad)。并将yi作为一份秘密份额给第i个人。

(2)秘密重构

获得满足条件的一组秘密份额，构造（1,0,0,⋯）并找出参数；将参数带入式++mod p=K中，获得秘密K。

#### 2.3.2.5 Additive Secret Share方案

Additive secret share即加性秘密共享。它是安全多方计算中使用较多的一种秘密分享方案，所谓的加性指的是数据的分裂和复原采用的是加法。

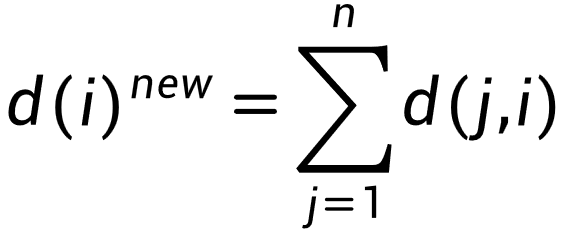
在加性秘密共享的过程中，密码份额具有主动化（Proactivization）的性质。主动化是指秘密份额可以在固定时间间隔后被刷新，以减少攻击者获取秘密的可能性。假设攻击者可以在比较长的时间内获取任何参与者持有的秘密份额，虽然单个秘密份额的泄露并不会揭示整个秘密；但一旦攻击者在较长的时间内获取了所有参与者的秘密份额，就可以复原出整个秘密。这就是主动化特性发挥作用的地方，具有主动化特性的秘密共享方案的所有秘密份额将在固定时间间隔后任意刷新，以使攻击者可能永远无法获得所有最新的共享。

在每个生命周期之后，该方案会生成一组新的加法共享，具体步骤如下：

1.每个加法份额 S(i)被细分为子片段 d(i，1)，d(i，2)……，d(i，n) 这样所有 d(i，j) 的总和产生 S(i) 和 n 是系统中的参与者数量。

2.子片段 d(i，j) 由所属方I分配给所属方j。以这种方式，所有参与者交换所拥有加法份额的子片段。

3.为了计算更新后的份额，将所有从其他参与者接收的子片段累加:



在秘密份额更新的过程中，每个参与者本质上都是对自己的秘密份额进行加法共享，并将产生的子份额分配给参与者。随后，每个参与者将收到的子份额相加。从而实现了在更新参与者所持秘密份额同时，保留原始秘密。

在文献[17]中，Herzberg等人首次在秘密共享方案中引入了定期更新份额的机制(Proactive secret sharing)，以应对长期攻击和信息泄露的威胁。Maram等人[18]提出了CHURP协议，协议在Herzberg等人提出的PSS协议上进行改进，从而使协议的通信复杂度低于以前的方案。

#### 2.3.2.6 Semi2k-SPDZ方案[20]

SPDZ是Damgard等人[19]于2012年提出的一种不诚实多数制计算协议，它能够支持两方以上的计算算术电路。它分为离线阶段和在线阶段。SPDZ的优势在于昂贵的公钥密码计算可以在离线阶段完成，而在线阶段则纯粹使用廉价的、信息理论上安全的基元。SWHE用于在离线阶段执行constant-round 的安全乘法。SPDZ的在线阶段是linear-round，遵循GMW范式，在有限域上使用秘密共享来确保安全。SPDZ最多可以对抗恶意敌手的 t≤n个腐坏方，其中t为敌手数量，n为计算方数。

SPDZ协议主要使用的技术有三部分内容：MAC(消息授权验证码)，保证了不受到恶意敌手方的影响；使用乘法三元组使乘法计算降维成线性组合；在预处理部分使用部分同态加密计算MACs和乘法三元组的分片数据。

Semi2k-SPDZ方案是在SPDZ协议的基础上，由Cramer等人[20]进行改进的方案。该方案选择2k作为模数，简化了SPDZ协议的实现和实际的应用程序，提高了效率。

# 第三章 作品测试与分析

## 3.1 安全性分析与证明

我们创新性地设计了基于ABY3-VSSS的安全隐私密码管理系统，为了确保我们的设计安全可靠，我们将对系统的各个方面进行安全性分析与证明，包括用户-浏览器阶段、服务器阶段以及信道传输等环节。我们作品的整体设计是基于半诚实条件下的，但是在服务器阶段我们补充了针对恶意参与方的应对方案。

在用户-浏览器阶段，我们的设计采用ABY3协议实现用户、浏览器与半诚实第三方的三方模型，我们也采用Cheetah协议实现了用户与浏览器的两方模型。我们采用的ABY3协议与Cheetah协议均为半诚实模型下的安全多方计算协议，具体的内容可见2.3.1节。所以在我们的系统第一阶段之中，是在半诚实条件下安全的。这里需要注意的一点是，两方模型具有更好的安全性，在两方模型之中我们的数据运算均在本机物理设备上进行的，不存在半诚实第三方，所以无需担忧第三方会私自获取本机的秘密份额。但是相应的，两方模型的可扩展性远不如三方模型，具体内容可见2.2.2节内容讲解。

在Web服务器阶段，我们的方案提供了两种模式：性能模式与安全模式。在性能模式下，前提假设大部分的Web服务器在其网络环境之中是可信的，我们集成的Shamir，Brickell以及Additive Secret Share均在半诚实条件下是安全的，所有的秘密共享方案均基于信息论的安全，在门限机制下没有获取足够的秘密份额时是无法重构秘密的。在安全模式下，我们的系统是基于Semi2k-SPDZ协议实现的，所以我们能够处理恶意参与方存在的情形，在恶意参与方参与秘密恢复时，无法通过MAC等认证机制，协议会自动停止运行。

不论是用户-浏览器阶段，还是Web服务器阶段，我们系统安全性不仅仅建立在安全协议的安全机制之中，我们的安全性会随着现有系统安全与网络安全的防护的增强而增强，除非短时间内拿到足够的秘密份额，否则攻击者将无法恢复用户密码这一秘密信息。

相对于用户-浏览器阶段而言，Web服务器阶段面临着网络中更大的风险，我们在Web服务器端秘密份额的分布式存储过程之中加入了节点标记，能够实现泄露后的数据溯源，及时发现异常情形的出现。我们在恢复阶段会随机选取不同序列服务器进行秘密恢复，防止重复登录导致的恢复模式泄露。同时我们的系统中可以动态地调整参与方数目或参与方的秘密份额，可以选择Additive Secret Share方案，定时地去更新Web服务器中的秘密份额，也可以选择Brickell方案来动态地增删参与方节点。在Web服务器阶段，因为Shamir等秘密共享方案缺失对于恶意参与方的检测，所以我们采用了优化后的组合方法来自我检查恶意的参与方，具体内容可见2.2.6节。在策略层面，我们实现了上述的策略，保证了系统的安全性。

最后，我们在证明了系统各阶段安全的基础上，仍需考虑我们在各阶段的连接之处的安全性，如图3-1-1红框标记所示：

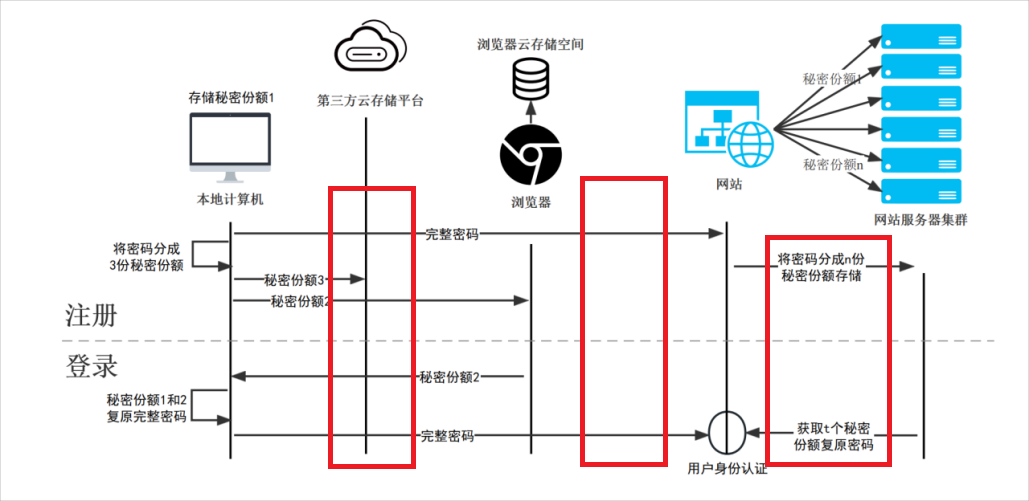


图3-1-1：系统各阶段通信

对于系统各个阶段的连接之处，也就是各个阶段的物理通信过程，我们在该过程之中的安全性是基于网络通信的加密措施的，无论是用户注册阶段传输的账户密码明文，还是通信传输的秘密份额，我们都可以使用HTTPS协议来保证我们系统各阶段通信时的安全性。

至此，我们的系统整体，包括各个阶段以及阶段间的通信连接都已经证明，在半诚实条件下是安全的了；在Web服务器阶段我们的系统还能抵御恶意方的参与。**我们的作品是可证明的，理论与现实情景下安全的。**

## 3.2 原型测试说明

我们提供了三个版本的测试程序以及其对应的可视化网页并将对应的功能融合进一个我们实现的四六级报考网站。这些程序基于ABY3协议进行了设计，在用户、半诚实第三方和网页之间建立了安全通信渠道。在网站服务器端，我们分别使用了Shamir、ASS方案和Semi2k-SPDZ，以确保数据在传输、存储和使用过程中都得到了充分的保护。

在Semi2k-SPDZ协议中，我们并未添加任何其他功能的秘密共享方案设计，以确保协议的简洁性。在Shamir方案中，我们通过对每一个秘密份额增加服务器签名的方式来完成数据泄露的攻击溯源，即网站服务器可以根据反馈的信息完成自检，在不多于t台服务器被攻陷的情况下可以准确识别被攻陷的服务器。我们利用ASS的特性，允许各个服务器上的秘密份额进行动态刷新，刷新后的秘密份额依然可以依照相同的恢复规则组合出原来的秘密。这样做可以增强秘密份额的安全性，避免长时间不更新引起的可能的安全风险。

ABY3-Semi2k的原型测试侧重于提供ABY3-VSSS的安全隐私密码管理系统的运行全流程展示。当用户进行登录时，如果浏览器检测到用户输入的用户名曾经保存过，浏览器将自动提供密码占位符，并使用ABY3协议从用户、浏览器和半诚实第三方中恢复出密码。然后，将恢复出的密码与使用Semi2k-SPDZ协议从网站服务器恢复出的密码进行比较。如果比较成功，则用户登录成功；否则，用户登录失败。如果浏览器检测到用户输入的用户名未曾保存过，则浏览器不提供任何占位符。此时，需要使用用户输入的密码与服务器端的密码进行比较。如果比较成功，则用户登录成功；否则，用户登录失败。

当用户进行注册时，用户需要输入自己的用户名和密码作为注册信息。浏览器会检查用户输入的用户名是否与已有的用户名重复。如果输入的用户名已存在，注册将失败；否则，注册将成功。注册成功后，用户的密码将使用ABY3协议分成三份，并放置在用户设备、浏览器和半诚实第三方中。同时，用户的密码也会根据Semi2k-SPDZ协议分成n份，并分别保存在n台服务器中。

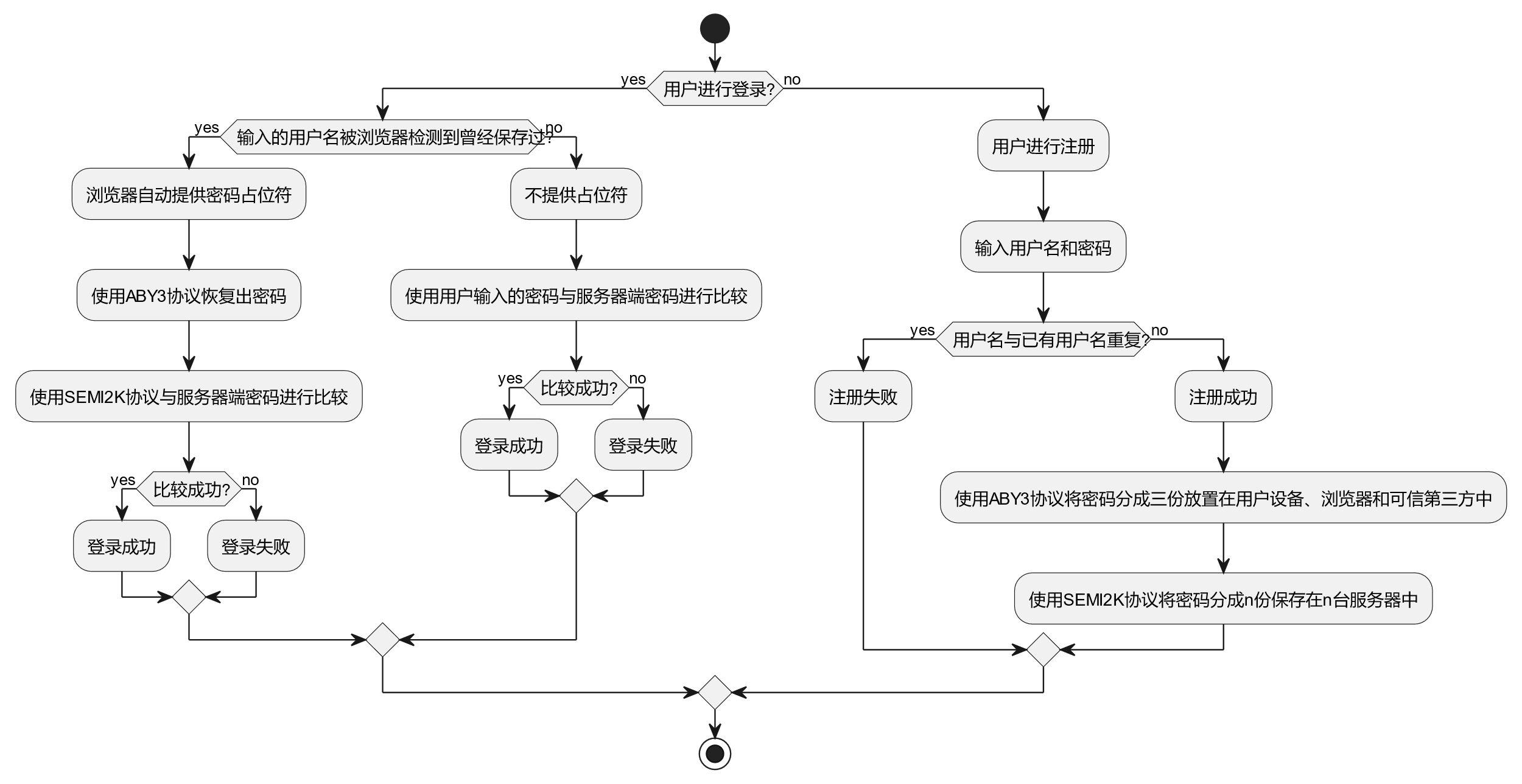


图3-2-1：ABY3-Semi2k原型测试流程图

ABY3-Shamir的原型测试的基本流程与ABY3-Semi2k的流程保持一致，只是服务端使用Shamir门限秘密共享方案对密码进行保存。并且，我们在Shamir门限秘密共享方案的基础上添加了数据泄露的攻击溯源功能，即网站服务器可以根据用户反馈的信息完成自检，在不多于t台服务器被攻陷的情况下可以准确识别被攻陷的服务器。

当攻击者点击攻击按钮时，系统会提示攻击者选择想要攻击的服务器台数k。如果攻击者输入了大于n台或小于1台的数字，则输入失败。系统还会要求攻击者输入想要更改的用户信息。攻击者需要选择一个已经存在的用户u进行攻击。如果选择了不存在的用户，则同样提示输入失败。再次点击攻击按钮后，n台服务器中会有k台遭受攻击。遭受攻击的服务器将随机更改u的Shamir秘密份额。

当用户进行登录时，如果浏览器检测到用户输入的用户名已保存过，浏览器将自动提供密码占位符。然后，在用户点击登录按钮后，使用ABY3协议从用户、浏览器和半诚实第三方中恢复出密码，并与使用Shamir门限秘密共享从网站服务器恢复出的密码进行比较。如果比较成功，则用户登录成功；否则，用户登录失败。如果用户一直登录失败，怀疑服务器异常，可以点击自检按钮让服务器进行自检。服务器会启动自检程序，如发现异常会告知用户存在异常。如果能够识别被攻击的服务器，则告知用户被攻击服务器的序号。如果无法识别被攻击的服务器，则报告有大量服务器被攻击。

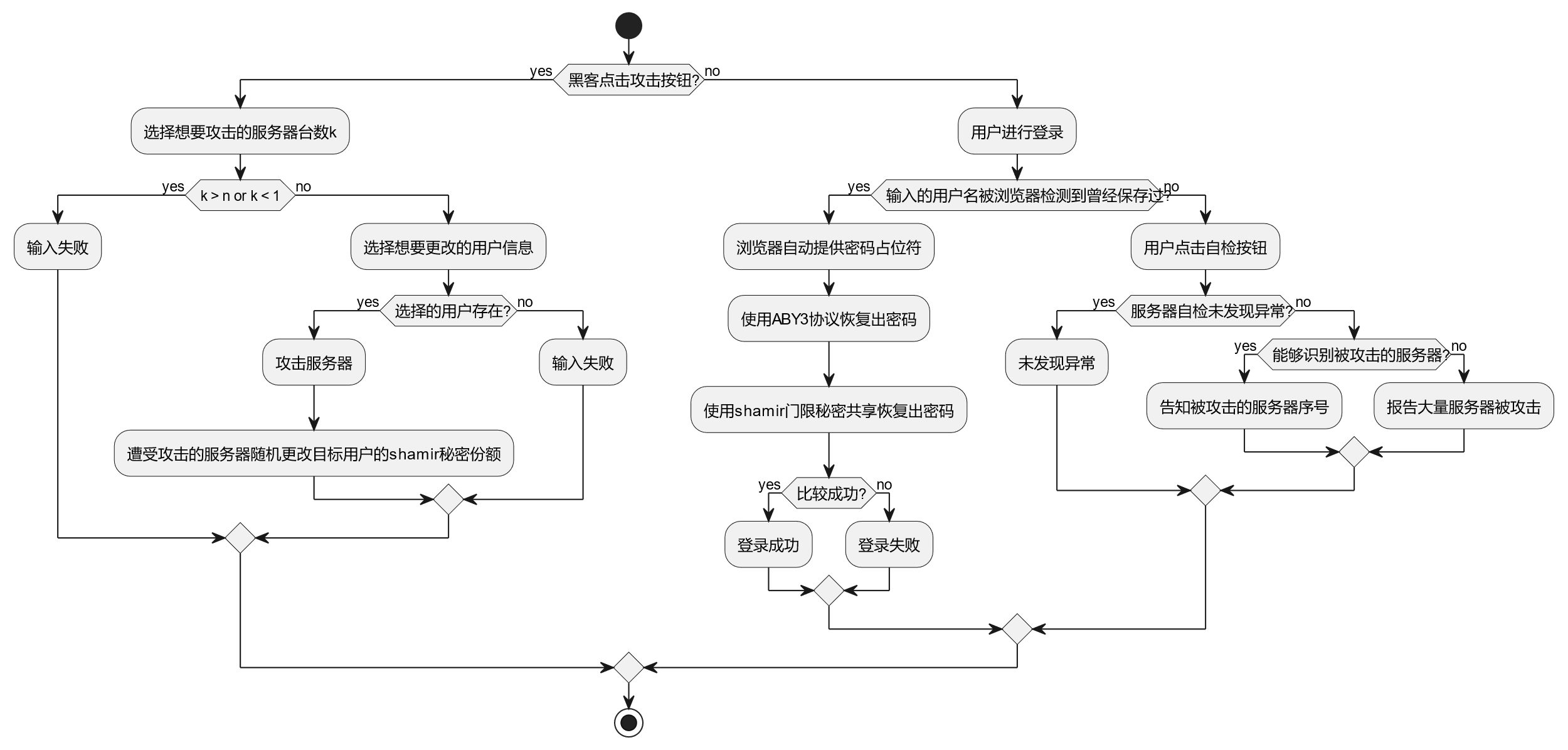


图3-2-2：ABY3-Shamir原型测试流程图

ABY3-ASS协议的基本登录注册流程与ABY3-Semi2k基本保持一致，我们在原有架构的基础上增加了刷新功能，即用户可以点击刷新按钮更新存储于服务器端的秘密份额而无需和本地、浏览器和半诚实第三方做过多交互。

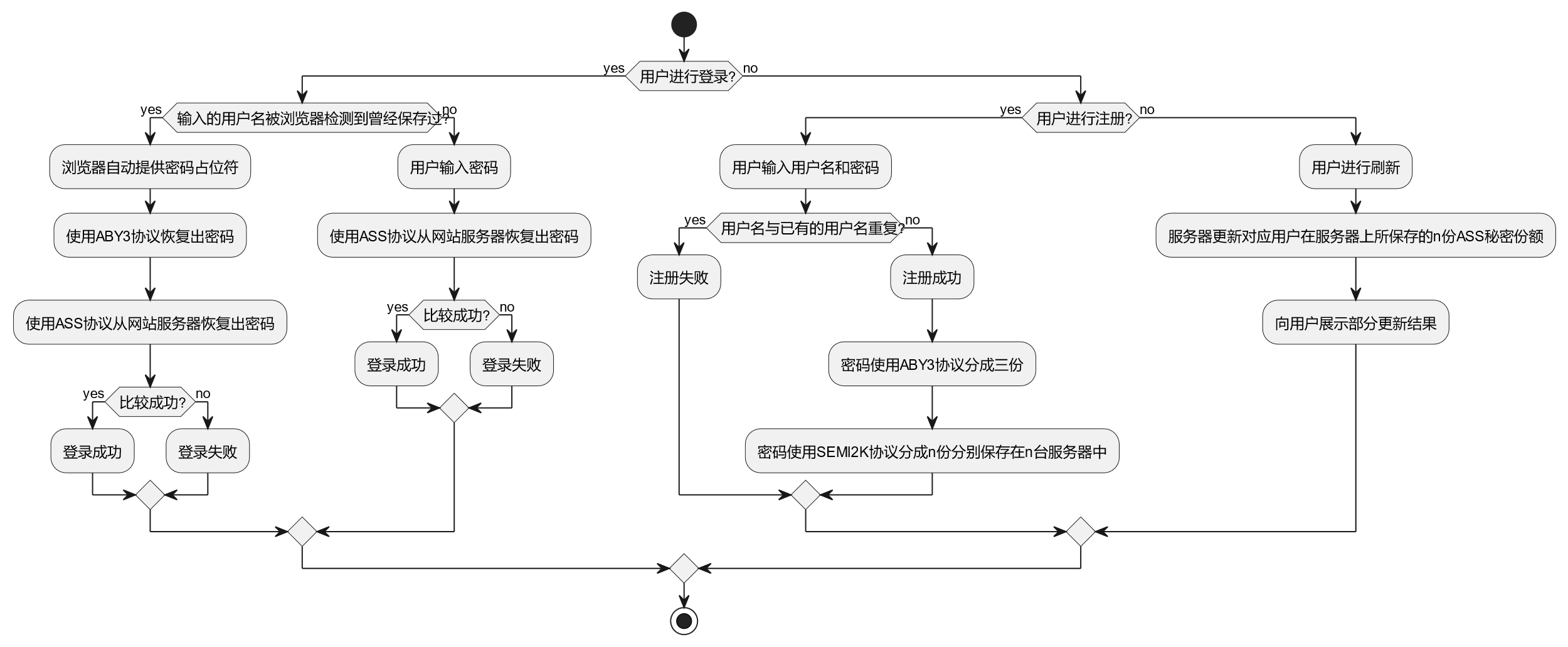


图3-2-3：ABY3-ASS原型测试流程图

## 3.3 系统各流程UI展示

### 3.3.1 系统UI展示

为了证明我们设计的基于ABY3-VSSS的安全隐私密码管理系统在现实世界中的可用性，我们将该系统融入进了由我们搭建的四六级考试报名网站并进行了应用测试。



图3-3-1：四六级考试管理员页面

我们搭建的四六级网站包括管理员以及考生两个用户权限，网站具体功能包括用户注册登录、考试信息查看、考试报名和缴费以及在线答题和阅卷等功能，这与现实世界中的考试系统基本保持一致。



图3-3-2：四六级考试报名页面

### 3.3.2 日志记录

为了进一步体现我们方案的详细流程，以备管理员查看和调试，我们为整个系统添加了十分详尽的日志系统，管理员可以通过分析日志记录在不侵犯用户隐私的情况下检查系统不同物理节点的工作情况是否正常。

下图展示了使用ABY3-ASS密码管理方案，考生Alice登录四六级报考网站时系统输出的日志记录，日志第1-2行展示了浏览器搜索其自身是否保存过Alice的用户名以及秘密份额，检索结果发现浏览器确实保存了其对应信息，于是浏览器对密码输入栏进行了自动填充，填充的内容并不是真实的密码，因为ABY3本质上是一个(2,3)门限方案，仅凭浏览器一方的秘密份额无法完成密码的重构，所以浏览器只能对密码输入栏做占位处理以提示用户无需输入密码。当用户点击登录后4-5行展示了用户-浏览器-半诚实第三方使用ABY3协议恢复密钥以及网站的五台服务器使用ASS协议恢复密钥的过程，八个节点分别从自身物理设备中读取对应秘密份额的内容并分别完成恢复，最后将恢复处的两个密码在服务器端进行比对，观察日志发现比对成功，则Alice顺利完成登录工作。

图3-3-3：系统日志记录

下图展示了使ABY33-ASS密码管理方案，考生fox注册四六级报考网站时系统输出的日志记录。当用户第一次注册时，系统会提示用户是否保存账户名和密码，在本例中fox选择了保存密码并点击注册按钮。于是，按照ABY3和ASS协议的具体内容，秘密信息被分别分为了3块秘密份额和5块秘密份额分存于用户-浏览器-可信第三方以及服务器端，为了确保系统的完备性，我们的系统会在之后分别尝试恢复一次fox的秘密份额并进行比对，如果比对成功说明秘密份额被正确分配和存储，系统则打印成功注册账户日志。

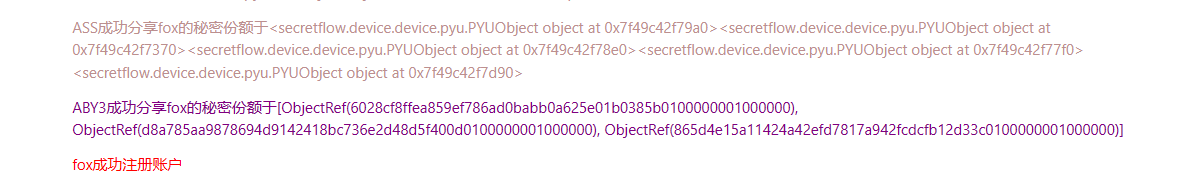


图3-3-4：注册日志记录

# 第四章 创新性说明

## 4.1 用户端启用全新的密码管理方式

在以往的密码管理系统中，用户会使用浏览器帮助存储密码，一旦浏览器存储密码的云服务器受到攻击、或浏览器“好奇”获取用户存储的密码信息，那用户密码的隐私问题就会受到威胁。在我们设计的方案中，用户在客户端将自己的密码使用**ABY3协议**分成**三份秘密份额**，在浏览器中存储一份密码的秘密份额而不是完整密码，在用户的本地计算机存储一份秘密份额，第三份秘密份额可以存储在一个可信的第三方平台；只有获取两份秘密份额并进行恢复，才有可能获取完整的密码。在这种方案设计中，只拥有一份秘密份额的浏览器没有获取用户密码完整的信息的可能，这种密码管理方式为保证用户密码的安全性和隐私性提供了一个全新的实现思路。

在上述情境下，如果存储在本地计算机或浏览器上的秘密份额丢失，用户可以尝试从第三方获取存储的秘密份额并恢复出原始密码，我们的密码管理方案具有抗丢失的能力。

我们的设计方案中，也充分考虑了用户的使用意愿。如果用户不信任第三方平台，我们提供了**基于Cheetah协议的半诚实两方模型**，该模型将用户个人隐私密码仅仅划分为两个秘密份额，分别存储在用户本机与浏览器之中。虽然这样做会牺牲一定的鲁棒性和密码恢复功能，但同时也提高了该阶段的安全性，从而让用户能够完全放心。

## 4.2 服务器端集成多种秘密共享方案

在服务器端，服务器存储密码的传统方式是存储用户密码的明文、哈希值或加盐哈希值于单台服务器上，一旦这台服务器遭受攻击，攻击者就可截获大量的用户密码信息。在我们的方案中，服务器端会部署分布式秘密共享的存储模式，将每位用户的密码分成n份（n>=5）秘密份额，每台服务器仅存储用户密码的一份秘密份额，只有获取至少t（t>=3）份秘密份额后，网站服务器才能恢复出用户的原始秘密并进行认证。使用**多台服务器联合存储**，攻击者至少需要攻破t台服务器的安全防御机制，获取一个用户密码的t份秘密份额才能恢复出用户的原始密码信息，从一台服务器到t台服务器，使用秘密共享的存储模式，极大地提高了攻击者的攻击难度。

此外，我们还创造性地提出**可变的秘密共享模式(VSSS模式)**。即在服务器端集成多种可用的秘密共享方案，有的方案具有更快的计算速度、有的方案具有更强的安全属性，服务器可以根据需求，选择需要的方案进行部署。我们将这些方案分成两种模式：性能模式和安全模式。

在**性能模式**下，Web网站认为大多数服务器都是诚实可信的，并且具有较高的安全防护级别。那么我们可以使用Shamir、Brickell和Additive Secret Share等秘密共享方案，这些方案实现简单并且计算性能优良。在选择性能模式时，服务器需要确保服务器环境相对安全可信。

对于需要更高安全级别的场景，我们更建议网站服务器选择**安全模式**，该模式提供了更多的安全机制和保护措施。该模式采用Semi2k-SPDZ协议，在协议运行过程中可以检测到恶意方后停止运行，以防止数据泄露。

## 4.3 系统安全鲁棒性扩展

虽然，我们作品中的系统设计相比于之前的方式更加的安全，但网站服务器仍然存在被攻击的风险。一旦某台服务器受到攻击，攻击者可能会发布或篡改所截获的秘密份额。为了快速锁定遭受攻击的服务器，我们在服务器存储每条秘密份额时连接**该服务器节点编号或哈希值**，从而在数据泄露时可以快速确认是哪台服务器泄露数据。

如果某台服务器被攻击者劫持，攻击者可能会篡改存储的秘密份额信息，导致网站使用错误的秘密份额进行恢复，进而导致恢复失败或生成错误密码的情况。本作品提出了一种优化后的组合方法，用于帮助网站确认是否有服务器被绑架，并快速找出被绑架的服务器。

此外，在我们的方案中，服务器端存储秘密份额的服务器数量 n 是可以**动态调整**的。我们在VSSS模式中集成了Brickell秘密共享方案，该方案支持动态增减参与者的数量，但该方案在加密和解密过程中的计算开销会增加，相对于Shamir方案而言，效率会有所降低。在VSSS模式中，我们还使用了Additive Secret Share方案，该方案的密码份额具有**主动化（Proactivization）**的性质，秘密份额可以在固定时间间隔后被刷新，来达到秘密份额不是一成不变的效果，来改变攻击者获取不同秘密份额的内容，从而大大提升攻击的难度。

## 4.4 实现本方案应用SecretFlow框架

在作品开发实现的过程中，我们使用了SecretFlow框架。SecretFlow是2022年7月发布的一个国产隐私保护数据分析和机器学习的统一框架，具有强大的包容性和可扩展性，坚持完备性、透明性、开放性、连接性的原则，可以提供以下三方面的功能：

1. 将多方安全计算（MPC）、同态加密（HE）、可信执行环境（TEE）等隐私计算技术抽象为密文设备，将明文计算抽象为明文设备。

2. 基于抽象设备的计算图，使数据分析和机器学习工作流程能够表示为计算图。

3. 基于计算图的机器学习/数据分析能力，支持数据水平/垂直/混合分割等场景。

作为一个开放的框架，SecretFlow框架简化了隐私计算技术的使用，让用户能够轻松地进行数据分析和机器学习建模，而无需深入了解底层技术细节。在设备层，框架与密码学、可信硬件和硬件加速等领域的专家密切合作，以提高协议安全性和计算性能。同时，该框架提供了良好的设备接口，第三方隐私计算协议可作为设备插拔式接入。在算法层，为机器学习提供了灵活的编程接口，算法开发者可以很容易定义自己的算法。通过上述方式，SecretFlow框架促进了隐私计算的广泛应用和更好的用户体验。SecretFlow提供了**密态计算设备SPU**，SPU是SecretFlow Processing Unit的简称，它作为隐语平台的密态计算单元，为隐语提供安全计算服务。SPU实现了多种MPC协议，包括：

1.ABY3:一个多数诚实的3PC协议，SPU实现了半诚实模型。

2.Semi2k-SPDZ :一个半诚实NPC协议。和SPDZ协议类似，但是需要一个可信第三方来产生离线的随机数。目前，在协议中默认使用可信第一方。

3.Cheetah :一个快速半诚实的2pc模型。因为协议不需要半诚实第三方，因此会需要一些计算开销。

我们的作品利用SecretFlow框架提供的SPU与PYU实现了上述三种MPC协议在安全隐私密码管理系统的使用。

# 第五章 总结

在现有的用户-浏览器-网站服务器的流程建模中，即使使用了加密和哈希等密码保护措施，仍然存在着一些潜在的隐私和安全问题。

1. 用户无法准确知道浏览器的行为以及对登录信息的处理，浏览器有可能轻松访问并偷窃用户存储的密码明文或者AES加密后的密码等敏感信息。
2. 即使浏览器没有未经授权地访问过这些信息并使用安全性较强的对称加密算法对用户密码进行了存储，用户主机和浏览器云端可能遭受病毒入侵或木马攻击，导致用户密码泄露。这种情况下，攻击者可以使用穷举攻击、离散字典攻击、差分攻击等方法，在相当可观的时间内破解用户的密码。
3. 由于网站服务器将大量用户的密码明文或者哈希值存储在单个服务器上，攻击者攻陷其中的单台服务器就可以获取到大量用户的密码信息，或使得网站登录崩溃。一旦用户的密码被泄露，攻击者可能会通过使用该密码访问用户的其他账户和个人信息，甚至进行钓鱼等攻击。

为了解决这些问题，我们提出了一种创新性的密码管理系统，该系统是基于ABY3-VSSS的安全隐私密码管理系统。在用户浏览器阶段，我们的方案将原始密码分为多个份额，由用户、浏览器和半诚实第三方所共有。除了改进用户和浏览器的密码管理方式，我们还对网站服务器的存储方式进行了优化。我们采用了可变的秘密共享模式VSSS对密码进行存储，从而有效解决了单台网站服务器被攻陷后造成的问题。通过使用秘密共享方案，我们实现了极强的鲁棒性和可扩展性，任何一方被攻击的情形下，我们仍然能够成功恢复密码并保持密码的机密性。我们还集成了数据泄露溯源模块、动态调整模块和自我检查模块，充分保护用户的隐私与数据安全。

总之，我们的方案是一种创新性的密码管理方案，它有效解决了现有密码管理系统中存在的不安全和潜在高风险问题。它应用了近期提出的秘密共享技术，确保了极高水平的用户安全和个人数据保护，实现了网站服务器端用户密码的安全存储，集成了数据泄露溯源与被劫持服务器自我检查等功能，为未来的密码安全和隐私保护提供了新的方向和参考。

# 参考文献

1. Jarecki S, Kiayias A, Krawczyk H, et al. Highly-efficient and composable password-protected secret sharing (or: How to protect your bitcoin wallet online)[C]//2016 IEEE European Symposium on Security and Privacy (EuroS&P)
2. Jarecki S, Kiayias A, Krawczyk H. Round-optimal password-protected secret sharing and T-PAKE in the password-only model[C]//Advances in Cryptology–ASIACRYPT 2014: 20th International Conference on the Theory and Application of Cryptology and Information Security
3. Bagherzandi A, Jarecki S, Saxena N, et al. Password-protected secret sharing[C]//Proceedings of the 18th ACM conference on Computer and Communications Security
4. Jarecki S, Kiayias A, Krawczyk H, et al. TOPPSS: cost-minimal password-protected secret sharing based on threshold OPRF[C]//Applied Cryptography and Network Security: 15th International Conference, ACNS 2017
5. Roy P S, Dutta S, Susilo W, et al. Password protected secret sharing from Lattices[C]//Applied Cryptography and Network Security: 19th International Conference, ACNS 2021
6. Demmler D, Schneider T, Zohner M. ABY3-A framework for efficient mixed-protocol secure two-party computation[C]//NDSS. 2015.
7. Mohassel P, Rindal P. ABY3: A mixed protocol framework for machine learning[C]//Proceedings of the 2018 ACM SIGSAC conference on computer and communications security. 2018: 35-52.
8. Huang Z, Lu W, Hong C, et al. Cheetah: Lean and Fast Secure {Two-Party} Deep Neural Network Inference[C]//31st USENIX Security Symposium (USENIX Security 22). 2022: 809-826.
9. Shamir A. How to share a secret[J]. Communications of the ACM, 1979, 22(11): 612-613.
10. Chor B, Goldwasser S, Micali S, et al. Verifiable secret sharing and achieving simultaneity in the presence of faults[C]//26th Annual Symposium on Foundations of Computer Science (sfcs 1985). IEEE, 1985: 383-395.
11. Feldman P. A practical scheme for non-interactive verifiable secret sharing[C]//28th Annual Symposium on Foundations of Computer Science (sfcs 1987). IEEE, 1987: 427-438.
12. Pedersen T P. Non-interactive and information-theoretic secure verifiable secret sharing[C]//Advances in Cryptology—CRYPTO’91: Proceedings. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001: 129-140.
13. Stadler M. Publicly verifiable secret sharing[C]//Advances in Cryptology—EUROCRYPT’96: International Conference on the Theory and Application of Cryptographic Techniques Saragossa, Spain, May 12–16, 1996 Proceedings. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001: 190-199.
14. Asmuth C, Bloom J. A modular approach to key safeguarding[J]. IEEE transactions on information theory, 1983, 29(2): 208-210.
15. Brickell E F. Some ideal secret sharing schemes[J]. J. Comb. Math. Comb. Comput., 1989, 6: 105-113.
16. Blakley G R. Safeguarding cryptographic keys[C]//Managing Requirements Knowledge, International Workshop on. IEEE Computer Society, 1979: 313-313.
17. Herzberg A, Jarecki S, Krawczyk H, et al. Proactive secret sharing or: How to cope with perpetual leakage[C]//Advances in Cryptology—CRYPT0’95: 15th Annual International Cryptology Conference Santa Barbara, California, USA, August 27–31, 1995 Proceedings 15. Springer Berlin Heidelberg, 1995: 339-352.
18. Maram S K D, Zhang F, Wang L, et al. CHURP: dynamic-committee proactive secret sharing[C]//Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security. 2019: 2369-2386.
19. Cramer R, Damgård I, Escudero D, et al. SPD: efficient MPC mod for dishonest majority[C]//Advances in Cryptology–CRYPTO 2018: 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19–23, 2018, Proceedings, Part II. Cham: Springer International Publishing, 2018: 769-798.
20. Damgård I, Pastro V, Smart N, et al. Multiparty computation from somewhat homomorphic encryption[C]//Advances in Cryptology–CRYPTO 2012: 32nd Annual Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2012. Proceedings. Springer Berlin Heidelberg, 2012: 643-662.