鸿蒙生态应用安全技术

白皮书 V1.0

HUAWEI

商标声明

以上为华为公司的商标（非详尽清单），未经华为公司书面事先明示许可，任何第三方不得以任何形式使用。

注意

华为会不定期对本文档的内容进行更新。

本文档仅作为使用指导，文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

华为终端有限公司

地址： 广东省东莞市松山湖园区新城路 2 号

网址： https://consumer.huawei.com

CONTENT

07 应用运行生命周期安全

08 展望

Chapter 1

前言

随着移动互联网技术的飞速发展和电子设备的快速普及，各类应用程序呈现爆发式增长，应用程序已渗透到人们工作、生活的各个方面，应用程序的安全问题也日益凸显。

应用程序由开发者提供，经过多种途径分发到用户手中。开发者拥有技术知识和资源，能够创建应用程序，而用户则依赖于这些应用程序来满足自己的需求，同时，用户对应用程序的内部运行和数据处理方式了解有限，导致他们在使用过程中容易受到应用行为不受控、用户隐私泄露等问题的困扰。这种信息不对称，导致应用开发者与用户之间存在着不平衡关系。

生态构建者作为连接应用开发者与用户的纽带，在其中起着至关重要作用。生态构建者是指参与构建整个移动应用生态系统的实体，包括设备厂商、操作系统和应用商店等。他们为开发者提供开发环境、工具套件和市场渠道，同时也负责维护和改进生态系统的稳定性和安全性。生态构建者对于移动应用的成功和推广起着至关重要的作用，同时生态构建者还致力于推动用户教育，增强用户的数字素养和自我保护意识，使其能够更好地管理自己的数据和隐私。

本白皮书从行业应用安全的典型问题分析入手，给出鸿蒙应用生态构建的安全目标，进一步详细介绍了鸿蒙生态在应用全生命周期不同阶段的安全设计。

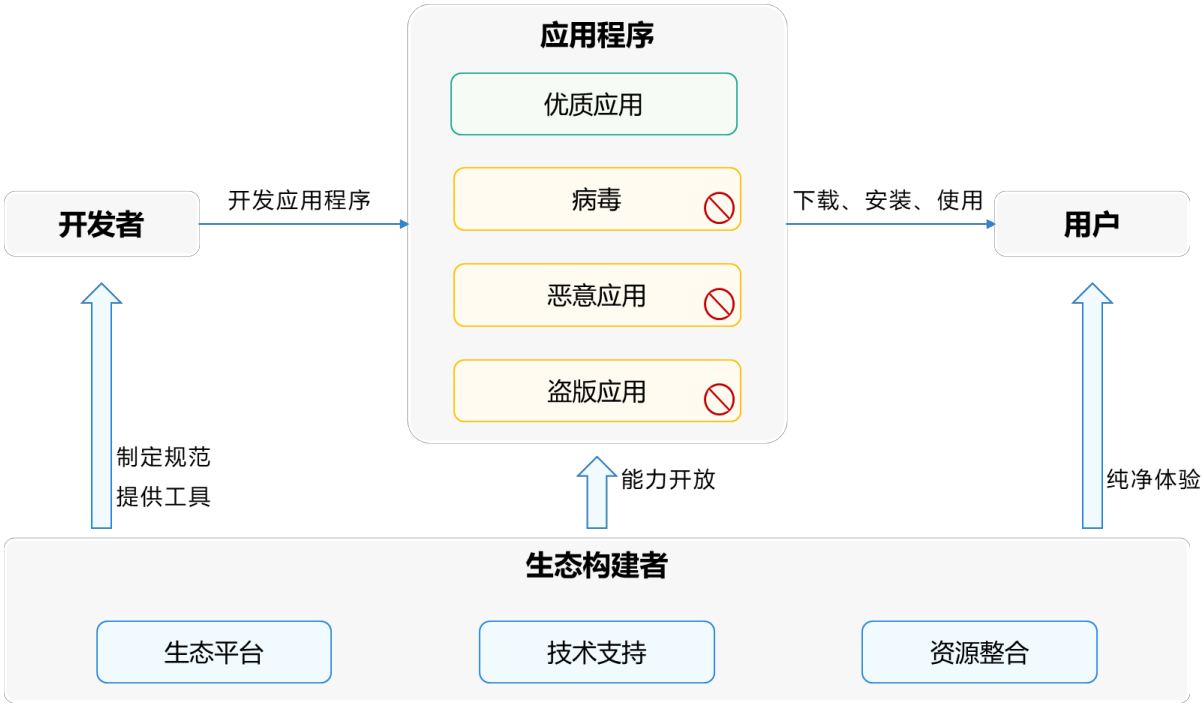
我们致力于建立一个健康、可持续发展的移动应用生态系统，把数字世界带入每个人、每个家庭、每个组织，构建万物互联的智能世界。

Chapter 2

鸿蒙应用生态安全模型

围绕着各式各样的应用程序，**用户**、**应用开发者**和**生态构建者**三个角色共同组成了一个应用生态系统。在这个系统中，三个角色相互依存、相互影响。安全，是这个生态系统的一个重要属性，因为一旦其中任何一个角色出现安全问题，都可能会对整个生态系统造成严重的影响。

一个安全、干净的应用生态系统，需要以下三个角色共同努力才能达成：



用户在应用生态系统中是最终使用者，也是最重要的角色。用户的需求和反馈是应用开发者和生态构建者进行产品迭代和优化的重要依据。用户的体验和满意度直接影响着应用程序的市场表现和生态系统的健康发展。与此同时，用户在使用应用程序时，也有获取良好纯净体验、保护个人隐私的基本安全需求。

应用开发者是应用生态系统中的核心角色。他们通过开发应用程序来满足用户的需求，通过不断地优化和改进自己的产品，以满足用户需求和市场变化。同时，他们也需要确保其

应用程序不会对用户造成任何安全威胁。这包括确保应用程序没有漏洞、没有植入恶意代码，并且能够及时更新以修复任何安全漏洞。

生态构建者是应用生态系统中的组织者和推动者。他们通过建立生态平台、提供技术支持和资源整合等方式，为应用开发者和用户提供更好的服务和体验。同时，他们需要确保整个生态系统的安全性，只有保障了安全，才能让用户和应用开发者都放心地使用和开发应用程序。

首先，生态构建者需要考虑用户的安全需求。用户的安全需求包括个人信息的保护、支付安全、网络安全等。生态构建者需要在平台构建相应的安全机制，如加密技术、身份验证、等，开放给应用程序使用，以保护用户的安全与隐私。

其次，生态构建者还需要考虑应用开发者的安全需求。应用开发者需要保护自己的知识产权、代码安全等。生态构建者需要为应用开发者提供相应的安全保障，如应用加密、代码签名等，以保护应用开发者的权益。

最后，生态构建者还需要考虑整个生态系统的安全。生态系统中的每个应用程序都可能会影响整个生态系统的安全。生态构建者需要对应用程序进行安全审查，确保应用程序没有安全漏洞和恶意行为，不会对整个生态系统造成安全威胁。

当生态构建者不作为，或少作为时，就会产生一系列的应用安全问题。下面我们列举了一些典型的问题，并给出生态构建者面临的挑战。

Chapter 3

应用安全面临的问题与挑战

- 1）诱导用户下载安装恶意应用
- 2）窃取用户数据
- 3）强制推送广告
- 4）利用漏洞攻击其他应用程序
- 5）盗版软件

从 PC 普及的时代，就开始存在病毒应用泛滥的问题，至今无法根除。PC 用户在日常使用过程中，常常因为下载软件或访问网站，导致用户计算机感染病毒应用。攻击者一般通过病毒应用破坏计算机系统可用性，配合其他社会工程学手段，达到获取利益的最终目的。

移动设备作为用户可随身携带的个人设备，用户每天使用时间更长，操作更频繁，也存储了更多的个人敏感信息，自然成为攻击者的目标。恶意应用一般在用户不知情的情况下安装在用户设备上，并在后台运行，执行恶意行为。这些恶意行为可能包括窃取用户的个人信息、监视用户的活动、强制推送广告等。

下面，我们对典型的应用作恶方式进行分析：

1）诱导用户下载安装恶意应用

为了给后续的恶意行为做铺垫，攻击者需要首先确保恶意应用的大量安装。因此，攻击者会想方设法将其开发的恶意应用分发到用户的设备上。

一般情况下，开发者可以通过官方应用分发平台，如苹果的 App Store，谷歌的 Google Play，华为的应用市场进行应用分发。官方的应用分发平台作为应用程序触达用户的第一道关卡，其运营者在应用程序上架时应做好审核管理工作。

然而，当系统内的应用下载安装不限制在官方的应用分发平台时，则给了攻击者可乘之机。攻击者会通过各种手段，如虚假广告、社交媒体、恶意链接等，诱导用户下载并安装应用。这些应用脱离监管，应用质量参差不齐，分发到用户设备上，会通过进一步的恶意行为来影响用户。

如何做好应用质量的监管，控制应用分发渠道，避免恶意应用分发到用户设备上，是生态构建者面临的第一个挑战。

2）窃取用户数据

用户数据是企业和组织的重要信息来源，可以用于市场营销、客户服务、产品开发等方面。攻击者可以通过窃取用户数据来获得经济利益，例如出售给黑市上的数据买家、进行身份盗窃、进行网络诈骗等。攻击者还可以利用用户数据进行勒索，例如勒索企业或个人支付赎金以避免数据泄露。攻击者还可以收集用户的浏览历史、位置信息、通讯录等数据，然后将这些数据出售给广告商。

为了窃取用户数据，攻击者开发的恶意应用在安装或运行时，要求用户授予不必要的权限，例如访问联系人、短信、相机、麦克风等，以便攻击者可以访问用户的个人信息。

对于大多数应用程序，在某些场景下获得用户数据有其必要性。生态构建者不能直接拒绝应用程序获得这些数据，但也不应该把是否提供数据的选择权，完全交给用户。开发者和用户之间天然的信息不对称，开发者拥有更多的技术知识和经验，而普通用户则相对缺乏安全认知和经验。攻击者作为一类特殊的开发者，会利用这种信息不对称，欺骗用户扩大授权范围。

如何提供**安全的数据授权机制，避免用户过度授权**造成的安全威胁，是生态构建者面临的第二个挑战。

3）强制推送广告

广告收入是恶意应用程序的主要收入来源。恶意应用通过提供部分功能获得用户的使用，然后通过提供广告媒介来实现牟利。恶意应用还会通过强制用户观看广告、弹出广告窗口、在应用程序中插入广告等方式来获取广告收入，包括虚假的赚钱机会、虚假的商品宣传等，不仅会影响用户的体验，还会对用户造成经济损失。

为了达到强制推送广告的目的，恶意应用首先利用系统提供的一些拉活保活机制提升自己的活跃时间。在存活状态下则利用系统通知、后台弹窗等系统机制来强制推送广告给用户，并且通过伪造关闭键、复写返回键等方式，避免用户关闭广告。这种恶意的广告推送行为，同时损害了多方利益，既影响用户正常使用设备和应用，又让用户对广告主产生抵触情绪。

保活拉活、系统通知、后台弹窗等，是系统提供的功能。生态构建者开放这些功能给应用程序使用的初衷，是为了让开发者可以给用户提供更好的用户体验。然而，这些功能被恶意应用使用后，反而给用户造成了很多困扰。

应该给应用程序开放什么系统功能，以及对于需要开放的系统功能**如何做到不被恶意利用**，是生态构建者面临的第三个挑战。

4）利用漏洞攻击其他应用程序

考虑到软件生态的复杂性、相互依赖性和外部威胁，在现实世界中，软件很难做到完全没有漏洞。软件生态中各种应用程序、操作系统、库文件、框架和其他软件组件，均由不同

的开发团队设计、开发和维护，涉及数百万行甚至更多的代码。即使软件交付过程中经过严格的测试和审查，仍然可能存在未被察觉的漏洞。

在应用程序的开发过程中，人为错误，设计缺陷和逻辑错误等问题，都可能导致潜在的漏洞存在。攻击者出于自身利益的考虑，会不断寻找软件弱点，一旦攻击者找到漏洞，就会进一步利用漏洞，通过制造缓冲区溢出、注入恶意代码等方式，改变应用程序原本行为，控制程序作恶达到攻击者的目的。

从用户角度，一旦发现应用程序中存在恶意行为，从现象上只能得出应用程序作恶的结论，无从知晓恶意行为是出于应用程序本意，还是由于应用程序被攻击。这些攻击既损害用户利益，又让应用开发者的信誉受损。

如何帮助应用程序最小程度地受到漏洞影响，是生态构建者面临的第四个挑战。

5）盗版软件

盗版软件横行是一个广泛长期存在的问题。随着技术的发展和互联网的普及，盗版软件已经成为了全球范围内的一种威胁。免费获取付费软件的诱惑以及经济利益的驱使，使得盗版软件的大量传播和使用。

盗版软件主要是通过仿冒应用程序或重新打包应用程序产生的。盗版软件带来了诸多负面影响。

首先，它侵犯了软件开发者的知识产权，剥夺了他们应有的利益和回报。这不仅会降低软件开发者的积极性和创造力，也可能限制了软件行业的发展和创新。

其次，盗版软件有潜在的安全风险。盗版软件通常通过篡改或注入恶意代码，给用户设备和个人信息带来潜在的威胁。用户下载和使用这些软件时很难得知其中是否隐藏恶意程序，从而对个人隐私和信息安全构成严重威胁。

最后，盗版软件也不利于合法软件市场的健康发展。免费盗版软件降低了用户对正版软件的需求，使得合法软件市场受到冲击。这不仅影响了软件开发者和厂商的利益，也限制了软件行业的繁荣和创新。

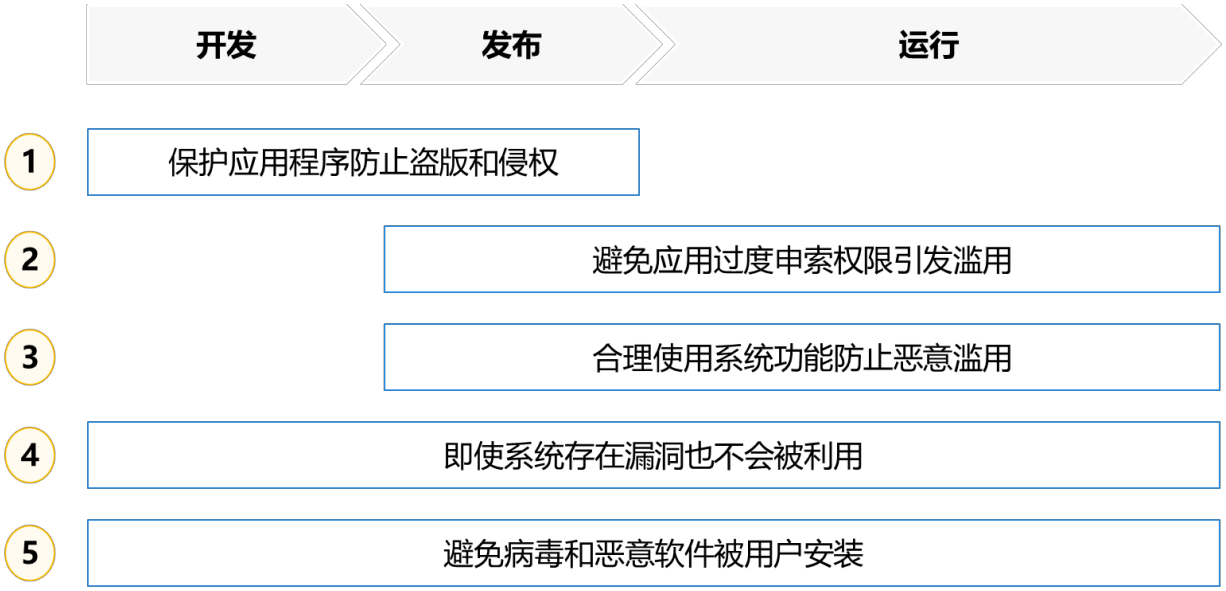
要彻底解决盗版软件问题，一方面需要政府加强知识产权保护力度，完善相关法律法规，加大对盗版软件的打击力度，另一方面，也需要软件开发者和厂商加强技术保护手段，提高软件安全性和防护能力。

如何为应用程序提供有效的核心数字产权保护手段，避免出现盗版软件问题，是生态构建者面临的第五个挑战。

Chapter 4

鸿蒙应用生态的安全目标

保护开发者和用户利益的同时维护整体系统的安全性，对生态构建者是至关重要的。以开发者为中心，构建端到端应用安全能力，保护应用自身安全、运行时安全，保障开发者权益，是鸿蒙应用生态构建的核心目标。



应用生命周期主要分为开发、发布和运行三个阶段。生态构建者为应对上述安全挑战，需要在应用生命周期的三个阶段分别给出解决方案。

为达到上述目标，在应用开发阶段，**我们要确保开发者身份的合法性，为开发者提供安全的开发工具，帮助开发者提高应用程序的安全性。**

在应用上架发布阶段，我们应**确保应用质量。**应用程序应该满足**权限最小化，数据使用公开透明，无不良内容，无恶意行为**等基本要求。同时**也要保证开发者的应用程序安全可用不被篡改，开发者的知识产权受到保护。**

在应用运行阶段，我们首先**确保应用运行环境安全**，同时**也要确保应用行为可知可控。**对有恶意行为的应用，要**建立分级管控措施**，根据应用行为的严重程度，按要求对应用的权限或能力、应用安装包、应用开发者等采用不同粒度的管控方式。

如上图所示，我们在应用全生命周期的不同阶段，分别提供不同的关键技术和措施，来解决生态构建的重大挑战问题。下文将对这些技术和措施分别展开介绍。

Chapter 5

应用开发安全

- 1）开发者注册和实名认证
- 2）申请开发者证书
- 3）申请应用 Profile 文件
- 4）代码安全检查
- 5）代码混淆

应用开发安全是指在开发过程中嵌入安全能力，使应用程序从源头上安全可靠。

开发者是应用程序的创作者，合法的开发者是创作出安全、可靠应用的前提条件；为了保证应用开发者身份真实可信，我们通过开发者证书对应用进行签名，保证应用来源可靠和完整性不被破坏；通过签发应用 Profile，保证 HarmonyOS 对应用关键属性和敏感能力进行有效管控；

应用开发安全构建过程中，一方面需要保证生态中的应用不作恶，另一方面是为了增强应用的安全性，避免应用被攻击，生态构建过程中通过代码安全检查减少安全漏洞，通过代码混淆结合应用发布阶段的应用代码加密，可以显著增加应用被逆向的难度，从而保证应用自身代码安全，保护开发者知识产权。

本章节重点阐述鸿蒙生态是在应用开发安全环节构建的安全能力。

1）开发者注册和实名认证

在开发鸿蒙应用之前，开发者首先需要在 HarmonyOS 应用开发官网完成注册并进行实名认证。认证通过后，官网为开发者分配开发者 ID，用于后续申请应用开发所需的开发者证书和应用配置文件，开发者开发的所有鸿蒙应用都会关联到对应的开发者 ID，可以有效地对应用进行追溯。

鸿蒙应用上架到鸿蒙应用市场分发前，应用市场会对应用进行严格地审核，确保鸿蒙应用的安全和质量。一旦在审核时发现应用不满足应用市场上架标准，会及时通知应用开发者对应用进行整改或优化。

综上，通过开发者注册和实名认证，我们可以确保鸿蒙应用的来源都是可信的。

2）申请开发者证书

鸿蒙应用开发者在拥有开发者 ID 后，可以申请开发者证书，用于后续对其开发的鸿蒙应用进行签名。鸿蒙应用签名是鸿蒙应用必须包含的内容，用于校验鸿蒙应用的完整性和来源可靠，只有签名校验通过，才能在应用市场发布，以及在 HarmonyOS 上安装。

鸿蒙应用的开发者证书遵从 X.509 公钥证书标准，证书中包含开发者公钥以及用于校验的证书链等信息；在应用上架和安装时，都会基于 HarmonyOS 信任根证书对鸿蒙应用的开发者证书进行校验，校验成功后再使用开发者证书中的开发者公钥校验鸿蒙应用安装包的完整性。

开发者证书中的开发者公钥，由开发者生成并提交给 HarmonyOS 应用开发官网；开发者公钥对应的私钥由开发者保管，开发者需要确保私钥的安全，严格管理私钥的访问权限，避免私钥泄露、损坏等情况出现。

鸿蒙应用开发者证书分为两类：调试证书和发布证书，分别用于鸿蒙应用调试阶段和上架发布阶段（开发者证书申请流程详见官网）。

- 调试证书：调试证书仅允许对调试应用签名，不能用于其他目的。为了平衡安全性和开发效率，使用调试证书签名的调试应用可以不经上架检测，直接运行在HarmonyOS设备上，限制条件是：调试应用只允许运行在指定的设备上（该设备的设备ID需要与应用调试Profile中的设备ID匹配），以限制其运行设备的范围。

- 发布证书：发布证书用于正式上架应用的签名。只有使用发布证书签名的应用才允许上架应用市场。使用发布证书签发的应用不对运行设备的ID进行限制，但必须经过应用市场检测上架后，才能分发到用户的设备上。应用市场会对经过检测的上架应用进行应用市场重签名。

综上，鸿蒙应用都是经过开发者证书签名的应用，在应用上架和安装时会对应用签名进行强制校验，确保安装在 HarmonyOS 设备上的应用安装包都是未经篡改和来源可信的；并且可以通过开发者证书关联到经过实名认证的开发者。

3）申请应用 Profile 文件（HarmonyAppProvision）

开发者进行实名认证获取开发者 ID，以及申请开发者证书之后，还需要申请待开发应用的 Profile 授权文件。应用 Profile 授权文件是应用的身份证明文件，用于 HarmonyOS 对应用进行识别和管理，该文件中包含应用的关键信息，包括应用的包名、应用受限权限（仅少数场景才需要申请）、应用的开发者 ID、应用的开发者证书等。

应用 Profile 授权文件需要在开发者网站申请并审核后签发，该文件使用 ECC 密钥签名，签名符合 PKCS#7 标准。应用 Profile 授权文件是应用必不可少的组成部分，被打包到应用安装包中，在应用上架审核和安装时进行校验。

应用 Profile 授权文件按照用途分为调试 Profile 和发布 Profile（应用发布 Profile 申请流程详见官网）。

- **应用调试Profile**，仅用于应用调试场景，不能用于应用上架。开发者使用开发者调试证书和应用调试Profile本地签署调试应用；调试应用可以直接运行在开发者本地

设备上，无需经过上架审核。由于未经上架审核的调试应用质量是未知的，因此为了保护用户权益，HarmonyOS强制要求调试Profile中必须包含本地设备ID列表和授权权限列表，调试应用只能运行在列表中的设备上，每个调试Profile包含的设备ID有最大数量限制。除了对设备限制，少数场景需要申请的受限权限（如读写联系人），也需要在应用Profile中申请。

- **应用发布Profile**，用于正式应用发布上架到应用市场，不能用于应用的本地调试。开发者使用开发者发布证书和应用发布Profile签署应用后，提交到应用市场，应用市场会审核应用的整体安全性和授权权限的使用合理性，通过上架审核后才允许上架应用市场。开发者需要认真审视应用市场上架审核规范，避免由于审核不通过导致的应用程序重复开发。

4）代码安全检查

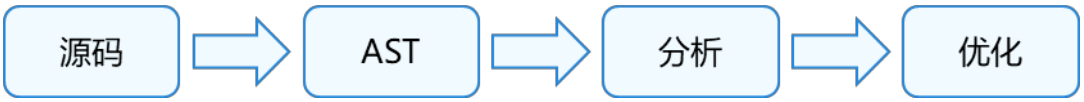
鸿蒙生态服务于开发者，帮助开发者开发高质量应用，有效提高代码质量，减少代码中的潜在安全风险。在 DevEco Studio 中集成检测能力，可以帮助开发人员及时发现代码中的问题，避免在后期发现问题，也可以提高代码的可维护性和可读性。

DevEco Studio 提供了 code linter 功能，对代码的通用规范性检测，同时也会重点对代码的安全性做检查，包括静态代码分析，二进制安全分析等，静态代码分析可以在开发过程中发现问题，开发者可根据扫描结果中告警提示手工修复代码缺陷（详见代码检查规则表）。

开发者完成编译构建后，可通过 DevEco Studio 安全检测插件对二进制进行安全检查，重点检测应用攻击面管控不足、权限或证书配置不当等可能会导致的安全风险，建议开发者在发布应用之前完成相关安全检查动作，并基于风险评估进行修改，其中主要安全风险及漏洞来源于攻击面分析（详见应用安全管理）。

5）代码混淆

移动应用的代码安全非常重要，为了保护应用开发者的代码，避免应用被恶意逆向分析，提高攻击者分析代码的难度，DevEco Studio 中默认提供了代码混淆能力，混淆后的 JS、TS、ArkTS 代码，不容易被逆向后读懂，混淆功能支持对名称进行混淆，包括对类、方法等做混淆处理。



代码混淆方案是基于源码混淆，将源码转为抽象语法树（AST），在 AST 上进行作用域分析和符号分析，混淆名称和属性，移除开发期间的日志打印代码，合并语句，压缩代码体积，生成 sourcemap 文件用于编译混淆后应用的调试，生成 namecache 文件用于热更新修复，在保证运行时性能无变化的前提下，有效保护鸿蒙开发者核心知识产权免受恶意逆向分析。

Chapter 6

应用发布安全

- 1）应用检测与审核
- 2）应用加密
- 3）应用签名

为了确保进入鸿蒙生态中的应用是安全可靠的，在应用上架发布阶段，我们首先对应用进行检测与审核，确保应用程序满足权限最小化，数据使用公开透明，无不良内容，无恶意行为等基本要求，同时我们也提供了应用加密、应用签名等功能，保护开发者应用程序完整性和机密性，保护开发者知识产权。

1）应用检测与审核

为了确保生态中的应用是纯净、安全、合规的，在应用上架前，会基于行业监管要求、应用市场安全隐私要求、应用提供的安全隐私声明，通过静态检测、动态检测、人工检测的多重检测手段，对应用进行全面检测，包括不限于权限检测、应用行为检测、病毒检测、隐私检测等，对于不合规的应用，开发者需要完成合规整改，具体要求详见应用市场审核指南。

权限检测

应用权限检测是指通过技术手段分析应用所申请权限列表的合理性和必要性，同时权限检测会评估应用程序是否存在过度请求权限、滥用权限、频繁申请权限而影响用户正常使用的行为，通过权限检测可以让开发者发现权限问题，保护用户的个人信息。

应用行为检测

应用行为检测主要是一种用于发现恶意行为的安全技术，通过监测应用程序的接口调用、文件访问等行为来识别不良软件或应用程序。为了给用户提供安全可靠的使用环境，应用不得含有试图滥用或不当使用任何网络、系统机制、系统功能、系统漏洞、以及干扰其他应用

或影响终端设备功能的恶意行为，影响用户的正常操作和体验。包含但不限于借助系统机制进行恶意保活拉活、恶意弹窗霸屏、异常后台行为、恶意隐匿等行为。

隐私检测

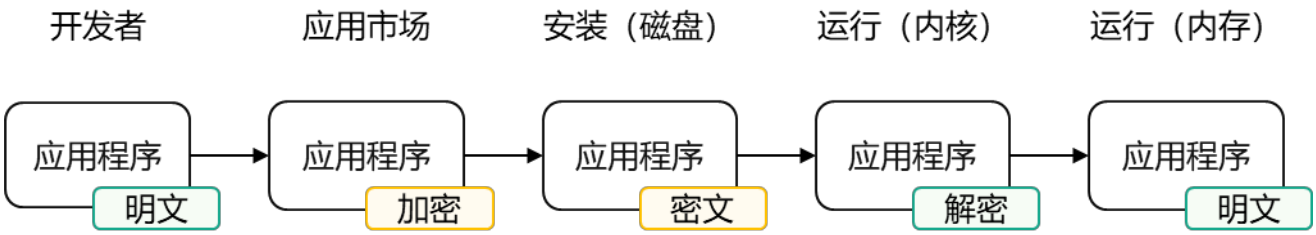
隐私检测是指在应用上架前，检测应用程序在运行中，是否收集了过多的个人信息。例如，某些社交媒体应用程序可能会要求使用者提供太多的个人信息，这可能会导致使用者的个人信息泄露，应用不应该出现违规收集个人信息、超范围收集个人信息、违规使用个人信息等隐私合规问题。

恶意应用检测

应用及其内置组件不得含有病毒、木马，包括但不限于通过可疑代码、文件及程序等形式对系统造成危害影响或侵害用户权益。鸿蒙生态将与业界安全厂商进行合作，对应用中的可疑代码片段、文件或者三方库，与业界安全厂商的病毒库进行对比，检查应用程序是否包含病毒风险。

2）应用加密

为了保护应用代码安全，保护开发者的核心资产，HarmonyOS 提供了端到端的应用代码保护机制，该机制以系统安全为基础，构建内核级应用生命周期内的代码安全保护能力。



开发者向应用市场提交上架申请，再经过应用市场审核后，应用市场会对上架应用做代码加密，应用在设备上安装时，安装文件落盘后仍是处于加密状态，有效的保护应用程序；当应用程序启动时，通过内核加载的应用文件是加密状态，因此这些文件会在内核中按页解密，然后提取文件明文在内存中执行。应用加密采用标准 AES 加密算法，解密后的明文只存在于内存中，不会存储到设备，形成端到端的加密方案，有效的保障应用程序的安全性。

3）应用签名

应用包签名

开发者使用发布证书和发布 Profile 对应用安装包签名后，上架应用市场。应用市场会对上架应用进行上架检测和质量审核，对于满足上架要求的应用，应用市场会对应用安装包进行重签名；只有经过重签名的上架应用，才允许在设备上安装。

HarmonyOS 对所有安装的应用都需要进行签名校验，确保应用来源可信和应用完整性；应用安装包签名校验发生在应用安装时，如果签名校验失败，则禁止应用安装。

HarmonyOS 使用根 CA 对应用安装包进行签名校验，应用安装包的签名证书都需要从根 CA 开始采用证书链的方式签署。

对于调试应用的安装，HarmonyOS 在校验安装包签名基础上，还需要严格匹配应用调试 Profile 中的设备 ID 与当前设备 ID 的匹配，如果不匹配的话禁止安装。

对于发布应用的安装，HarmonyOS 仅允许经过应用市场审核通过后，由应用市场重签名过的安装包进行安装。

应用代码签名

应用包签名可以为应用提供分发和安装时的合法性校验和完整性保护，但是恶意软件常在应用安装后，滥用私有的热更新机制，在运行时下载恶意代码，从而绕过应用市场的安全审查。为维护鸿蒙应用生态的纯净与用户体验，HarmonyOS 引入强制代码签名机制，提供加载时和运行时的代码合法性检查以及完整性保护，确保端侧执行的代码来源可靠，未经审核的代码无法执行。

开发者上架的鸿蒙应用在通过应用市场的安全合规审核后，由应用市场签名，企业应用和开发者的调试应用则使用华为颁发的证书做代码签名。HarmonyOS 系统实施强制代码签名检测，在应用代码加载前，强制检查其签名合法性，防止应用执行未经审核、包含恶意行为的代码；在应用代码执行前，按页做哈希校验，验证其内容的完整性，确保代码在存储时未被篡改；在代码运行过程中，提供基于污点的页标记机制，防止代码在运行时被篡改。

为尽量避免恶意代码的安全威胁，实现鸿蒙生态的安全与纯净，达到提高用户体验的目的，针对鸿蒙应用的方舟字节码和二进制机器码提出以下安全原则：

1）禁止应用执行未包含合法签名的代码，包括但不仅限于应用本身、补丁、插件以及共享库的代码；

2）禁止应用修改已签名的代码内容，实现与提交给应用市场宣传不一致的功能特性；

3）禁止应用绕过代码签名的管控。

在维持上述安全原则的情况下，为满足应用对动态代码更新的诉求，HarmonyOS 提供安全的受限运行环境，允许应用在其中执行未签名代码。受限执行环境通过隔离未签名代码并限制其系统能力，确保未签名代码在安全管控之内，不会对系统造成威胁。对于应用解释执行的代码，禁止用于绕过应用市场审核，实现与宣传不一致的功能特性，改变应用的主要用途。

Chapter 7

应用运行生命周期安全

- 1）基础安全架构模型
- 2）基础安全防护
- 3）应用权限管控
- 4）隐私控制设计
- 5）用户身份认证
- 6）应用数据保护
- 7）应用运行时安全检测
- 8）应用风险行为管控
- 9）应用快速修复

应用运行生命周期安全，包括在应用安装、启动、运行以及更新阶段提供安全解决方案，保障应用的安全性和可靠性。

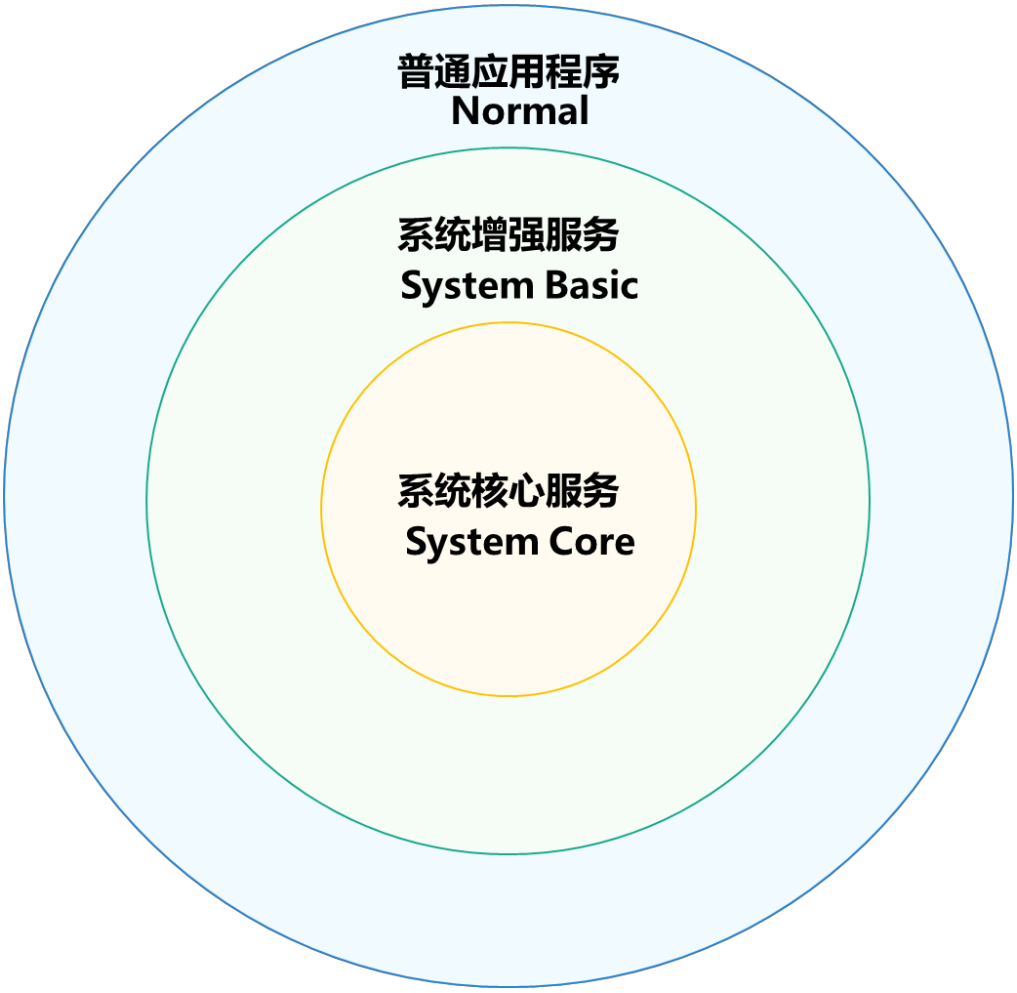
HarmonyOS 在提供基础安全机制，保护应用运行安全的同时，还通过系统安全能力为应用提供用户数据保护的安全手段。

在保护用户隐私上，HarmonyOS 除了提供基于权限管控的隐私保护机制外，更推荐开发者使用 Picker、安全控件等权限细粒度管控手段，从根本上解决权限滥用的问题。

同时，我们也构建了风险行为感知与管控功能，具备快速识别对系统功能或用户体验有严重影响的恶意行为的能力，HarmonyOS 会适时主动限制或撤销恶意应用的相关功能。

1）基础安全架构模型

HarmonyOS 构建**基于洋葱模型的分级安全机制**，是构建运行生命周期安全的基础。HarmonyOS 将应用分为三个 APL（Ability Privilege Level）：normal，system basic 和 system core。应用各自运行在独立的沙盒化环境中，默认仅允许访问自身的文件，如需访问其他应用或者系统的信息，则需要通过权限来实现。



不同 APL 等级可申请的权限范围也不一样，对应规则可以总结为如下表格：

APL 等级	说明
system_core	该等级的应用提供操作系统核心能力。这类应用可申请的权限涉及到开放操作系统核心资源的访问操作，鉴于该类型权限对系统的影响程度非常大，目前只向系统服务开放。
system_basic	该等级的应用提供系统基础服务。这类应用可申请的权限，涉及允许访问操作系统基础服务相关的资源。
normal	普通系统应用和所有三方应用。这类应用可申请的权限，对用户隐私以及其他应用带来的风险很小。

权限管控是 HarmonyOS 提供的众多安全机制中的一个例子。作为生态构建者，HarmonyOS 提供了应用生命周期端到端的安全解决方案，为开发者提供放心的应用开发和运行环境，为用户提供安全的应用和设备使用体验。同时，我们也实施了一系列的监管措施，以确保最大程度地减少恶意应用对鸿蒙应用生态的威胁。

2）基础安全防护

应用程序的基础安全防护是确保系统安全、稳定、以及用户数据安全的重要元素。应用程序给用户带来了巨大的便利，其设计和实现的正确合理至关重要，否则可能破坏系统的完整、稳定，甚至影响用户数据的安全。HarmonyOS 为应用程序提供了多层次的安全防护机制，来确保其不被恶意软件所篡改和窃取信息。上述保护机制确保应用程序访问用户数据的过程是安全可控的。在上述安全机制的加持下，用户可放心地使用应用程序，而无需担心恶意软件、非授权的数据访问等问题。

沙盒隔离与访问控制

HarmonyOS 上运行的应用程序均部署在受保护的沙盒中，通过沙盒的安全隔离，限制应用程序间互相非法访问数据，甚至篡改设备。每一个应用程序拥有唯一的 ID，并基于此 ID 进行访问的识别与限制。应用沙盒限定了只有目标受众才能访问应用内的数据，同时也限定了应用只能访问受限的数据范围，包括：

- 应用程序的安装目录。
- 应用沙盒的数据目录。

- 应用沙盒的分布式数据目录。该目录下存放的文件能够被超级终端内其他设备上的同应用访问。应用需保证放置在该目录下的子目录和文件与其他设备上存储的不冲突。

为确保应用的数据安全，应用私有数据不应设置为可供所有人访问。如需将文件数据提供给其他应用，可选择应用间文件分享。通常情况下应用无法访问用户公共文件数据，如需访问可申请访问用户公共存储空间的权限。

HarmonyOS 根据应用的 APL 等级设置进程域和数据域标签，并通过强制访问控制机制限制应用可访问的数据范围，从而在机制上消减应用数据泄露的风险。

漏洞防利用

漏洞是威胁系统和应用安全的核心要素。为缓解漏洞对应用安全的威胁，HarmonyOS 构建了应用全生命周期的漏洞治理体系。

在应用程序开发阶段，开发者可利用 DevEco Studio 提供的代码安全检测工具，识别、发现、并修复代码中存在的漏洞，如整数溢出检测、数组越界检查、使用已释放内存等典型的代码漏洞。

在运行阶段，HarmonyOS 提供了程序运行时抑制漏洞利用，并消减漏洞利用危害的安全加固机制，可有效保护应用程序和用户数据。主要有如下安全机制：

- 栈保护。栈溢出是经典的攻击方法，攻击者基于栈溢出漏洞可劫持程序控制流，进而实现攻击目的。利用栈线性溢出的特征，栈保护机制通过在栈特定位置插入canary保护字，防止栈溢出时非法覆盖函数返回地址，从而避免应用程序控制流被非法

劫持。

- ASLR（地址空间随机化）主要针对内存破坏漏洞。可确保应用程序的内存区域在启动时被随机化，包括堆、栈、共享库等，增加攻击者预测攻击目标的难度。如针对return to libc这类攻击技巧，随机化可大幅增加攻击者定位库函数地址的难度。

- DEP（数据不可执行）。同时具备可写可执行特性的内存是攻击者执行恶意代码的最佳条件，利用处理器的XN（不可执行）特性，可将内存页标注为不可执行属性，从而避免攻击者在应用程序中直接注入恶意代码。值得注意的是，HarmonyOS原则上仅在几个安全受控的场景可同时赋予内存写和执行权限。

- CFI（控制流完整性保护）。为应对攻击者常用的JOP/ROP等攻击方法，HarmonyOS同时为应用程序提供了前向和后向CFI保护技术，用于对抗攻击者利用内存错误非法篡改应用程序的函数返回地址和函数指针。CFI技术可在攻击路径的关键节点进行有效拦截，起到极佳的防御效果。

3）应用权限管控

HarmonyOS 以洋葱模型为基础，提供了一种允许应用访问系统资源（例如，通讯录等）和使用系统能力（例如，访问摄像头、麦克风等）的通用权限访问方式。为了防止应用过度索取和滥用权限，HarmonyOS 基于应用的 APL 等级，配置不同的权限开放范围。不同级别的权限对应用开放范围不同。例如，normal 权限表示对所有应用开放，但如果三方应用需要使用 system_basic 权限和 system_core 权限，则必须通过向应用市场申请 HarmonyAppProvision 授权证书的方式，来申请使用这类权限。

合理的使用场景有助于应用权限申请和使用。开发应用时权限申请需要满足如下要求：

- 应用（包括应用引用的三方库）所需权限必须在应用的配置文件中严格按照权限开发指导逐个声明。

- 权限申请满足最小化原则，禁止申请不必要的、已废弃的权限。应用申请过多权限，会引起用户对应用安全性的担忧以及使用体验变差，从而也会影响到应用的安装率和留存率。

- 应用申请敏感权限时，必须填写权限使用理由字段，敏感权限通常是指用户隐私密切相关的权限，包括地理位置、相机、麦克风、日历、健身运动、身体传感器、音乐、文件、图片视频等权限。

- 应用敏感权限须在对应业务功能执行前动态申请，满足隐私最小化要求。

- 用户拒绝授予某个权限后，应用与此权限无关的其他业务功能应能正常使用

应用运行时 HarmonyOS 采用 TokenID（Token identity）来唯一标识应用。TokenID 作为通用的应用标识符，独立于任意一个内核的应用标识符，有助于构建跨内核异构的鸿蒙生态。权限管理服务通过应用的 TokenID 来管理应用的 AT(Access Token)信息，包括应用身份标识 APP ID、子用户 ID、应用分身索引信息、应用 APL(Ability Privilege Level)、应用权限授权状态等信息。在资源使用时通过 TokenID 作为唯一身份标识映射获取对应程序的权限授权状态信息，并依此进行鉴权，管控程序的资源访问行为。

4）隐私控制设计

隐私设计原则

鸿蒙生态始终将用户隐私放在首位。隐私是用户的基本权利，安全是产品的基本属性。基于这种理念，隐私设计基本原则包括：

- 数据最小化：坚持仅采集实现业务功能所必须的个人数据，以服务于应用需求。
- 透明可控：当采集个人数据时，清晰、明确地告知用户，并确保用户知道数据被如何使用，以及如何退出。
- 身份保护：使用隐私增强技术，在数据离开用户的设备时，隐藏用户的身份。
- 数据本地化：坚持尽可能在用户的设备上完成个人数据的处理和分析。
- 数据安全保障：坚实的数据安全是隐私保护的基础，围绕硬件、OS、应用及服务持续构建数据安全能力。

隐私权限

HarmonyOS 提供透明可控机制帮助用户对应用的访问行为可知可控，这些机制贯穿于应用整个运行期间，包括敏感数据或者能力被访问前、被访问中和被访问后各个阶段。

- 隐私权限授权（访问前）：应用访问敏感数据或者能力前需要申请相应的权限，当应用申请权限时，开发者必须填写权限使用理由字段，以便帮助用户理解应用申请此

权限的合理性并作出正确的选择；

- 隐私指示器（访问中）：敏感数据或能力（例如，麦克风）被应用持续访问时，通过状态栏显示实时提醒用户，便于用户感知应用访问行为；

- 权限使用记录（访问后）：HarmonyOS支持用户查看应用访问敏感数据或者能力的历史记录，便于用户完整地审视应用行为。当某个应用长时间未被用户使用或者应用存在风险行为时，HarmonyOS将对该应用权限进行自动回收，保护用户隐私。

除了通过隐私权限方式访问隐私数据或者能力，HarmonyOS还提供无权限的访问方式，包括安全控件和picker机制。

安全控件

隐私权限管理需要用户手动授权，这对于一些普通用户来说可能比较困难，另外，应用程序对某些敏感信息使用可能在不同的使用场景，用户使用应用过程中，隐含着对于权限的授予意愿，安全控件是系统提供的一组系统实现的 ArkUI 基础组件。应用可以自由集成该类组件，当组件被用户点击后，应用将被授予临时授权，无需向用户弹窗授权就可访问受限资源，实现通过识别用户主动行为自动授权的设计思路。

整体方案由安全控件 UI 组件、安全控件管理服务、安全控件增强组成：

- UI组件实现了固定文字图标的样式便于用户识别，同时提供了相对丰富的定制化能力便于开发者定制。

- 控件管理服务提供控件注册管理能力、控件临时授权机制、管理授权生效周

期，确保应用后台、锁屏下无法注册使用安全控件。

- 安全增强部分实现了包括地址随机化、挑战值检查、回调UI框架复核控件信息、调用者地址检查、组件防覆盖、真实点击事件校验等机制，防止应用开发者通过混淆、隐藏、篡改、仿冒等方式滥用授权机制，泄露用户隐私。

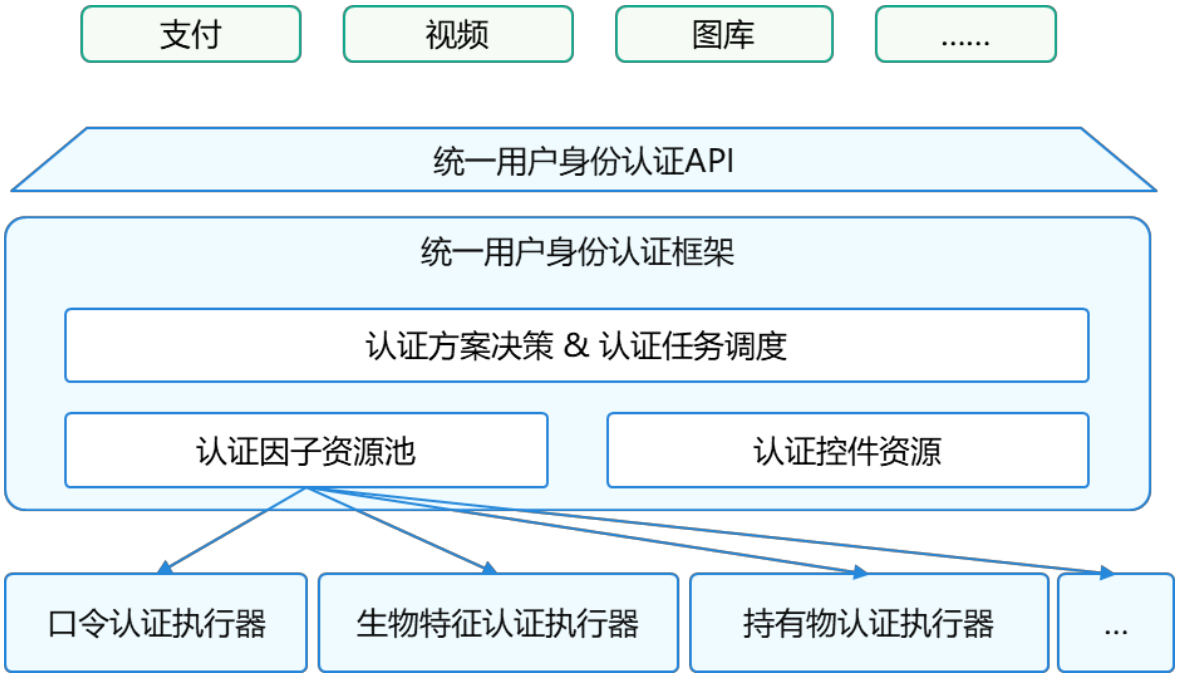
Picker

Picker 是系统提供的一组系统实现的数据选择接口，不同于安全控件，开发者不需要集成 picker 到应用中。应用可以直接调用系统 Picker 接口。借助系统 Picker，应用无需申请权限就可以权限允许范围内的访问敏感数据。应用通过拉起系统 picker 界面，允许用户在预定义的数据集合中选择允许访问的范围，用户选择后应用便可以获取数据。目前，HarmonyOS 提供了六种 picker：

- 照片picker
- 文件picker
- 音频picker
- 相机picker
- 联系人picker
- 电话picker

5）用户身份认证

用户身份认证与访问控制子系统（User Identity and Access Management，简称用户IAM）提供了系统级用户身份识别与认证能力。系统提供的统一用户身份认证 API，可以帮助开发者快速实现安全可靠的用户身份认证，并在认证过程中保护用户的敏感信息和特征数据，防止用户数据泄露。统一用户身份认证框架提供了丰富的用户身份认证功能：包括不同的用户身份认证方式，如口令认证、人脸认证、指纹认证等，以及相应的认证用户界面，可用于锁屏解锁、支付认证、系统服务的二次访问控制等用户身份鉴别场景。 调用统一用户身份认证 API 的应用，在认证通过后可获取在安全隔离环境中签发的用户身份认证通过凭证（authToken），该凭证可以提供给其他系统服务用于证明用户身份，系统服务校验过凭证的有效性后，可以响应请求执行一些需要用户鉴权的系统操作，或是允许访问系统上的用户个人数据。



用户 IAM 子系统主要由统一用户身份认证框架、身份认证控件，以及口令、指纹、人脸等各种因子的认证执行器组成。其中统一用户身份认证框架是核心，负责管理和调度各个认证执行器和认证控件，为上层应用提供多样的用户身份认证功能和统一的身份认证体验。该认证框架支持各种认证执行器弹性接入，使得各种用户身份认证能力可以根据设备硬件支持情况按需部署。

统一用户身份认证框架和各个认证执行器在设备本地的安全隔离环境中维护着用户注册的身份认证凭据信息。这些数据从生成、使用到销毁，整个生命周期都不会离开该安全隔离环境，也不会在认证过程中提供给系统上其他任何服务或应用，更不会在无用户授权的情况下传输到其他设备。

6）应用数据保护

数据生命周期保护策略

HarmonyOS 为用户和开发者数据，提供了全生命周期的安全防护措施，确保在每一个阶段，数据都能获得与其个人数据敏感程度、系统数据重要程度和应用程序数据资产价值匹配的保护措施。

根据数据在智能终端设备上的处理的过程，数据生命周期包括生成（Create）、存储（at Rest）、使用（in Use）、传输（in Transit）、销毁（Destroy）这几个阶段：

- 生成：智能终端和其上的应用软件通过采集、直接生成、从其它终端接收或其它方式转入等方式产生数据。

- 存储：数据在智能终端设备上存留。
- 使用：数据在智能终端设备上被访问、处理等。
- 传输：数据离开源设备、转移到目的设备的过程。
- 销毁：数据在智能终端设备上被销毁，保证其不可被检索、访问。

针对数据 at Rest 保护，HarmonyOS 提供文件级加密与敏感数据加密存储机制。应用开发者可利用该机制保护不同的文件或应用内敏感数据。

针对数据 in Use 保护，通常与访问控制机制关联，HarmonyOS 提供系统级密钥管理机制，应用开发者可将数据的访问控制归约为密钥的访问控制条件，HarmonyOS 密钥管理机制将根据设定的访问控制条件管控密钥的使用，使密钥仅在满足对应条件时可用于解密应用数据或对应用数据进行签名。

针对数据 in Transit 保护，HarmonyOS 原生超级终端可信组网机制可保护超级终端分布式特性的数据流转端端加密。如应用开发者想要进一步管控应用数据在传输出端后权限依然可控、同时限制数据的二次转发行为，可利用 HarmonyOS 提供的文件受控分享机制对待分享的数据进行保护。

文件级加密

FBE（File-based encryption）文件级加密。在启用了 FBE 的设备上，每个用户都有两个可供使用的存储空间，即 CE（Credential Encryption）和 DE（Device Encryption）存储空间。

- CE存储空间：文件加密与用户凭证相关，提供了用户凭证后才能使用，即用户解锁之后才能使用。

- DE存储空间：在设备执行验证启动后就能使用，即在设备启动之后就能使用。

应用产生的数据，开发者可指定数据存储空间，默认情况下存放在CE存储空间下，但是对于某些场景，应用需要在用户解锁前就可访问的文件，例如时钟、闹铃、壁纸等，此时应用需要将这些文件存放到DE存储空间。

敏感数据存储

敏感数据存储，又称为关键资产存储，提供了关键敏感的隐私数据的本地加密存储，应用可以将用户高安全敏感的关键资产短数据（如用户的 APP 账号密码，银行卡号等）在本地加密存储，加密这些数据的密钥存储在安全的隔离区，只有合法的应用才能访问并解密这些数据。与此同时，关键资产存储还支持以下一些安全措施：

- 支持基于用户身份认证的二次访问控制：

  支持设置访问数据需要经过用户的再次即时的授权，如验证用户锁屏口令，或指纹人脸等生物特征。即使用户的设备在解锁状态下给他人使用，他人也无法未经授权访问获取这些隐私数据。

- 支持基于设备当前状态的访问控制：

  支持设置数据在不同设备状态下是否可以被访问，包括以下几种状态：

- 设置密码时有效：数据只有在设备设置了锁屏密码的时候才可访问，如果用户清除设备锁屏密码，数据也会被清除。
- 解锁时有效：只有设备处于解除锁定的状态下，数据才可被访问，如果设备重新被锁定，数据无法被访问。
- 第一次解锁后有效：每次重启设备后，必须在用户输入一次锁屏口令后，数据才可被访问。
- 总是有效：只要设备处于开机状态下，数据都可被访问。

密钥管理

通用密钥库系统（英文全称：HarmonyOS Universal KeyStore，以下简称 HUKS）是 HarmonyOS 提供的系统级的密钥管理服务，为应用提供密钥的全生命周期管理能力，包括密钥生成、密钥存储、密钥使用、密钥销毁等功能。 HUKS 基于系统安全能力，为应用提供密钥全生命周期的安全管理，应用无需自己实现，利用 HUKS 的系统能力，就能确保业务密钥的安全。HUKS 提供的核心安全机制包括以下几点：

- **密钥明文不出安全环境**：HUKS的核心特点是密钥全生命周期明文不出安全环境HUKS Core，可根据设备能力的不同，对接TEE（Trusted Execution Environment）。在支持TEE的设备，HUKS Core运行在硬件安全环境中。上层业务只能通过密钥别名或句柄生成、访问密钥，密钥在安全隔离环境中使用和运算，任何人（包括密钥所属的上层业务自身）都接触不到密钥明文。

- **系统级安全加密存储**：业务密钥使用设备根密钥派生的密钥加密保护，在有硬

件条件的设备上，设备根密钥通常是硬件唯一密钥，只可在安全隔离环境（如TEE）中获取和使用，在这些设备上，业务密钥在安全隔离环境中基于硬件唯一根密钥派生的密钥加密保护，明文不出安全隔离环境，获得TEE级别的存储安全性。此外，在有锁屏口令的设备上，业务密钥还会叠加用户锁屏口令加密保护密钥。

- **严格的访问控制**：只有合法的业务才有权访问密钥，上层应用只能访问属于它自己的密钥（由应用自身生成的密钥），无权访问其他业务的密钥。

同时 HUKS 框架支持用户二次身份认证访问控制以支持应用的高安全敏感场景下安全访问密钥的诉求。应用可以在生成密钥的时候，指定访问密钥需要进行二次身份认证，之后应用访问密钥，只有通过即时的身份验证（如验证锁屏密码，人脸，指纹等），才能访问密钥。身份认证的结果在安全隔离环境中生成和验证，即使设备 REE(Rich Execution Environment)系统被攻击（如被 root），攻击者也无法绕过二次身份认证访问控制非法访问密钥。

文件受控分享

HarmonyOS 面向应用构建了一系列数据防泄漏保护技术，文件受控分享是其中关键的系统级数据权限保护服务。该服务为应用提供了文件跨设备传输后（不限定传输途径）的访问管控能力，保证了发送方对于文件设置的访问控制策略（如只读、编辑等）在接收端能够正确执行。

文件受控分享服务的核心安全机制包括：

- 系统级的文件加解密凭据管理和身份认证机制：系统绑定云服务帐号或者登陆

域帐号后，会申请帐号身份绑定的用于文件加解密保护的凭据，并在系统密钥管理服务内进行管理。用户选中待分享文件后，可以指定授权用户身份、设置文件访问权限、并使用加解密凭据加密待分享文件；系统身份认证机制将保证仅文件的授权用户才能操作权限文件。

- 系统级的权限控制：用户加密保护的文件在独立沙盒单独处理，与普通文件隔离，通过沙盒管控应用打开文件时对于系统服务（如文件操作、屏幕控制等）的访问，从而保证文件的访问权限正确执行。

文件受控分享服务是系统级的机制，在应用程序不适配的情况下，用户仍然可以通过文件管理器实现文件的权限设置和权限文档的打开和权限管控拦截。权限的可设置项包括：文件的授权用户帐号、文件的权限类型（只读、编辑）等。

在应用适配后，可以实现在应用内设置和查看权限的体验，也可以指定受控应用打开文件，拦截应用恶意泄露文档的潜在风险。

7）应用运行时安全检测

应用生态普遍面临既要开放生态能力，又要防止生态能力被恶意利用而侵犯用户权益、违背国家法律法规的情况出现，因此可以说应用运行时安全检测是应用生态安全的最后一道防线；为保护用户的安全隐私需求，系统提供了应用运行时安全检测，以确保应用对获取的行为符合法律法规等要求。运行时的安全检测包含应用对 OS 关键行为的检测、对权限获取的检测以及其他可能会影响用户正常使用的安全检测。通过联合应用管控能力，为用户提供纯净安全的用户体验。

应用运行时安全检测不同于基于样本的应用动静态检测，而是结合应用实际运行时行为、所在系统上下文、应用信誉、开发者信誉等进行综合分析，以判断可能存在的应用风险行为。

HarmonyOS 在构筑业界领先的安全防护能力的前提下，基于威胁感知、分析、响应正向循环的理论模型，构筑起坚实的应用安全生态治理闭环体系。

HarmonyOS 始终关心用户隐私，分析技术的实现都是在端侧，端侧分析的风险结果会经经过脱敏后在云侧进一步基于大数据模型和 AI 模型进行分析，通过聚合关联进一步提升应用行为风险感知的能力。HarmonyOS 云风控系统分析的结果会经过安全运营进一步确认后，联合应用管控能力对恶意应用实时管控，从而为用户提供纯净安全的用户体验。

8）应用风险行为管控

HarmonyOS 为应用程序提供了丰富的接口，支撑应用程序实现丰富多彩的功能，满足用户的各种需求；但是应用开发者也可以通过 HarmonyOS 提供的能力实现一些非法推广销售、违规频繁后台弹框、违规过度申请或诱导用户授予不必要的系统权限等行为，损害系统体验或造成用户信息泄露。

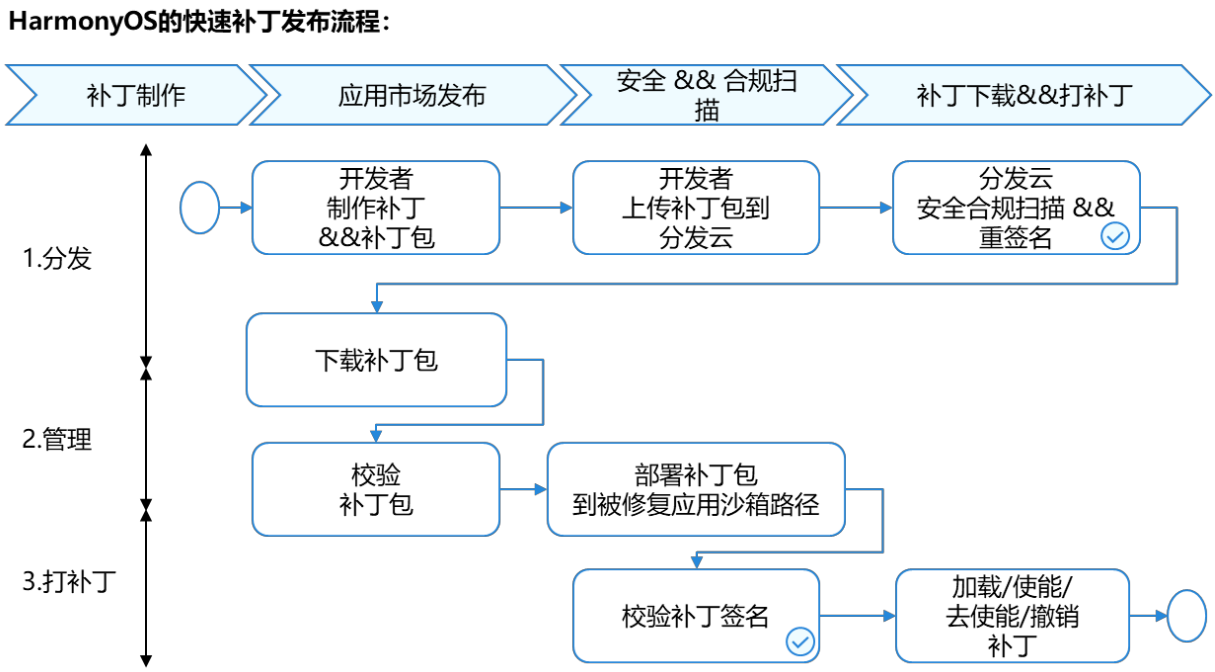
HarmonyOS 为净化应用生态，保障 OS 纯净运行，构建了应用风险行为管控机制，对应用的违规行为进行动态约束。应用风险行为管控依赖 HarmonyOS 应用运行时的安全检测，对已经识别的应用风险违规行为及时下发管控策略，限制应用的风险行为，从而保障用户的合法权益。

应用风险行为管控依赖端云协同机制，云侧对已发现的应用风险违规行为签发应用违规行为管控策略，随着应用的安装或更新下发到端侧；端侧接收到应用违规行为管控策略后立

即生效；应用违规行为策略支持对应用的恶意拉活保活行为、非法干扰用户行为、诱导用户授予权限行为、过度使用系统权限行为等场景进行管控。如果应用整改完毕，修改其违规行为后，Harmony OS 支持更新管控策略，撤销相关应用的管控项。

9）应用快速修复

应用在运营过程中，可能会发现已经发布的应用程序有安全或功能性问题，需要紧急修复，为了解决这种场景下的问题，HarmonyOS 提供了快速修复能力，快速修复是系统提供给开发者的一种技术手段，支持开发者以远快于应用升级的方式对应用程序包进行缺陷修复。和全量应用升级软件版本相比，快速修复的主要优势在小、快且用户体验好。在较短的时间内不中断正在运行应用的情况下（即不需要重启应用），修复应用的缺陷。



快速修复发布流程如上所示，开发者通过 DevEco Studio 制作补丁包，经过安全检查、应用市场重签名后，在端侧才能生效，未经签名校验的补丁包无法实现修复的目的。

Chapter 8

展望

安全是鸿蒙生态建设的基石，只有保障了安全，生态才能健康繁荣。我们呼吁行业人员积极共建，尤其是开发者们发挥创造力和责任心，积极参与鸿蒙生态安全建设，从根本上杜绝恶意行为和数据滥用。鸿蒙生态应用安全需要全行业共同努力，携手合作，相互支持。

在未来的鸿蒙生态中，我们坚信优秀的应用将会得到更多用户的认可，我们坚信安全的鸿蒙生态将吸引更多优秀的开发者加入进来，让我们携手共建一个值得用户信赖、创新蓬勃的美好未来！

HUAWEI