原文：《Solutions to Delay Attacks on Rollups》by Ed Felten，Offchain Labs 联合创始人

编译：隔夜的粥，DeFi 之道

Rollup 协议设计者面临的一个微妙问题是如何应对延迟攻击（delay attacks）。在这篇文章中，我将讨论它们是什么，以及 Arbitrum 如何防范它们，从而带来一些令人兴奋的新发展。

延迟攻击是试图阻止 Rollup 协议取得进展的恶意行为，它们不会攻击协议的安全性，也就是说，它们不会试图强制确认不正确的结果。 相反，延迟攻击通过试图阻止或延迟任何结果的确认来攻击协议的活性（liveness）。

这些问题可能很微妙，老实说，协议设计者们通常不喜欢谈论延迟攻击，但每一个 Layer 2 系统，无论是 Optimistic Rollup、ZK Rollup 还是其他，都需要应对有关延迟和协议进展的问题。

本文深入探讨了延迟攻击问题，并讨论了各种版本的 Arbitrum rollup 协议如何处理该问题。

延迟攻击是什么？

在一次延迟攻击中，一个恶意方（或一组恶意方）在 Rollup 协议内采取行动，遵循旨在阻碍或延迟确认结果返回 L1 链的策略。

这不同于拒绝服务（DoS）攻击，在拒绝服务攻击中，攻击者试图阻止在协议中采取任何行动。相比之下，在延迟攻击中，行动会继续发生，但攻击者的行为方式，会阻碍或延迟结果的确认(即延迟向 L1 提取资产)，并迫使诚实的验证者燃烧 gas 。

任何看似合理的 Rollup 协议都需要参与者进行质押，因此延迟攻击者必然会失去一份或多份质押权益。我们在这里假设攻击者愿意在一定限度内牺牲质押权益，以追求他们的攻击。

我们还将保守地假设，攻击者在将交易上链方面具有优势，因此每当攻击者与诚实的一方竞争获得链上交易的优先权时，攻击者总是会赢。

最后，我们假设攻击者可以审查对底层 L1 区块链的访问，以排除 Rollup 交易，但仅限于一段有限的时间内进行，我们称之为“挑战期”。特别是，攻击者可以启用

和禁用一种概念上的“审查模式”。当启用审查模式时，攻击者可以完全控制哪些交易可以到达 L1。但是，攻击者只能在一个挑战期内启用审查模式。(我们假设任何一组审查模式期加起来超过一个挑战期，将触发 L1
社区的社会反应，以阻止审查尝试。)

评估针对延迟攻击的协议

在评估协议时，我们可以问六个问题：

该协议是否有欺诈证明机制？（否则，延迟攻击就没有实际意义，因为参与者不能延迟确认任何结果——即使是欺诈性结果也不行。）是否有中心化运营商或证明者，可以通过简单地停止或扣留数据来阻止进展？ 如果是这样，那一方可能会造成无限延迟。该协议是否为最终进展提供了无需信任的保证？ 换句话说，无论攻击者做什么，一个诚实的参与者是否可以强制最终取得进展？如果协议保证无需信任的进程，那么攻击者可以造成的延迟上限是多少？攻击者的成本如何与造成的延迟时间成比例？诚实方回应的总成本如何衡量？

确定这些标准后，让我们评估 Arbitrum rollup 协议的两个历史版本。

协议 1 ：学术论文协议

2018 年的Arbitrum 学术论文?大致使用了以下协议（忽略与此无关的一致模式）。任何质押者都可以坚称提议的结果，我们称之为断言（assertion）。在一个时间窗口内，其他质押者的任何子集都可以挑战断言，断言者必须对每个挑战者捍卫自己的断言，一次一个挑战者。在每次挑战结束时，输的一方将失去他们的质押权益。

（请注意，有必要允许多个质押者反对断言，并给每个挑战者机会来推翻断言。这是必要的，因为恶意方可能会故意输掉他们“应该”赢得的挑战。给每个挑战者一个单独的挑战，可以确保一个诚实的挑战者可以击败不正确的断言，无论有多少恶意方故意输掉挑战。）

如果没有挑战，或者断言者赢得了所有挑战，那么断言将得到确认，协议将继续前进。但是如果断言者输掉了任何挑战，它的断言将被拒绝，并且协议状态将回滚到做出断言之前的状态。

评估

该协议具有有效的欺诈证明，但不保证进展，因为恶意参与者可以无休止地做出不

正确的断言，每次都会牺牲质押权益，但会导致做出和拒绝相同断言的无休止循环，导致持续回滚和缺乏进展。

协议 2：分叉和裁剪

当前的 Arbitrum 协议（自 2020 年以来已部署在每个版本的 Arbitrum 上），通过引入分支改进了之前的协议。其思想是允许多个质押者做出相互竞争的断言，并将相互竞争的断言视为分叉链。然后，一系列的挑战让分支相互对抗，最终裁剪掉所有分支，只留下一个分支，让剩下的一个分支得到确认。

具体的方式如下。区块链中的每个 Rollup 区块都会跟踪其第一个子区块（即后继区块）由质押者创建时的时间戳。其他质押者可以创建额外的子区块。
每个子断言都隐含地声称它的所有年长的区块都是不正确的。

创建断言的质押者需要对其进行质押，其他质押者也可以选择对其进行质押。

如果两个质押者对姊妹（siblings）断言下注，并且两个质押者都没有在挑战当中，那么两个质押者之间可以发起挑战，其中两个姊妹断言中较早的质押者正在捍卫该年长姊妹断言的正确性，而另一个质押者正在挑战该正确性。输掉挑战的质押者，将丧失其质押权益，并被移除质押者集。

该协议包括行动的最后期限。首先，创建父断言的子断言的截止日期，是创建第一个子断言之后的一个挑战期。其次，对断言进行质押的截止日期是该断言创建后的一个挑战期。

如果对断言进行质押的截止日期已过，并且没有质押者对该断言进行质押，则该断言将被删除。 被修剪断言的任何子代、孙代或其他后代也同时被修剪掉。

如果一个断言早于一个挑战期，并且没有未修剪的姊妹断言，则可以确认该断言，从而代表协议取得进展。

评估

该协议具有有效的欺诈证明，没有中心化运营商或证明者可以阻止进程，因此任何参与者都可以推动进程。为了推动进程，一个诚实的质押者可以发布一个正确的子断言（如果还不存在的话）。在此之后，在截止日期之前将经过有限的时间，以确保不会创建更多的姊妹断言，也不会有更多的质押者可以在现有的姊妹断言上下注。从那时起，诚实的质押者将参与一系列挑战，一次击败和移除错误质押的一方（如果有多个诚实的参与者参与质押，他们可以同时击败对手）。一旦所有这些参与

方都被移除，就可以确认正确的子断言。

针对此协议最有效的延迟攻击，是让一个恶意质押者对一个不正确的姊妹断言进行质押，并让 N-1 个恶意质押者对正确的姊妹断言进行质押。无论有多少诚实的质押者押注在正确的姊妹断言上，攻击者都能够安排不正确的质押者在诚实方发起挑战之前，先对其同盟发起挑战 (悲观地说，这假设攻击者总是能在诚实的一方之前进入交易)。同盟者也会故意放弃挑战，尽可能拖延时间（由于延迟规则，每个同盟者可能需要一到两个挑战期）。只有在所有 N-1 同盟者都牺牲了自己之后，被错误下注的质押者才会被要求与诚实的一方进行挑战，诚实的一方会获胜，最终淘汰错误下注的质押者。

这种攻击实现了大约 N 个挑战期的延迟，而攻击者付出了 N 份质押权益的代价。

诚实方针对这种攻击的代价，与诚实方的数量成线性关系，因为每个诚实方都需要在质押截止日期之前质押。

小结：

保证进程攻击成本与造成的延迟成线性关系诚实方的成本与诚实方的数量呈线性关系许可式验证

目前部署在 Arbitrum 上的协议 2 所面临的延迟攻击问题，是我们选择暂时将验证者角色限制在一组受许可的各方，而不是使验证完全无需许可的原因。实现无需许可验证的最后一步，需要最大限度地抵抗延迟攻击的协议版本。你可能已经猜到有这样一个协议版本，而我们目前正在努力实现它。

即将推出的协议 3

这篇文章已经很长了，所以我不会提供有关新协议的具体描述，那将是之后的文章将会介绍的东西。

简单说一下要点：Arbitrum 协议的下一个版本对断言和挑战的工作方式做了一些小的，但（我认为）优雅的改变，这样最坏情况下的延迟攻击只能造成一个挑战期的延迟（无论攻击者愿意失去多少质押权益）。

这是基于 Arbitrum 研究团队的一项技术突破，该技术突破使 「all-against-all」挑战变得可行且高效。这允许一个诚实的质押者有效地击败一大群发布恶意分支断言的攻击者。

新协议的规范已经完全确定，目前正在实施当中。当然，在彻底测试和审核代码之前，我们不会在主网上推出它。

在以后的文章中，我将深入探讨该新协议。