

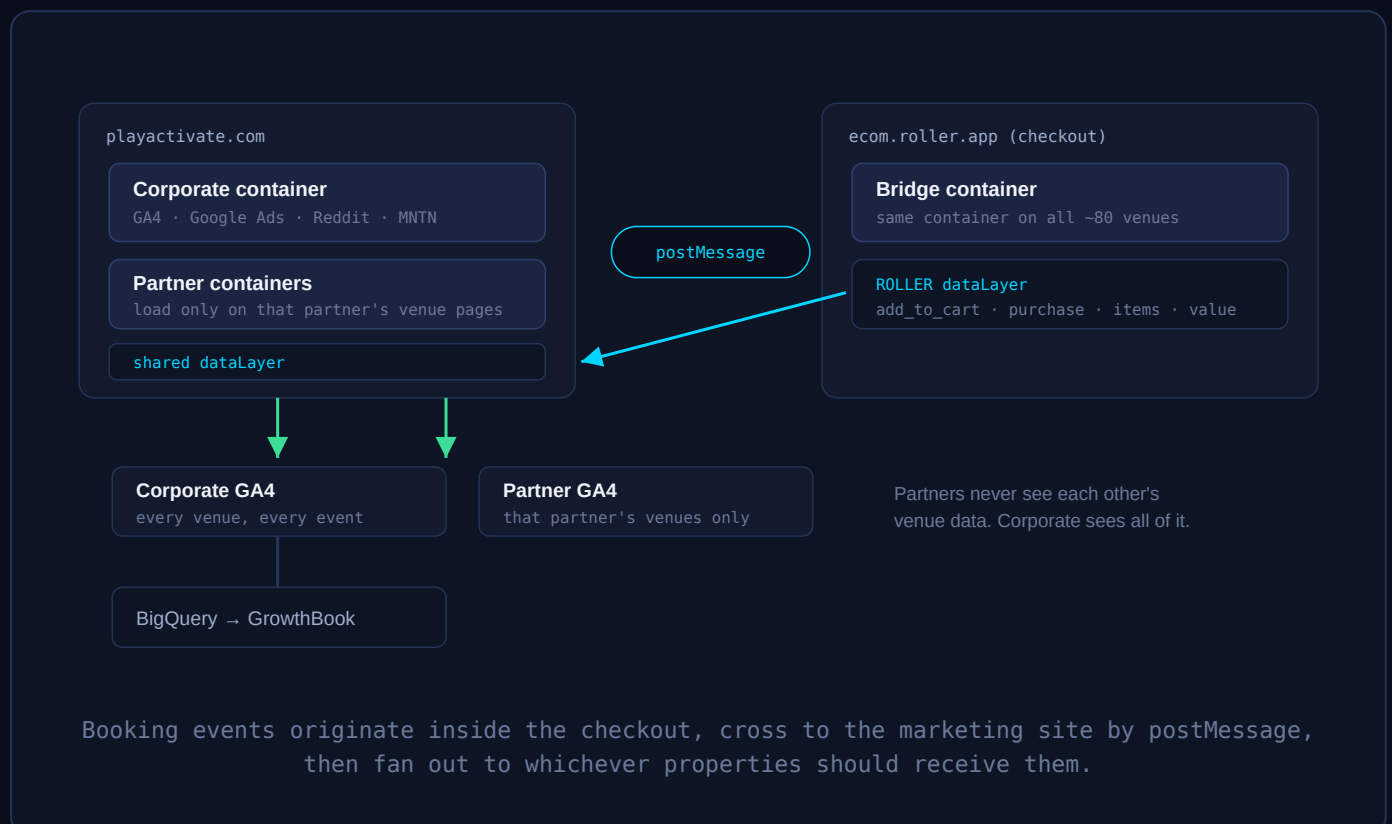
● Verified end-to-end · September 2026

Every booking, tracked once, attributed to the right venue.

Activate's checkout runs on ROLLER, on a different domain from the marketing site. This document explains how booking data crosses that boundary, why it's built this way, and what changes at launch.

How it fits together

Three pieces of tracking run on every booking. The marketing site carries one container, the ROLLER checkout carries another, and a relay passes events between them so everything lands in one place.



The important detail is that **nothing sends analytics from inside the checkout directly**. Events are relayed out first and sent from the marketing site. The next section explains why that matters.

Why it's built this way

ROLLER offers a built-in analytics integration: you paste a GA4 Measurement ID into each venue's settings and it sends events for you. It's the obvious choice, and we tested it against real bookings.

47%

of real transactions captured by ROLLER's built-in GA4 integration

85%+

captured by the relay approach, measured against the same reports

4 days

of transactions compared across two venues to confirm the gap

Both figures were checked against ROLLER's own transaction reports, not against each other. On one full day at Orlando, ROLLER recorded 12 online bookings and the built-in integration reported 6. Over two days at Burlington, ROLLER recorded 146 bookings and the relay reported 140.

What causes the gap

The checkout runs on `ecom.roller.app`, a different domain from the marketing site. Analytics running inside it is in a third-party context, where browsers restrict the cookies it needs to identify a visitor. ROLLER's own documentation notes that the overlay checkout may not work with GA4 tracking on Safari.

Relaying events out to the marketing site sidesteps this. The event fires from `playactivate.com`, where those cookies are first-party and work normally. `postMessage` is the standard browser mechanism for this, not a workaround.

This also explains a second problem the relay solves. When analytics runs inside the checkout, the ad click that brought the visitor is invisible to it, so a booking can't be traced back to the campaign that produced it. Firing from the marketing site keeps that connection intact.

The three containers

Each one has a single job and a single place it runs. Nothing overlaps.

CORP

Corporate container

Runs on every page of the marketing site. Holds corporate's GA4, corporate's ad platforms, consent management, and the listener that receives events from the checkout.

Runs on: every page · Owner: Activate corporate

BRDG

Bridge container

Runs inside the ROLLER checkout. Two tags only. One relays booking events out to the marketing site. The other reads guest details from the checkout form and hashes them for Enhanced Conversions matching. The same container ID goes in every venue, so adding a venue is one paste.

Runs on: the checkout, all venues · Owner: Activate corporate

PTNR

Partner containers

One per partner, loaded only on that partner's venue pages. Holds the partner's own GA4 and their own ad pixels. Partners control their container entirely and cannot affect corporate's measurement or another partner's.

Runs on: that partner's venue pages · Owner: the partner

Why partner containers don't load on shared pages

Partner containers fire only on that partner's venue pages, never on the homepage or other network-wide pages. If they loaded everywhere, a partner in one market would build remarketing audiences full of people shopping a different market, and every shared page would carry twenty containers' worth of script.

Partners still see where their traffic came from. A visitor who arrives through a corporate campaign and then books a venue keeps that original source in the partner's own reporting.

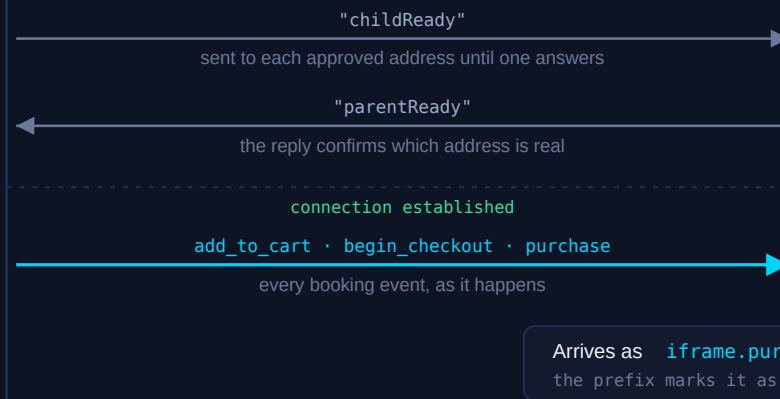
The checkout bridge

Browsers keep a page and an embedded frame from a different domain isolated from each other. `postMessage` is the one sanctioned channel between them, and it is what carries booking data out of the checkout. It is a long-established web standard, supported across all browsers since 2015, and its documented purpose is exactly this case: communication between a page and an iframe embedded within it.

More information: [MDN Web Docs, Window: postMessage\(\)](#)

Checkout (inside the frame)

Marketing site (parent page)



The two sides confirm each other before any data moves. Events are renamed with an `iframe.` prefix so they cannot be confused with events from the marketing site itself.

Approved addresses

The relay only sends to addresses on an approved list, and only accepts replies from that same list. This is what keeps booking data from leaking to a page that happens to embed the checkout.

<code>playactivate.com</code>	the live site
<code>www.playactivate.com</code>	the live site with www
<code>staging...amplifyapp.com</code>	the staging build, for testing before launch
<code>fun.playactivate.com</code>	reserved for the launch A/B test

Adding a new address

Any new domain that embeds the checkout has to be added to this list in both the relay and the listener, or booking events silently stop arriving from it. Nothing errors. The data simply stops appearing. This is the single most likely cause if bookings stop showing up after a domain change.

What we track

Two sources feed the same reporting. The site describes where a visitor is and what they are doing; ROLLER describes what they are buying.

From the site

Event	What it means
page_view	Any page. Carries the venue, page type and partner so every later event can be traced to a location.
click_booknow	Someone opened the booking flow. The top of the funnel. Kept under its existing name for continuity with historic reporting.
group_inquiry_submitted	A group enquiry form, tagged with the type: birthday, school or corporate.
newsletter_signup	Email signup.

From the checkout

Event	What it means
view_item_list	The product list loaded.
view_item	A specific experience was opened.
add_to_cart	Tickets added. Some products go straight to checkout without this step.
checkout_button_clicked	The true start of checkout. Cannot fire with an empty cart. Clicking it takes the guest straight into guest details.
checkout_progress	Movement through checkout steps, numbered.
purchase	Booking completed. Carries transaction ID, items, value, tax and any coupon.

Important: the checkout step was measuring the wrong moment

`begin_checkout` fires the instant the booking panel opens. Nothing is in the cart yet, no product has been chosen, and its value is always zero. Anyone who clicks Book Now and immediately leaves still produces one.

Until now, the Initiate Checkout conversion sent to Google Ads, Meta and TikTok was firing on that event. So all three platforms were optimising toward panel opens rather than real checkout starts, and every reported checkout figure includes visitors who never selected anything.

`checkout_button_clicked` is the accurate signal. It cannot fire with an empty cart, it carries the real cart value, and clicking it moves the guest directly into entering their details. That is a genuine checkout start.

Going forward, Initiate Checkout is reported on `checkout_button_clicked` across all platforms. `begin_checkout` is kept as a separate, lower-funnel-entry signal so the two can be told apart.

What to expect after the change

Checkout counts will drop noticeably in Google Ads, Meta, TikTok and GA4. That is the correction, not a loss. The previous figure counted panel opens; the new one counts guests who selected a product and moved to checkout.

Historic checkout numbers are not comparable to numbers after the change. Anything spanning the switch should be read as two separate periods.

Why funnel reports have not worked

A standard ecommerce funnel assumes events arrive in a fixed order. Browse a list, open a product, add it to the cart, start checkout, buy. Reporting tools are built on that assumption, and GA4's funnel exploration in particular treats each step as something that must happen after the one before it.

ROLLER does not emit events in that order, and there is a reasonable explanation for it. ROLLER calls its booking panel a **slide-out checkout**. By that definition, opening the panel *is* the start of checkout, so `begin_checkout` fires the moment it slides open.

That is internally consistent for ROLLER. It is also the reason a conventional funnel breaks.

Standard ecommerce	What ROLLER emits	What Activate now reports
1 session_start 2 page_view 3 view_item_list 4 view_item 5 add_to_cart 6 begin_checkout 7 purchase Checkout begins after a product is chosen.	1 view_item_list 2 begin_checkout panel opens, cart empty 3 view_item 4 add_to_cart 5 view_cart 6 checkout_button_clicked 7 purchase Checkout appears to begin before anything is browsed. A funnel built on this order reads near-zero completion.	1 session_start 2 page_view 3 click_booknow 4 begin_checkout panel opened, not checkout 5 view_item_list 6 view_item 7 add_to_cart 8 Initiate Checkout fires on checkout_button_clicked 9 purchase Order now reflects real intent.

The middle column is why funnel exploration returned unusable numbers. The right column is the corrected sequence.

What changed

The conversion named **Initiate Checkout** now fires on `checkout_button_clicked` rather than on the panel opening. That is the moment a guest has selected a product, has a populated cart, and is moving into entering their details. Roughly sixty percent of the way through the booking flow rather than zero.

`begin_checkout` is kept, but as its own distinct signal meaning the panel was opened. It is no longer treated as the start of checkout by any platform.

Names were deliberately left unchanged

The conversion actions keep their existing names in Google Ads, Meta and TikTok. Only the event they fire on has changed. That keeps historic reporting readable and avoids breaking any campaign, audience or report that references them by name.

The consequence is that Initiate Checkout counts before and after the change measure different things. Anything spanning the switch should be read as two periods.

Funnels are not one shape

Default funnel reports assume a single journey, and that journey is retail: browse a list, add to a cart, buy. Plenty of businesses do not work that way. Booking platforms, membership products, anything with a configurator, all break the assumption. When they do, the report does not fail loudly. It returns numbers that look broken rather than numbers that are wrong, and the usual conclusion is that tracking is at fault.

The answer is not to force ROLLER's events into a shape they were never going to fit. It is to build a funnel that matches the journey guests actually take.

That exploration now exists in GA4 as **ROLLER Funnel**, using the sequence in the right-hand column above. Alongside it, `checkout_progress` supports a second, narrower funnel measuring drop-off inside the checkout itself, since it carries a step number on every fire.

What ROLLER actually sends

This is the complete list of events the checkout emits, in the order a guest encounters them. It is worth knowing precisely, because several look similar and two of them mean very different things.

Order	Event	When it fires
1	<code>view_item_list</code>	The product list loads.
2	<code>checkout_progress</code>	Fires repeatedly through the flow, carrying a step number.
3	<code>button_clicked</code>	ROLLER's own click tracking, including its consent prompt.
4	<code>begin_checkout</code>	The panel opens. Cart is empty, value is zero.
5	<code>view_item</code>	A specific experience is opened.
6	<code>add_to_cart</code>	Tickets added. Some products skip this and go straight to checkout.
7	<code>remove_from_cart</code>	Items removed.

Order	Event	When it fires
8	view_cart	The cart summary is opened.
9	checkout_button_clicked	Checkout genuinely begins. Requires a populated cart.
10	checkout_progress	Guest details submitted, later step number.
11	purchase	Booking complete. Carries transaction ID, items, value, tax and coupon.

Two events that are easy to confuse

`begin_checkout` is not the start of checkout. It is the panel opening, and it fires before the guest has chosen anything.

`checkout_button_clicked` is the start of checkout. It cannot fire without items in the cart.

Two further notes worth carrying into any reporting built on this. `add_to_cart` does not fire for every product, because some options move straight to checkout, so it should not be treated as a required funnel step. And `checkout_progress` fires more than once per booking with a step number attached, which makes it useful for measuring drop-off inside the checkout itself.

How revenue is counted

This is the part most worth reading closely, because two different numbers are both correct and they answer different questions.

Many Activate bookings are paid with a deposit. A guest books a \$400 group session and pays \$39 today. ROLLER's checkout reports the amount collected, because that's what was charged. But for judging whether an ad campaign worked, the \$400 booking is what matters. No-shows run 2 to 4%, so the booking is almost always worth its full value.

What ROLLER sends

items[]
price 40 × quantity 10

value
39.41
the deposit actually charged



value

\$400.00

full booking value, what ads optimise against

roller_amount_charged

\$39.41

what reconciles to ROLLER's finance reports

Both numbers are recorded on every booking. Reporting uses full value. The charged amount is kept so figures can be reconciled against ROLLER.

Expect GA4 revenue to exceed ROLLER's collected revenue

This is intentional, not a discrepancy. GA4 reports the value of bookings made; ROLLER's finance reports show money received on a given day. A booking made in September for an October visit appears in GA4 in September and in ROLLER's takings across both months.

Use `roller_amount_charged` when the two need to tie out.

One known limitation

Full booking value is calculated from list price. On a discounted booking, reported value reflects the pre-discount total. Discount depth isn't currently available as a separate figure from the checkout, so heavily promoted periods will read high. Worth revisiting if promotions become a regular part of the mix.

Location and partner data

Every location has two identifiers, and both are needed because they come from different systems.

Field	Source	Example
location_id	Marketing site	41, the site's own venue reference

Field	Source	Example
venue_slug	Both	<code>activateorlando</code> , matches ROLLER exactly, which is what lets a browsing session join to its booking
partner_id	Marketing site	<code>0</code> for corporate, and a distinct value for each partner
page_type	Marketing site	<code>location</code> or <code>page</code>
locale	Marketing site	<code>en</code> , ready for the Quebec launch

`venue_slug` is the one that does the real work. Because the marketing site and the checkout both use the same value, a visitor who browses three venues and books one can be followed all the way through. Without it, browsing and booking would be two disconnected datasets.

What this makes possible for the first time

Revenue by venue. The previous setup didn't attach a venue to bookings at all, so network revenue couldn't be broken down by location. It can now, on every event.

Consent

Consent is managed by Termly on the marketing site. Google's tags read the visitor's choice automatically. Everything else has to be told explicitly to wait for consent, and that is configured per tag. This covers Meta, TikTok, Reddit, MNTN and the affiliate pixel.

The checkout shows its own consent prompt, separate from the site's. ROLLER supports passing the site's choice through so guests see one prompt instead of two, which is worth doing: fewer interruptions in the booking flow, and fewer guests declining a second time at the point of purchase.

Recommendation: pass consent into the checkout

As it stands a guest can be asked about cookies twice in one booking. Once on the site, and again inside the checkout after they have chosen a product and are ready to pay. ROLLER supports receiving the site's consent decision, which removes the second prompt entirely.

This is worth doing for reasons beyond tidiness. Consent prompts are one of the most consistently measured sources of friction in ecommerce. Baymard Institute's checkout usability research treats intrusive interstitials as a measurable cause of abandonment, and ContentSquare's benchmarks have recorded bounce rate increases of 25 to 40 percent on pages where a full-screen consent overlay appears before content. A 2022 ETH Zurich study of European consent banners found that burying the reject path two clicks deep correlated with a 12 to 18 point drop in engaged sessions.

Those figures describe a single prompt shown at the start of a visit. No published research measures what a second prompt costs, because few sites present one. What can be said is that the second prompt here appears at the worst possible moment: after the guest has selected a product, entered the checkout, and formed intent to buy. Friction at that point is more expensive than friction on arrival, because the visitor has already invested effort and is closer to converting.

Two benefits, not one

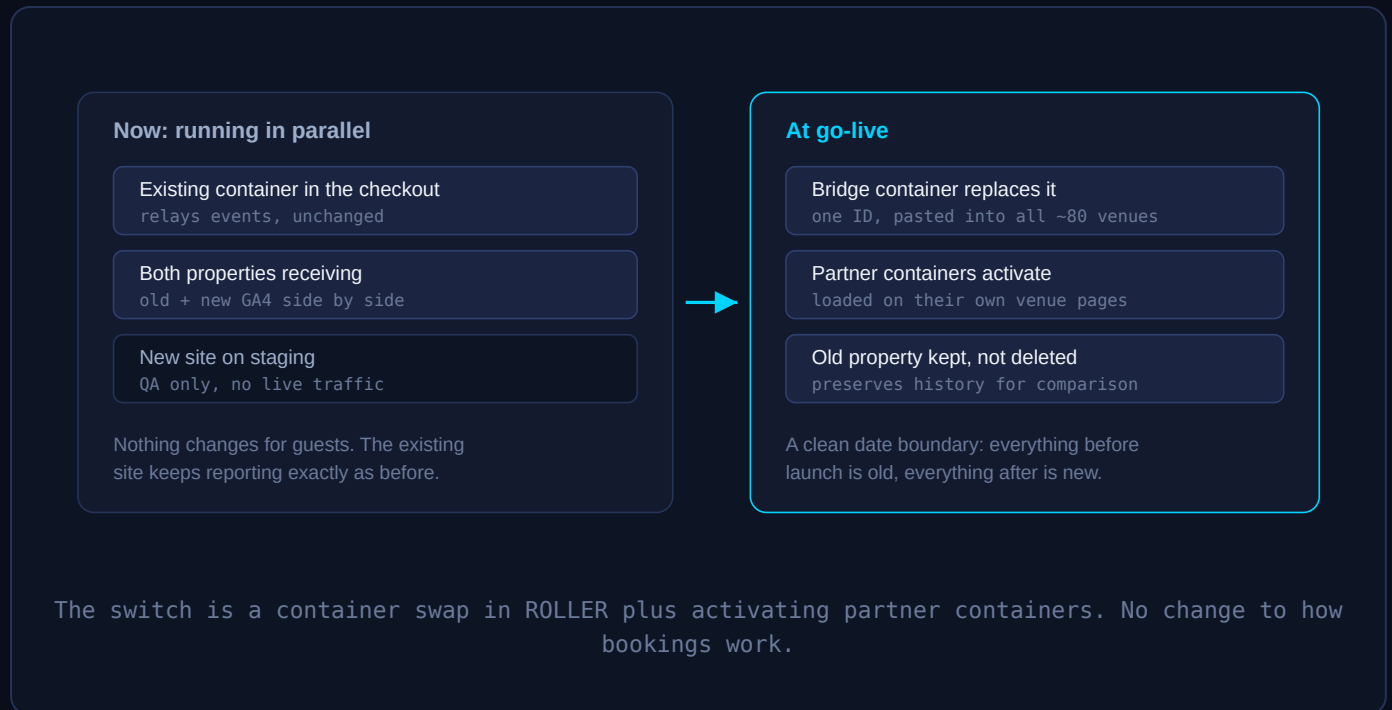
Fewer interruptions in the booking flow. One consent decision instead of two, at the point of highest purchase intent.

Better data as a side effect. A guest who accepted on the site currently gets a second chance to decline inside the checkout. Passing the decision through means their original choice is honoured, which is both more respectful of the guest and more complete for measurement.

ROLLER documents this as supported configuration rather than custom work. It sits outside the analytics scope, but the effect is felt directly in booking conversion, which is why it is raised here.

Today vs go-live

Both the existing and the new setup run side by side through launch, so the old site's numbers stay intact for comparison. The change at go-live is small and well defined.



Do not turn off the existing property

It holds the only record of how the current site performs. Analytics can't backfill a property with data from before it existed, so the old property is the baseline half of every new-versus-old comparison. It should stop receiving new traffic at cutover and stay in place after.

Adding a partner

Onboarding a partner is a short, repeatable sequence. Nothing about it touches the checkout or any other partner.

#	Step	Detail
1	Collect their details	GA4 ID, ad account IDs, conversion labels, pixel IDs, and the venues they operate.
2	Clone the template	A pre-built container with everything wired and placeholders for their IDs.
3	Swap in their IDs	Replace the placeholders. No structural changes needed.
4	Scope it to their venues	The container loads only on the venue pages they operate.
5	Verify one booking	Run a test booking through and confirm it reaches their property and no one else's.

The template is built once and reused. What changes per partner is a handful of ID values and the list of venues.

How to verify it's working

Three checks, in increasing order of certainty. Anyone on the analytics team can run all three.

Live, while browsing

Open the site with Tag Assistant connected and walk through a booking. Booking events should appear on the parent page prefixed with `iframe.`, and that prefix confirms the relay is working. If they're missing entirely, check the approved-addresses list first.

Immediately after a booking

GA4's DebugView shows events in real time. A completed booking should produce exactly one `purchase` event carrying a transaction ID, a venue, and a value. Two purchase events for one booking means something is sending twice.

The following day

BigQuery holds the durable record. Comparing a day's bookings against ROLLER's own transaction report for the same venue is the definitive check, and it is the method used to establish the capture figure quoted earlier.

Verified on 3 September 2026

A live test booking was traced through every stage: relayed out of the checkout, received on the marketing site, recorded with correct full booking value and venue, and confirmed in BigQuery. All identity fields for Enhanced Conversions were present and correctly hashed.

Looking ahead

The measurement setup described here is complete and verified. A few things remain worth planning for, none of which block launch.

The one worth planning for

Moving the checkout to an Activate-owned subdomain removes the cross-domain boundary entirely, rather than working around it. It's supported by ROLLER and would improve capture further, but it is DNS and payment configuration across every venue. A project in its own right, not a launch task.

Two smaller items sit alongside it. Passing the site's consent decision into the checkout would remove the second prompt guests currently see mid-booking, which is supported configuration rather than custom work. And server-side tagging would recover part of the attribution that browser privacy restrictions still cost, at every stage of the funnel.

Both are natural next phases once the new site is established and the current measurement has a full period of data behind it.

Mountains To Sea Media x Very Good Strategies · Analytics architecture for Activate Games

September 2026