支付页面安全与 E-Skimming 防护-- ---浅谈 PCI DSS v4.0.1 要求 6.4.3 与 11.6.1 的实施

- 浅谈 PCI DSS v4.0.1 要求 6.4.3 与 11.6.1 的实施

作者：atsec 高磊

关键词：支付页面安全、E-Skimming、PCI DSS v4.0.1、第三方脚本、风险管理、持卡人数据、数据安全、第三方服务提供商、TPSP、内容安全、网页监控、恶意脚本攻击

atsec information security

Tel +86-10-53056681
Fax +86-10-53056678

www.atsec.com

目录

1 引言

随着电子商务的迅猛发展，支付页面安全已成为保护消费者支付卡数据的核心防线。然而，电子窃取（E-Skimming）攻击通过注入恶意脚本窃取支付信息的威胁日益加剧，尤其是在高度依赖动态脚本的现代电商环境中。

PCI DSS v4.x 引入的要求 6.4.3 和 11.6.1 旨在通过管理和监控支付页面脚本，防范此类攻击，为机构提供明确的合规路径。要求 6.4.3 聚焦于脚本的授权、完整性验证和清单管理，要求 11.6.1 则强调检测和响应未经授权的修改。

PCI 安全标准委员会（PCI SSC：Payment Card Industry Security Standards Council）于 2025 年 3 月发布的《支付页面安全与防止 E-Skimming 指导》为这些要求提供了详细的技术实施建议。atsec 作为 PCI GEAR（Global Executive Assessor Roundtable）成员之一，也积极参与了该指导的提出和编写工作。

atsec 顾问此前编写发布的文章《PCI DSS 针对恶意脚本防范的新要求及其方案探讨》也提及了这些要求的背景和通用解决方案。本文则进一步聚焦于技术实施细节，结合 PCI DSS v4.0.1 的最新要求和指导文件，探讨机构在控制措施层面的实施和部署建议。

2 当代环境下电商面临的威胁和风险

2.1 电商平台的安全隐患

当代电商平台依赖多种脚本（如 JavaScript、Web Assembly）实现动态交互功能，例如表单验证、支付处理和用户体验优化。然而，这种依赖性也带来了显著的安全隐患。支付页面脚本可能来源于第一方服务器（由支付机构或者商户直接控制）、第三方服务器（如服务提供商 TPSP：Third-Party Service Provider 提供），甚至第四方服务器或者其他机构提供（由第三方加载的额外脚本）。这些脚本在消费者浏览器中执行，若被篡改，可能在实体不知情的情况下泄露支付数据。

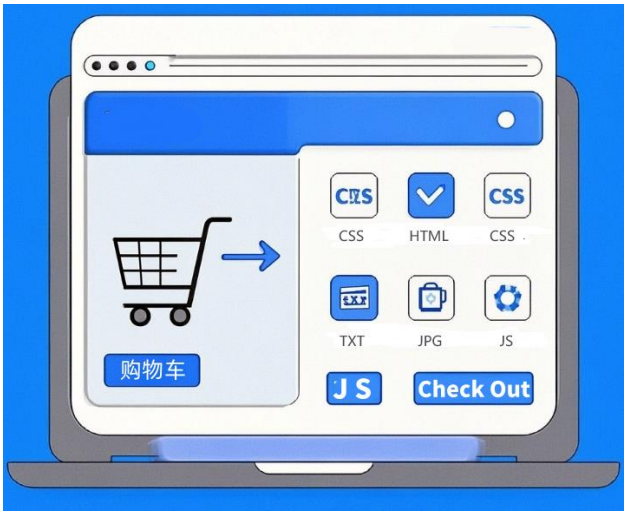

图 1：电子商务网页的组件

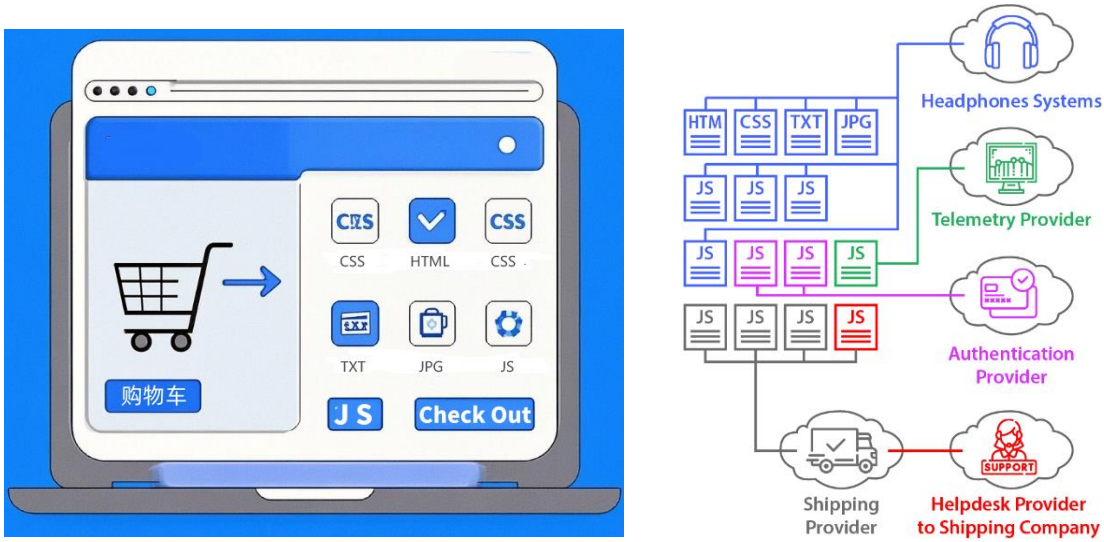

图 2：电子商务网页中的第三方组件（来自 PCI SSC）

在某些情况下，脚本能够与网站的敏感部分进行交互，如果这些脚本未获授权或未受监控，可能会无意中为攻击者创造可乘之机。例如，用于分析、营销、使用情况跟踪、标签管理、表单验证或支付处理的第三方脚本可能会被恶意行为者操纵，从而窃取支付数据。

特别是第三方脚本，因其不受商户直接控制，成为攻击者的主要目标。例如，一个广告脚本可能通过供应链攻击被注入恶意代码，进而窃取支付表单数据。这种复杂性和不可控性显著增加了电商平台的安全风险。

2.2 脚本如何被攻破

脚本可能通过以下方式被攻破：

- **供应链攻击：**许多电子商务网站依赖第三方服务提供商（TPSP）提供的脚本来实现各种功能。如果攻击者攻陷了这些第三方脚本，他们就可以在 TPSP 或商家没有立即察觉的情况下插入恶意代码。
- **脚本注入攻击：**攻击者将未经授权的脚本注入到支付页面中。这些被修改的脚本会捕获支付详细信息，并将其传输到攻击者控制的域名。

这些类型的攻击称为 Magecart 或 Formjacking。由于当代网站在很大程度上依赖脚本来实现其交互功能，攻击者可能会利用任何在消费者浏览器中运行的脚本来窃取支付数据。

通过主动管理和监控电子商务脚本，各实体可以降低这些攻击发生的可能性。虽然此类威胁将持续存在，但可以通过完善的控制措施（包括 PCI DSS 要求 6.4.3 和 11.6.1 中规定的措施）有效地应对这些威胁，这些控制措施实现了强大的脚本授权和完整性检查。专注于主动的脚本管理使机构能够在享受使用脚本带来的优势的同时，最大限度地减少其暴露于潜在安全问题的风险。

当 Web 浏览器遇到脚本时，它会解释并运行该脚本中的指令。这些指令可以读取、更改或删除其他页面内容，或者响应用户输入触发操作。例如，当消费者点击某个图标时，脚本可能会显示一个菜单，或者允许消费者在页面上拖动元素。在支付方面，脚本可以：

- 检测消费者何时在支付卡字段中输入数据。
- 识别消费者何时完成数据输入并移动到下一个字段。
- 验证输入的支付卡数据，并在验证失败时向消费者提供反馈。例如，脚本可以：
  - 更改支付卡字段的显示方式以指示问题（例如，将输入字段的边框更改为红色）。
  - 显示一条警告消息，提示数据输入不正确。
  - 禁用提交按钮，以防止提交不正确的数据。

这些功能可以立即发现输入错误，而无需等待授权失败。但是，如果支付页面缺乏适当的保护，恶意脚本也可能捕获支付账户数据并将其传输给攻击者。由于页面上运行的任何脚本都可能访问和窃取支付账户数据，因此 PCI DSS v4.0 在 2022 年引入了两项新的 PCI DSS 要求：6.4.3 和 11.6.1，以帮助预防和检测未经授权的脚本。

2.3 恶意脚本攻击方式解析

恶意脚本攻击通常以隐秘或欺骗的方式执行，尽管在不同类型的攻击中都会盗取支付账户数据，但消费

者在每种情况下的体验却有所不同，以下是两种常见类型：

- **静默攻击（Silent Skimming）**：恶意脚本在后台运行，消费者和商户均无察觉，在交易完成时窃取数据。由于其隐秘性，此类攻击可能长期未被发现。例如，攻击者可能通过键盘记录脚本捕获输入的卡号和 CVV2，并将其发送至外部服务器。
- **双重输入攻击（Double-Entry Skimming）**：攻击者在合法支付表单前插入虚假表单，要求消费者输入两次数据，第一次输入被窃取。此类攻击因用户体验异常（如重复输入提示），或者是商家在测试网站时发现了异常，相对容易被察觉。 例如，伪造的“验证码”页面可能诱导用户输入敏感信息。

这些攻击利用了脚本的动态执行能力，如读取 DOM（Document Object Model）元素、修改页面内容或发起未经授权的网络请求。

2.4 对行业的影响

脚本攻击可能导致严重后果，包括支付数据泄露、监管罚款和品牌声誉受损。主动管理和监控脚本可显著降低此类风险。例如，供应链攻击可能通过第三方脚本提供商影响多个商户，而直接脚本注入则利用网站本身的漏洞（如未修补的 XSS：Cross-Site Scripting）。

以下是笔者在编写本文时所获取的相关安全事件信息，谨供参考：

事件	发生年份	影响
GrimResource攻击（微软）	2024年	利用未修复的XSS漏洞执行恶意代码，绕过安全措施，威胁机构数据完整性及用户信任。
奥运会售票网站攻击	2021年	攻击者利用第三方脚本漏洞窃取用户支付数据，影响全球用户信任及平台运营稳定性。
英国航空（British Airways）事件	2018年	约有380,000张支付卡信息，包括姓名、地址、银行卡详情和CVV码被泄露，影响了约500,000名客户。
某旅游服务商脚本漏洞	2018年	第三方脚本漏洞导致30万用户信息泄露，引发百万美元罚款和客户流失。

表 1：近年来公开披露的重大脚本攻击事件

3 PCI DSS 要求 6.4.3 和 11.6.1 介绍

3.1 要求 6.4.3：脚本的授权、完整性验证、与清单管理

PCI DSS 要求 6.4.3 包含三个核心要素——授权、完整性以及包括技术或业务必要性说明的脚本清单，其目标在于从源头防范“电子窃取”（E-Skimming）等恶意攻击。根据 PCI DSS 的要求 6.1.1，机构应当编制并管理与第 6 章相关的安全策略和操作流程，包括为满足 6.4.3 而需要的各项措施。同样地，根据 PCI DSS 要求 6.1.2，机构应明确执行第 6 章各项活动的角色和职责，也包括那些与 6.4.3 相关的活动。所有支付页面脚本在消费者浏览器中加载和执行时，必须做到以下几点：

- **授权（Authorization）：必须采取某种方式来确认每个脚本都经过授权。**

  6.4.3 的第一个要素要求对每个脚本进行授权。标准本身没有具体规定授权方式，机构内如何进行或由谁提供授权，该过程是人工或自动完成。

  鉴于第三方脚本可能在机构毫不知情的情况下被篡改，标准 6.4.3 的指导栏中特别澄清，授权既可以在脚本新增或变更前执行，也可以在发现脚本发生变更后尽快进行审批或验证。

- **完整性（Integrity）：必须采取某种方式来确保每个脚本的完整性，避免其包含未经授权或恶意内容。**

  6.4.3 的第二个要素聚焦于脚本完整性，即确保脚本中不包括未经授权或恶意的内容。实现这一点通常需要在以下两个关键时刻进行检查：

  - 当新脚本被引入时。
  - 当现有脚本被更新时。

  PCI DSS 并不要求机构必须采用某种特定的技术来保证脚本完整性。在标准针对 6.4.3 和 11.6.1 的指导栏中，常见的示例方案包括内容安全策略（CSP：Content Security Policy）和子资源完整性（SRI：Sub Resource Integrity），但机构也可以选择其他能达成相同目标的替代方案，如专门的脚本管理工具等。

- **脚本清单及必要性说明（Inventory and Justification）：必须维护一个包含所有脚本的清单，并在其中说明每个脚本存在的业务或技术理由。**

  6.4.3 的最后一个要素强调在支付页面中维护一个详细的脚本清单，并对每个脚本的存在给出业务或技术层面的合理说明。

  只保留“已知且必需”的脚本，可以有效减少攻击者可利用的潜在入口。PCI DSS 并未规定必须采用何种形式来管理脚本清单：对于简单环境，一份电子表格可能就足够；而对于复杂环境，则可使用自动化工具或工作流系统来进行更全面的追踪与管理。

  为了满足 PCI DSS 要求 6.4.3 中关于维护脚本清单的要求，以下提供了一个详细的脚本清单模板。该模板旨在帮助机构记录和管理所有在支付页面中使用的脚本，确保每个脚本的存在都有明确的业务或技术理由，并定期进行审核和更新。

脚本名称	版本	目的	必要性理由	来源	最后审核	状态	备注
参考填写 ExampleScript.js	1	处理用户身份验证	用户登录功能所需	内部	2024/10/1	活跃	每 6 个月审核一次
参考 ThirdPartyLib.js	2.3	提供数据可视化	报告仪表板所需	第三方	2024/9/15	活跃	每季度检查更新
参考 ThirdPartyLib.js	1.2	处理支付交易	支付处理所必需	内部	2024/10/5	活跃	审核安全漏洞
参考 ThirdPartyLib.js	3.1	对旧系统的遗留支持	迁移完成前需要	内部	2024/8/20	不活跃	计划在 2025 年第一季度弃用

表 2：脚本清单管理模板（示例）

上述针对 PCI DSS 要求 6.4.3 中提供的这些措施共同确保支付页面脚本环境可控，防范 E-Skimming 攻击。

**另外，对于采用 3DS 解决方案的商户，由于在商户的尽职调查、入驻流程以及双方业务协议中已建立了内在的信任关系，因此 3DS 相关脚本无需符合 PCI DSS 要求 6.4.3。但凡用于除 3DS 功能之外的任何脚本，则必须遵守 PCI DSS 要求 6.4.3。*

3.2 要求 11.6.1：支付页面的变更和篡改检测机制

变更和篡改检测机制的目的是为了及时发现支付页面中的未经授权修改，从而规避潜在的安全漏洞，并确保机构能够通过监控 HTTP 头和页面内容的变化来评估影响。

- **告警功能：**检测到支付页面中安全影响性 HTTP 头或脚本发生未经授权的修改（包括存在被攻击迹象、内容变更、增加或删除）时，机制能够及时向相关人员发出告警。

  当检测到下面情况的时候需要做到及时告警：

  - **新增：**当检测到新的 HTTP 头或脚本被引入时（例如，出现第二个 CSP 指令或新增一个 JavaScript 文件）。
  - **变更：**当预期加载在支付页面的某个 HTTP 头或脚本发生了改动，或其行为与之前授权的状态不一致时。
  - **删除：**当某个关键的 HTTP 头（如 CSP 指令）或防御性脚本被移除时。

  这些要求旨在预防或尽量减少支付数据被窃取的风险。如果检测机制仅用于告警而不自动阻断变更，机构应制定快速响应计划，确保在收到告警后能够及时采取纠正措施，从而为机构和客户提供最佳保护。

- **评估功能：**该机制应配置为对接收到的 HTTP 头信息和支付页面内容进行评估，并与预定基线状态进行比对。

篡改检测机制必须同时对安全影响性 HTTP 头和支付页面的脚本内容进行评估。可以采用单一机制或多个机制组合的方式，确保整体满足要求。

- **检测频率：**机制的检测操作应至少每周进行一次，或者根据机构目标风险分析（TRA，参照 PCI DSS 要求 12.3.1 中规定的所有要素）定义一个可接受的检测频率。

  PCI DSS 要求 11.6.1 规定，该机制的功能至少每周执行一次。或者，机构可以依据目标风险分析（TRA，参见 PCI DSS 要求 12.3.1）确定一个符合自身风险承受能力的替代检测频率。无论机构选择每周检测还是其他频率，都需要注意，在检测间隔期间，恶意修改可能会在未被发现的情况下持续存在，因此建议在条件允许的情况下，采用更频繁甚至实时的监控方式。

PCI DSS 11.6.1 要求认识到未经授权的变更是可能发生的，因此该控制措施并非要求完全杜绝所有未经授权的变更，而是确保一旦发生此类变更，能够被及时发现并发出警报，以便迅速采取纠正措施。一般情况下，未经授权的变更往往会触发对要求 6.4.3 中所涉及的控制措施（授权、完整性、清单及必要性说明）的复查。

此外，机构还应明确制定执行 PCI DSS 要求 11（参见要求 11.1.2）的各项活动的角色和职责，包括满足要求 11.6.1 所需的相关工作。值得注意的是，要求 11.6.1 本身并未明确规定相应的政策和流程，因此本节特别引用了要求 11.1.1，用于指导相关政策和流程的文档编制。

4 常见支付页面场景及其职责划分

4.1 常见的支付场景

支付页面场景的不同直接影响 6.4.3 和 11.6.1 的实施责任。下面列出以下场景及职责划分：

商户托管支付表单：

当支付表单由商户托管时，所有支付数据输入字段均位于商户的服务器上。这些字段不位于支付处理方或第三方服务提供商（TPSP）的站点。支付数据可通过多种方式传输至支付处理方，但支付表单本身始终由商户环境加载并控制。下面表格描述了商户托管支付的常见架构：

商家托管支付架构	特征
商家发布支付	▪ 消费者的浏览器将支付数据返回到商家的网站。 ▪ 商家网站将支付数据提交给支付处理方/第三方支付服务提供商（TPSP）。 ▪ 所有支付数据都流经商家环境。
直接发布支付	▪ 消费者的浏览器直接将支付数据提交给支付处理方/TPSP。 ▪ 支付表单字段托管在商家网站上。 ▪ 来自消费者的支付数据不流经商家环境。

表 3：常见的商户托管支付架构

嵌入式支付表单（iframe）：

通过使用 iframe，可以将支付表单嵌入到商户网站（即父页面）中。该支付表单可以由 TPSP/支付处理方提供，也可以由商户直接管理，并可通过 iframe 标签或 JavaScript 加载。当使用脚本来创建 iframe 时，PCI DSS 要求 6.4.3 和 11.6.1 同样适用于这些脚本。

- 单一文档或实例（无 iframe 的支付页面）
- 跨域 iframe 中的文档
- 分别置于多个 iframe 中的文档

以下表格展示了三种常见的 iframe 使用场景，帮助理解支付表单在不同模式下的实现方式：

单一文档或实例	跨域 Iframe 中的文档	跨域 Iframe 中的文档

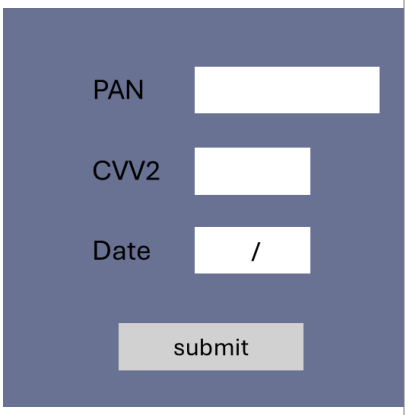	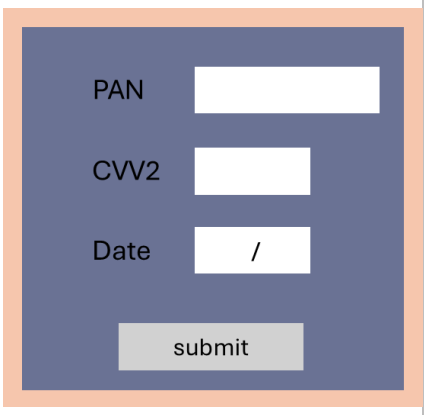	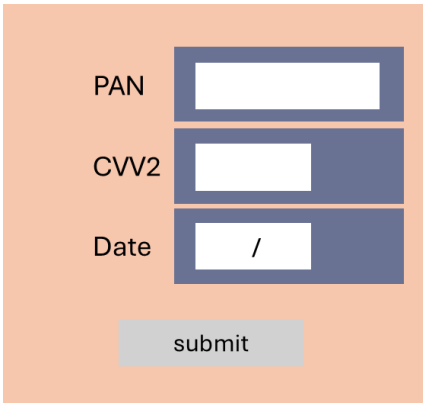
包含支付卡数据所有输入字段的单个网页。数据可以提交到实体服务器或第三方服务提供商。在此场景中没有 iframe。	包含所有支付字段的 HTML 页面，并在网页（父页面）的 iframe 中呈现。	每个字段都包含在一个单独的文档中，并且每个文档都在网页（父页面）的 iframe 中呈现。

表 4：iframe 示例

重定向机制：

重定向机制是指将消费者从商户网站转移到由支付处理方/TPSP 拥有的独立网站或托管支付页面。消费者根据商户网站的设置，跳转到新的 URL、新的浏览器窗口或弹出窗口。

重定向可通过服务器端配置（例如 HTTP 30x 重定向）、HTML 标签或 JavaScript 来实现。当重定向机制中涉及到脚本时，PCI DSS 要求 6.4.3 和 11.6.1 将适用于这些脚本。

完全外包网站：

在完全外包的支付场景中，所有支付数据的采集和处理均由支付处理方/TPSP 负责。商户环境不再处理或托管任何支付表单或处理支付数据的脚本。常见的示例包括：商户向消费者提供支付处理方/TPSP 的“支付链接”，消费者通过该链接直接跳转到支付处理方/TPSP 的网站完成支付。

4.2 要求 6.4.3 和 11.6.1 适用性和职责

PCI DSS 要求 6.4.3 与 11.6.1 针对不同的支付解决方案实施方式，对各机构的适用性有所不同。利用“常见支付页面场景”中描述的各类情形，本节重点阐明了这两项要求在不同电商场景下的适用情况，并明确了商户与 TPSP 在支付页面脚本安全管理方面的职责划分。

以下表格总结了各类支付页面场景中商户和 TPSP 的职责范围，帮助机构明确各自的安全管理义务：

支付页面场景	商户职责	支付服务提供商（TPSP）职责
商户发布的支付（Merchant posted payment）	商户网页上的任何脚本。	TPSP为其提供的服务所包含的任何脚本。
直接提交支付（Direct post payment）	商户网页上的任何脚本。	TPSP为其提供的服务所包含的任何脚本。
嵌入式支付表单（内嵌框架）（Embedded payment forms (iframes)）	商户网页上包含内嵌框架的脚本（即非支付页面上的脚本）。	TPSP提供的内嵌框架内的脚本（即支付页面上的脚本）。
重定向机制（Redirection mechanisms）	商户网页上包含重定向机制的脚本（即非支付页面上的脚本）。	TPSP为其提供的服务所包含的任何脚本。
完全外包的商户网站（Fully outsourced merchant website）	与要求6.4.3和11.6.1无关的内容。	TPSP为其提供的服务所包含的任何脚本。

表 5：PCI DSS 要求 6.4.3 和 11.6.1 - 适用性和职责

PCI DSS 要求 6.4.3 与 11.6.1 均包含在以下自评问卷（SAQ：Self-Assessment Questionnaire）中：SAQ A-EP、商户版本的 SAQ D 以及服务提供商版本的 SAQ D，同时这两项要求也体现在 PCI DSS 合规报告 ROC（Report on Compliance）模板中。

需要注意的是，SAQ A 不涵盖要求 6.4.3 或 1.6.1，但其仍包含针对电商商户的“资格标准”，即“商户已确认其网站不易受到可能影响其电商系统的脚本攻击”。关于 SAQ A 商户如何证明其满足此资格标准，请参阅《小商户最佳实践》中的相关说明。

同时，即使商户符合完成自评问卷的条件，适用哪种 SAQ 也会因电商环境的具体实现方式、商户在电商环境实施与管理中的参与程度、以及为哪些服务采用了 TPSP 而有所不同。

最终，机构是采用 SAQ 报告 PCI DSS 评估结果，还是必须使用 QSA 提交 ROC 报告评估结果，由管理合规计划的相关机构（如收单行、支付品牌或其他相关机构）决定。各机构应与这些机构密切合作，明确自身在合规验证和报告中的责任与义务。

4.3 PCI DSS 在电子商务环境中的适用范围

根据 PCI DSS v4.0.1 第 4 章“PCI DSS 要求的适用范围”规定，其范围包括：

- **持卡人数据环境 CDE（Cardholder Data Environment）包括以下部分：**
  - 存储、处理或传输持卡人数据 CHD 及/或敏感认证数据 SAD（Sensitive Authentication Data）的系统组件、人员和流程；

- 虽然不直接存储、处理或传输 CHD/SAD，但与存储、处理或传输 CHD/SAD 的系统组件之间具有无限制连接的其他系统组件。

- **可能影响持卡人数据及/或敏感认证数据安全性的系统组件、人员和流程。**

这其中也包括电子商务支付页面以及嵌入 iframe（无论是第三方 iframe 还是由商户管理的 iframe）的父页面。

许多电商商户试图通过不在商户网页上存储、处理或传输任何支付账户数据来缩小 PCI DSS 的适用范围，而是依赖第三方服务提供商（TPSP），通过嵌入 TPSP 的支付页面或通过重定向至 TPSP 页面来实现支付功能。需要注意的是，PCI DSS 要求 6.4.3 和 11.6.1 不适用于通过 HTTP 30x 重定向、meta 标签或 JavaScript 重定向到 TPSP 页面的商户网站。

尽管将 TPSP 的支付页面嵌入商户网页可能会减少适用于这些网页的 PCI DSS 要求，但嵌入第三方支付页面或表单并不能完全将商户嵌入页面（父页面）排除在评估范围之外，也不意味着要求 6.4.3 和 11.6.1 不适用。关于 PCI DSS 要求 6.4.3 和 11.6.1 在不同自评问卷以及合规报告模板中的适用性和职责划分，请参见“要求 6.4.3 和 11.6.1——适用性与职责”。

多页应用（Multi-page Applications）

传统的电子商务网站通常采用多页应用模式：用户在流程中每个步骤都导航到一个新的 URL，每个新页面均为“全新”加载。此方式会重建浏览器环境，从而有效清除前一页面加载的脚本。在这种多页设置下，通常只有嵌入了 TPSP 支付页面的父页面需要纳入商户在要求 6.4.3 和 11.6.1 中的评估范围。

单页应用（SPA：Single-page Applications）

单页应用近年来越来越受欢迎，并已占据现代电商网站的较大比例。在 SPA 中，当用户进行页面导航时，浏览器不会完全重新加载页面，而是通过 JavaScript 动态添加或移除内容。由此，用户会话期间加载的所有脚本一直驻留在内存中并持续运行。

在这种情形下，从浏览器的角度看，整个 SPA 就像是一个连续的“页面”，包括其中嵌入的支付表单。由于所有脚本都处于同一运行环境中，因此要求 6.4.3 和 11.6.1 将适用于单页应用中的所有页面（或“视图”），这也可能影响到嵌入式支付表单的安全管理。

5 第三方脚本的风险与管理

5.1 第三方脚本管理方法

第三方脚本的管理具有独特挑战，因为这些脚本往往不受使用它们的机构（例如商户或服务提供商）的直接控制。然而，历史上这些脚本与窃取支付数据的攻击（Skimming attacks）密切相关，因此，机构必须充分了解并有效管理这些脚本。

如果某个脚本供应商不被视为 TPSP，机构就必须建立相应流程来管理这些第三方脚本，包括明确这些脚本的来源、将它们纳入机构的管理范围，并确保按照机构既定的流程来满足 PCI DSS 要求 6.4.3 和 11.6.1。

机构需要建立一种机制，该机制既能评估 HTTP 头信息，也能对脚本内容进行检测。具体来说，该机制应将当前的 HTTP 头和脚本内容与之前已知的版本进行比对，从而发现异常变化。此机制可以仅仅发出告警而不自动阻断变更，从而为人工迅速介入提供灵活性；或者，如果机制不能实时阻断变更，机构则应制定快速响应方案，及时处理告警信息。

根据机构的技术能力和资源状况，管理第三方脚本可以采用不同的方法，常见方法包括：

- **内部自研工具：**利用定制化解决方案来管理第三方脚本，涵盖脚本白名单、完整性验证（例如使用哈希比对）以及其他安全控制措施。
- **商业与开源解决方案：**借助现有技术实现第三方脚本的监控、控制和安全措施的执行。
- **混合方法：**结合内部工具和外部扫描工具或服务，共同检测和响应异常情况。

这些方法各有优势，机构可根据实际情况选择或组合应用，以实现对第三方脚本的全面有效管理。

**另外：如果第三方脚本提供商仅提供与支付处理无关、且这些脚本不会影响持卡人数据及/或敏感认证数据安全性的脚本，则该提供商不视为 PCI DSS 要求 12.8 和 12.9 中的第三方服务提供商（TPSP）。*

5.2 第三方脚本风险最小化的最佳实践

通过安全控制措施与技术手段，可以构建一个全面的机制来满足 PCI DSS 要求 6.4.3 和 11.6.1。虽然 PCI DSS 主要关注对脚本的管理与监控，以防止未经授权的变更，但本节所述的控制措施可多种方式部署，以应对电子窃取（E-Skimming）及其他针对电子商务网站的威胁。

降低脚本数量

如前所述，要求 6.4.3 和 11.6.1 关注支付页面以及嵌入或影响支付页面的非支付页面（或父页面）。因此，缩小评估范围的最简单方法之一是仅保留严格必要的脚本。其他补充措施包括：

- **将脚本移至独立的 iframe：**通过使用跨域或沙箱化的 iframe，开发者可以将脚本限制在一个与电子商务支付页面或父页面相互隔离的受控环境中。经过适当沙箱化的 iframe 可以限制脚本的功能，防止其与主页面上的敏感支付元素发生不必要的交互。

- **限制脚本来源：**限制脚本加载的 URL 是降低风险的另一有效方法。常见做法是采用内容安全策略（CSP），利用如 script-src 指令来定义允许执行 JavaScript 的可信来源。或者，也可以采用基于代理的解决方案或基于代理的网页监控工具，检测、拦截或管理来自未经授权来源的脚本。
- **了解并建立脚本行为基线：**在隔离环境中测试脚本，观察其标准行为及输出，进而限制或密切监控非必要功能。例如，可以使用沙箱环境运行脚本，记录其网络请求和 DOM 操作，建立行为基线。若脚本行为偏离预期（如发起未经授权的请求），则触发告警或限制其执行。
- **技术安全评估：**定期或持续对第三方脚本进行安全评估，旨在识别潜在漏洞和异常情况。这些评估可以包括静态代码审查、行为分析以及运行时监控等手段，以确保脚本始终符合安全要求。

6 要求 6.4.3 和 11.6.1 的控制措施与技术手段

本节探讨一些控制措施与技术手段，它们可以组合使用以满足要求 6.4.3 和 11.6.1 的各个要素（授权、清单、完整性和告警）。这些措施和技术还可用于检测安全影响性 HTTP 头以及支付页面与非支付页面脚本中的变更、新增、删除或可疑行为（例如被攻击的迹象）。

以下表格总结了有助于满足 PCI DSS 要求 6.4.3 和 11.6.1 的控制措施与技术手段，列出了针对脚本授权、完整性验证、变更检测等方面的具体方法，以及与内容：

要求及要素	用于满足要素的控制措施	与控制措施配合使用的技术
PCI DSS要求6.4.3		
授权（Authorization）	网页监控、基于代理的授权方法或其他授权方法。	—
完整性（Integrity）	内容安全策略（CSP）、子资源完整性（SRI）、网页监控、基于代理的完整性方法或其他完整性方法。	文件哈希、通过URL限制脚本来源、行为监控、静态分析、防篡改脚本或其他技术。
清单（Inventory）	内容安全策略（CSP）、网页监控、基于代理的清单方法（例如列表）。	—
PCI DSS要求11.6.1		
告警（Alerting）	内容安全策略（CSP）、网页监控、基于代理的告警方法或其他告警方法。	—
安全相关的HTTP头部（Security-impacting headers）	网页监控、基于代理的方法或其他可检测变更的方法。	—
脚本内容变更（Script content changes）	内容安全策略（CSP）、网页监控、基于代理的方法或其他检测变更的方法。	文件哈希、行为监控、静态分析、防篡改脚本或其他技术。
脚本内容的入侵指标（Script content indicators of compromise）	内容安全策略（CSP）、网页监控、基于代理的方法或其他检测入侵指标的方法。	通过URL限制脚本来源、随机数（Nonces）、行为监控、静态分析、防篡改脚本或其他技术。
执行频率（Frequency）	内容安全策略（CSP）、子资源完整性（SRI）、网页监控、基于代理的调度方法或其他调度方法。	—

表 6：有助于满足要求 6.4.3 和 11.6.1 的控制措施与技术手段

表 6 所述控制措施的详细说明

本节的下面介绍了一些安全控制措施，它们可以检测并在某些情况下预防未经授权的脚本活动，同时向相关利益方发出告警。这些控制措施通过保护支付数据环境来支持 PCI DSS 要求 6.4.3 和 11.6.1 的实施。

6.1 内容安全策略（CSP）

CSP 是浏览器自带的功能，允许网页应用设置加载和执行内容的策略。虽然 CSP 在脚本授权以及一定程度上的脚本完整性方面十分有效，但机构仍需配合其他流程，才能全面满足 PCI DSS 要求 6.4.3 和 11.6.1 的各项要求。

- **实施方式：**CSP 指令可以通过 HTTP 响应头（例如 Content-Security-Policy）或 meta 标签传递。通过 CSP 可以限制脚本加载的来源（script-src 指令）以及规定允许加载 iframe 的来源（frame-src 指令）。同时，通过基于哈希值或 nonce（一次性令牌）的方式，可以进一步保证页面上允许执行的脚本的完整性。
- **报告功能：**CSP 可结合 HTTP Reporting API（例如 report-to、report-uri）使用，以捕捉和报告未经授权的脚本被拦截或违反策略的事件。

优点：

- 基于浏览器原生功能，广泛支持。
- 有助于构建脚本清单、进行完整性检查，并能部分检测到未经授权的新增或修改。
- 提供了一套违规报告的框架。

局限性：

机构需要额外实施授权、告警、HTTP 头变更追踪等流程；例如，仅靠 CSP 无法生成未经授权脚本的清单或对安全相关 HTTP 头的变更发出告警。

- CSP 不能维护正常活动的基线，其静态配置无法跨会话追踪历史或预期状态。
- 在动态环境中，维护一套健全的 CSP（尤其是包含哈希值的配置）具有一定难度。
- 除非同时与不允许的域进行通信，否则 CSP 无法检测到安全影响性 HTTP 头的删除，也无法确认恶意脚本是否改变了其内部功能。

CSP 配置示例：

Content-Security-Policy: script-src 'self' https://trustedscripts.example.com;

 # 限制脚本来源为本地和可信域名，防止未授权脚本执行。

6.2 子资源完整性（SRI）

SRI 通过将 HTML 标签中的哈希值与浏览器加载的资源进行比对，帮助确认加载的资源（如脚本）未被篡改。如果比对结果不匹配，则该资源将被阻止执行。

优点：

- 对静态脚本配置简单直接。
- 有助于确保第一方脚本以及某些第三方脚本（前提是更新可预测且不频繁）的完整性。

局限性：

- 对于频繁变化或不可预测的动态第三方脚本来说，SRI 并不实用。
- SRI 检测失败时不会发出告警，缺乏原生的通知机制。
- 不支持违规报告。

SRI 配置示例：

```
<script src="https://example.com/script.js" integrity="sha384-oqVuAfXRKap7fdgcCY5uykM6+R9GqQ8K/uxy9rx7HNQlGYl1kPzQho1wx4JwY8wC"></script>

# 通过完整性校验，确保脚本未被篡改。
```

6.3 网页监控

网页监控指的是对支付页面在消费者浏览器（或合成环境）中呈现时进行检测，以发现恶意或异常的脚本活动。这种监控主要针对客户端运行的网页应用，观察各个脚本与页面组件及其他浏览器资源（如 cookies、local storage 等）的交互情况。常见的监控模式有两种：

- 基于代理的监控：在页面中注入监控脚本，跟踪 DOM 变化、资源请求及行为模式。
- 无代理监控：采用例如无头浏览器等方式，定期模拟用户操作，观察加载的脚本、HTTP 头以及行为，而无需在真实用户会话中注入额外脚本。

优点：

- 能够实时或近实时地洞察实际脚本行为。
- 可检测到意外的表单覆盖攻击、可疑的数据外泄等情况。

局限性：

- 基于代理的解决方案需要集成到每个页面中，不当的实现可能影响页面性能或与其他脚本冲突。
- 无代理的方案可能在处理验证码、登录或状态驱动流程时遇到困难，并且依赖于预定的检查频率而非持续监控。

6.4 基于代理的解决方案

基于代理的解决方案通过在反向代理或内容分发网络（CDN：Content Delivery Network）边缘拦截流量，从而在 HTML 与脚本到达消费者浏览器之前对其进行修改或分析。这类方案可以同时监控静态与动态的脚本引用，强制执行 CSP 等策略，并能实时拦截未经授权的内容。

优点：

- 部署时对应用修改要求较低，可作为无代理方案使用。
- 通过监控脚本加载情况，可提供完整的脚本清单。

局限性：

- 配置可能较为复杂，若未优化好可能引入延迟。
- 效果取决于检测算法，可能会产生误报。

6.5 表 6 中所涵盖的技术手段——第三方脚本风险最小化的最佳实践

上面的表 6 中所列出的几种技术手段，这些手段可与上述安全控制措施配合使用，帮助机构满足 PCI DSS 要求 6.4.3 和 11.6.1。常见的技术手段包括：

文件哈希：计算脚本的加密哈希值，并验证脚本内容未发生变化。对于静态脚本适用，但对于频繁变动或针对每个用户定制的脚本则不适用。

URL 来源限制：利用 CSP 及其他控制措施对特定域名实施白名单策略，若检测到可疑的域名使用则发出告警。

Nonce（一次性令牌）：CSP 可通过 nonce 授权脚本，即在脚本标签和 CSP 头中插入唯一令牌，若令牌不匹配则阻止脚本执行。nonce 能检测未经授权的脚本，但无法监控授权脚本内部的变更；对于第三方托管的脚本，使用 nonce 则较为困难。

将脚本清单集成到开发流程中：机构可通过自动化持续集成/持续交付（CI/CD：continuous integration/continuous delivery）流水线来维护脚本清单，自动跟踪和更新授权脚本列表。此过程可以包括对第一方和第三方脚本的验证检查，确保仅部署经过批准的脚本，并在脚本上线前进行安全评估，检测未经授权的变更。

手动流程：部分要求（例如维护脚本清单和验证脚本授权）可通过人工审查、签字确认或定期网站内容分析来完成，通常作为自动化控制的补充。

行为监控：不仅关注脚本的静态签名，还监控脚本在实际运行中的行为，例如是否捕捉键盘输入、修改或访问支付字段、或向未知 URL 发送数据；当脚本行为偏离授权的正常模式时，系统可触发告警或直接阻断。

静态分析脚本监控：定期对从网站或代理处收集的脚本进行扫描，检测已知的窃取或恶意代码模式。静态分析虽然适用，但可能会产生误报或漏检经过复杂混淆处理的代码。

防篡改脚本：利用编译工具对脚本进行转换或嵌入监测机制，从而检测或预防恶意修改。不论是在运行前或运行期间，一旦检测到篡改，脚本可以选择拒绝执行或触发告警。

以上这些控制措施与技术手段，可以协同作用，帮助机构更好地满足 PCI DSS 要求 6.4.3 和 11.6.1 的安全控制目标，确保支付数据环境的安全性。

6.6 其他技术手段

除了 CSP 和 SRI 等技术手段外，还可以引入其他防护技术来增强支付页面的安全性，本节重点探讨一下 WAF 的应用。

WAF 在支付安全中的作用

WAF（Web Application Firewall）是一种保护 Web 应用程序安全的设备或服务，部署在应用程序与用户之间，通过实时分析 HTTP 流量并阻断潜在威胁，为支付页面提供安全屏障。它主要防止攻击者注入恶意脚本，保护用户支付数据（如信用卡信息）不被窃取，同时支持合规性要求（如 PCI DSS）。

核心功能与应用

- 实时防护：检测并阻断异常 HTTP 请求，如包含恶意 JavaScript 的攻击，防止其到达应用程序。
- 灵活配置：支持自定义规则，例如拦截可疑 IP 或特定脚本流量，并通过虚拟补丁临时修复漏洞。
- 应用场景：拦截脚本注入、保护敏感数据传输、生成合规性日志，帮助满足 PCI DSS 6.4.3 和 11.6.1 要求。

配置与优缺点

- 配置要点：设置规则拦截异常请求、定期更新威胁规则、启用实时监控和告警。
- 优势：提供实时保护、配置灵活、部署简便。
- 局限性：可能误报或引入延迟，需结合 CSP、SRI 等技术优化防护效果。

7 TPSP 在此领域的技术帮助

提供电商或支付服务的第三方服务提供商（TPSP）可以通过以下方式协助商户满足要求 6.4.3 和 11.6.1：

- **托管支付页面：**将支付页面内所有代码的安全控制责任转移给 TPSP。
- **提供安全 SDK：**在商户实现中直接嵌入防护措施，并利用抗篡改脚本提供监控或拦截功能。
- **提供监控服务：**为商户集成的支付流程跟踪电商窃取（E-Skimming）指标，及时发现异常行为。
- **推荐最佳实践：**借助技术专长指导商户安全嵌入支付表单。

商户应确认 TPSP 的哪些服务已在其 PCI DSS 评估范围内，这通常应在 TPSP 的 PCI 合规性声明（AOC）中有所体现。

TPSP 需满足 PCI DSS 要求 12.9.1，即向客户提供书面协议，并明确承认 TPSP 对账户数据安全负有责任；同时也需满足要求 12.9.2，向客户提供有关其 PCI DSS 合规状态的信息，说明哪些 PCI DSS 要求由 TPSP 负责，哪些由客户负责，以及双方共同承担责任的要求。这些责任分工应以清晰的格式呈现，并说明各方如何或是否需要针对要求 6.4.3 和 11.6.1 采取相应措施。

需要注意的是，提供电商或支付服务的 TPSP 无法完全解决商户网站上的窃取风险，因为他们并不完全控制商户网站。商户应积极实施完善的机制，评估和管控允许在商户网页上执行的脚本。

以下简要提及一下针对不同规模商户在与 TPSP 合作时应考虑的最佳实践：

- 对 TPSP 进行评估，确保其具备足够的技术和控制措施，保障其提供的支付处理服务的安全。
- 商户应与 TPSP 合作，共同确定如何管理和启动 TPSP 的支付页面，从而缩小 PCI DSS 电商脚本要求的适用范围。
- 同时，商户应与 TPSP 沟通，获取关于如何安全实施其解决方案的指导建议。

针对小商户的最佳实践

- 选择提供嵌入式支付页面或表单（例如一个或多个 iframe）的 TPSP 或支付处理方，并确认按照其说明实施后，其解决方案具备防范脚本攻击的技术手段。
- 另外，小型商户也可以选择自行实施 PCI DSS 要求 6.4.3 和 11.6.1 中详细描述的防范措施，以保护其网页不被针对账户数据的恶意脚本攻击；这些措施可以由商户自行部署，也可由第三方协助完成。

针对中大型商户的最佳实践

- 考察所使用的脚本管理技术，明确如何设计网页以最大程度降低基于脚本的支付攻击风险。
- 实施机制对网页上执行的脚本进行评估和控制，确保所有脚本符合安全要求，并及时发现异常行为。

通过以上措施，商户在与第三方服务提供商合作时能够更好地满足 PCI DSS 要求，提升整体支付数据环境的安全性。

8 结论

PCI DSS v4.0.1 的要求 6.4.3 和 11.6.1 为电商支付安全提供了系统性的框架。通过授权和验证脚本、维护动态清单以及监控和响应未经授权的修改，机构可以有效地防范电子窃取（E-Skimming）等威胁。指导文件的技术建议为具体实施提供了清晰的路径，而针对不同支付场景的职责划分和最佳实践则确保了措施的实用性和灵活性。对于所有处理支付卡数据的商户而言，理解并有效实施这些要求是至关重要的，不仅能够保护持卡人的敏感信息，避免数据泄露造成的经济损失和声誉损害，也是满足 PCI DSS 合规要求的必要步骤。

atsec 期待结合多年来在 PCI DSS 合规方法论方面的积累，为不同机构提供全面的支持，确保支付页面以及其他关键业务流程的安全性，并提供产业的安全合规。

A 参考资料

(i) Payment Card Industry Data Security Standard, v4.0.1, June 2024

(ii) Payment Page Security and Preventing E-Skimming - Guidance for PCI DSS Requirements 6.4.3 and 11.6.1 , March 2025

(iii) Mozilla Developer Network. (n.d.). Content Security Policy (CSP)

(iv) PCI SSC 的词汇表 https://www.pcisecuritystandards.org/glossary

(v) Third-Party Security Assurance, v1.1, March 2016

(vi) www.atsec.com