Stateful CGNAT Was a Symptom, Not a Necessity

Rethinking the Carrier-Grade Address Translation Chokepoint through a Stateless IPswen NAT Architecture

Bing Swen Sun <bswen@pku.edu.cn>
School of Computer Science, Peking University

Abstract

For decades, the field has treated the ***per-flow state*** carried by Carrier-Grade NAT as inherent to large-scale IPv4 address sharing, while it becomes the dominant source of increasing complexity. Research effort has nonetheless gone entirely into alleviating its effects: faster NAT tables, smarter port allocation, leaner logging, clever clustering and state sync. Almost none has gone into asking *whether the underlying address translation could be stateless* instead.

We argue ***it can***. CGNAT's statefulness is a symptom of a specific historical condition — insufficient IPv4 address space, not a necessity of translation itself, and that condition no longer forecloses the alternative as tightly as it once did. As evidence, we revisit this chokepoint through *CGSAT, a stateless NAT architecture* that replaces per-flow session state with a reversible address-embedding scheme built on IPswen, a variable-length IPv4 extension. CGSAT is deployable as a backward-compatible CGNAT upgrade with no IPv6 dependency. We do not claim CGSAT is a finished cure, only that a cure is possible, and it can be used to reopen a set of deployment, security, architecture upgrade and protocol transition questions that decades of symptom management have left unexamined.

CCS CONCEPTS

• Networks → Network architecture → Network design principles

KEYWORDS

CGNAT, Stateless NAT, IPv4 extension, IPswen

ACM Reference format:

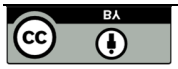
Bing Swen Sun. 2026. Stateful CGNAT Was a Symptom, Not a Necessity: Rethinking the Carrier-Grade Address Translation Chokepoint through a Stateless IPswen NAT Architecture. In *Proceedings of HotNets '26, November 16-17, 2026, Salt Lake City, Utah, USA.* 9 pages. https://doi.org/10.1145/3847290.3859942

1 Introduction

The Internet scales well because most of its architectural components are stateless [1-5]: IP routing, forwarding and fragmentation are designed to work without packet state. Network address translation (NAT) and its carrier-grade extension CGNAT (or NAT444) [39-41], however, are stateful mechanisms: per-flow information (from layer 3 to 4) is necessary for translating multiple internal (private) addresses to/from a single external (public) address. This is a long-standing architectural assumption that for decades has been treated as inherent to large-scale address sharing.

NAT and CGNAT in particular play an indispensable role in Internet growth, and have been a controversial topic for a long time [30]. There are both operational benefits and drawbacks, including the limited protection from external attacks. Although CGNAT has become the de facto mechanism for extending the lifetime of IPv4, and is still the "cash cow" technology for telecom operators and ISPs to generate revenue, as the Internet is still IPv4 dominant [31-33], its reliance on per-flow state has led to increasingly complex and costly network infrastructure [38,40,43]. Maintaining massive translation tables, synchronizing state across redundant gateways, and supporting log retention impose substantial operational burdens and technical challenges. While its costs are widely recognized, the networking community has largely treated stateful translation as an unavoidable consequence or a necessary cost of address translation and sharing, namely, *state maintenance must be inevitable*. Research effort thus has gone almost entirely into alleviating the symptom of CGNAT to manage its state more efficiently: faster session tables, smarter port allocation, leaner logging, clustering and state synchronization, etc.

On the other hand, for IPv6 transition [26-29] various stateless translation techniques have been devised. For example, MAP-E, MAP-T, and Lightweight 4over6 (by IETF WG *softwire* and *v6ops*) [44-46] replace per-flow state with algorithmically computed address and port mappings. But they do not interwork with CGNAT, and they do so by moving the carrier's network onto IPv6 and requiring the customer's own equipment to actively participate in the translation, provisioned via dedicated DHCPv6 options [47]. For an operator able to run an IPv6-capable access network end to end, this is an acceptable solution. IPv6 also helps to solve address scarcity and restoring (internal) end-to-end connectivity. As we discuss below, IPv6 is not an ideal



This work is licensed under a Creative Commons Attribution 4.0 International License.
HotNets '26, November 16-17, 2026, Salt Lake City, UT, USA
© 2026 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-3029-0/2026/11
https://doi.org/10.1145/3847290.3859942

carrier layer of IPv4, and will help little to solve the problem for IPv4 native networks stuck behind stateful CGNAT. Operators with large embedded or legacy equipment bases, or regulatory environments tend to exclude IPv6 from the network. In fact, CGNAT happens to be the biggest obstacle in a smooth transition to IPv6 [31,34], since address scarcity is essentially and arguably preferably addressed by CGNAT itself [30]. The real cost is behind stateful translation, where every translation flow generates state.

As the central thesis of this paper, we challenge such an architectural stateful assumption, and answer questions: *Is per-flow state an inherent necessity of large-scale address sharing? Can we "upgrade" CGNAT to be stateless without any IPv6 dependency? And at what cost? What questions and new lines of inquiry does the idea reopen?* To find answers, we need to consider a new way of scoping as well as a paradigm shift. We need a stateless address translation architecture within the internal extensibility of IPv4's architecture, to address all the major shortcomings inherent to stateful CGNAT — statefulness included, but also other related difficulties: one-way connectivity, broken end-to-end, isolation of NAT branches, etc. We need to scope the carrier-grade translation to stay entirely within existing IPv4 end hosts, with full backward compatibility, touching neither the customer devices nor software.

Under such constraints, we present **CGSAT** (*Carrier-Grade Stateless Address Translation*), a new architectural direction that replaces stateful per-flow translation with deterministic, stateless address translation enabled by the IPswen addressing scheme [18,19], a variable-length IPv4 address extension. Rather than viewing carrier-scale address sharing as state management, CGSAT reformulates it as a packet re-addressing and routing problem, eliminating entirely the need for L4 port tracking, while significantly simplifying gateway operation. This shift has the potential to improve scalability, simplify high-availability design, reduce memory requirements, and enable more efficient hardware implementations. It also helps to find practical solutions to address challenges in existing ISP/cloud networks, where internal connectivity is hampered by multilayered NAT and CGNAT. We use a prototype implementation to demonstrate the feasibility of the proposed architecture, which requires zero per-flow memory and constant complexity of session management and logging, removes state maintenance, port exhaustion and connection loss completely, and achieves substantial improvements in all performance indicators of CGNAT.

Beyond analyzing CGSAT, we argue that stateless carrier-grade address sharing opens a broader design space for Internet architecture. CGSAT suggests that address translation can be performed statelessly if an appropriate addressing scheme is adopted: based on the IPswen scheme, address sharing can instead be reframed as a deterministic, single-stack and reversible address embedding. Our perspective is "spending bits instead of state", using a few more bits in IPswen addresses to get rid of per-flow state, and reframe address translation from a stateful scaling problem to a stateless computational problem. We also need to further understand the architectural implications of this approach, compare it with existing carrier translation mechanisms, and identify key research challenges that remain to be addressed.

In summary, our contribution is that using CGSAT as a case study, we challenge to a long-standing architectural assumption that stateful per-flow translation is inherent to large-scale IPv4 sharing. By exploiting IPswen's backward compatibility and bidirectional interworking with IPv4, CGSAT proposes that address translation can instead be deterministic and stateless, potentially simplifying carrier networking infrastructure while opening a new research space for network architecture. As a new NAT architecture and design space, the contribution of CGSAT over the prior work IPswen is that CGSAT is a more general architecture in embedding various addressing schemes, compatibility support and performance improvement. We believe this work will stimulate discussion on the evolution of Internet infrastructure, and matter to the broader networking community beyond the NAT and CGNAT industry.

2 Stateful CGNAT as a Symptom

Understanding that NAT emerged as a stopgap under acute address scarcity [30,43] to multiplex a single address for a few L4 protocols (TCP, UPD and ICMP that have port numbers or analogous message ID numbers) helps to understand how CGNAT looks the way it does. Statefulness was a handy choice over addressing extension. The reason is historical rather than technical. Then, operational scaling made this choice calcified into permanent infrastructure, where only the few L4 protocols prevail. For small to medium scale traffic, stateful translation works relatively well. With the explosion of per-device flow states from mobile apps, IoT telemetry, and AI LLM tokens with persistent online connection, stateful translation, adopted as a necessary cost, is becoming unsustainable.

2.1 A Chokepoint on the Internet

Once an indispensable function of early Internet explosive growth, CGNAT has now been a challenge for long-term Internet evolution. As "the chokepoint of the Internet",

CGNAT has many sources of complexity to qualify for that designation: session tables, port allocation, timeout management, failover synchronization, logging retention. And none of these functions forward packets; They exist primarily because address translation is stateful [38-43].

Among various major drawbacks of CGNAT, the most prominent issues include: ***Unidirectional connectivity*** *breaks end-to-end;* ***Isolation*** *among NAT branches hinders internal connectivity;* ***Overhead*** *of NAT state tables increases sharply with concurrency;* ***Port conflicts*** *and port exhaustion severely impair network throughput; Huge session* ***logging and tracing*** *data are hard to maintain;* ***Blocking*** *a shared IP address can inadvertently affect many unrelated users*. With the increase of network bandwidths, CGNAT has become one of the Internet's most expensive pieces of infrastructure because it is built upon the assumption that for large-scale address translation unlimited resources are needed to maintain per-flow state. State management is the dominant complexity source of this symptom.

2.2 Why IPv6-as-carrier isn't a feasible substitute

Another way to overcome the statefulness of CGNAT is to adopt IPv6, thereby eliminating address sharing entirely. Quite a few stateless IPv4/IPv6 translation methods (NAT64, MAP-T/E, IVI, etc.) were proposed for IPv6 transition [44-46, 48-51]. It is tempting to consider IPv6 as a carrier layer as well as a single stack environment to overcome the state complexity of CGNAT. *Three things complicate this.*

First, every IPv6 based stateless translation requires full IPv6 deployment and breaks IPv4 backward compatibility. Relying on IPv6 transit without being able to operate pure IPv4 domains is a critical gap. Customer edge is not a passive conduit for IPv6-carried IPv4 traffic. A backward compatible upgrade to CGNAT requires replacing nothing at the customer's premises. IPv6 dependent solutions have to be ruled out to avoid ending up with dual-stack designs.

Second, operators most committed to CGNAT are, by construction, unlikely to be the operators best positioned to deploy IPv6 across their entire access network. Treating IPv6-as-carrier as a stateless substitute for stateful CGNAT asks the operators who need a cure to first acquire the IPv6 resource they are unlikely or unwillingly to have. If IPv6-as-carrier were truly a free substitute, CGNAT deployment should have declined roughly in step with IPv6 access-network availability. It is an empirical question outside this paper's scope, but is the right one for the field to be asking.

Additionally, a strong prerequisite for IPv6 deployment makes it nearly impossible for IPv6 to supplant or upgrade CGNAT. IPv6 is normally deployed in "clean slate" environments [1,2] where the operators need to have full control of both ends of the network, including the services and the end points. *CGNAT as an IPv4 based architecture is not the case.* NAT traversal [52-54], for example, blocks most IPv6-as-carrier mechanisms, the same as packet fragmentation and L3 (layer 3) dependencies of L4 (layer 4) block protocol translation based NAT/PT. Whereas in such heterogeneous environments where legacy devices are pervasive, in work, and unnecessary or impossible to upgrade [34,35] IPswen tends to be more applicable.

3 Statelessness with SATII

Based on this perspective, we thus take the absence of an IPv4-native stateless translation design not as evidence that none is possible, but as evidence that the question needs to be seriously asked. Constrained by the huge installed base of NAT and CGNAT, a stateless "upgrade" to CGNAT needs to be **1**, strictly backwards compatible to and interworking with existing CGNAT, and **2**, incrementally deployable in existing carrier network environments while fixing CGNAT's statefulness. To this end, we must stop hacking L4 ports and instead expand the L3 addresses dynamically using appropriate IPv4 addressing extensions, so to shift the burden from L4 port-state to a larger L3 address container that legacy translation can safely ignore, but upgraded translation can route statelessly.

3.1 IPswen: Address Bits instead of State Tables

Like the stateless transition mechanisms in IPv6, carrier-grade address translation can be performed statelessly if the address architecture is sufficiently extensive.

As a case study, here we choose IPswen [18,19,25] for this purpose. IPswen adopts a variable length addressing and routing scheme designed to preserve the entire IPv4 stack while remaining bidirectionally interworkable with it. IPswen addresses consist of the standard 32-bit IPv4 *base address* plus a variable-length array of up to 7 *extended address* bytes. Thus, IPswen provides a variable length addressing space made up of 8 levels of subspaces, with address lengths ranging from 4 to 11 bytes, where ordinary 32-bit IPv4 addresses are 1-1 mapped to a "level 0" subspace, e.g., the three IPswen addresses 1.2.3.4.**0**, 1.2.3.4.**1**.5 and 192.168.0.1.**7**.1.2.3.4.5.6.7 are of extension level 0 (no extension), level 1 and level 7 respectively.

To encode the extended address bytes in the IPv4 packet header, two unused option sets of the reserved *type 11* are adopted [18,19], where the type byte $\{111f_T\ f_S xxx\}$ has the bit f_T for a source ($f_T = 0$) or a destination ($f_T = 1$) address extension; the f_S bit as the sign flag of the extension level ($f_S = 0$ for positive levels and $f_S = 1$ for negative levels from

-1 ~ -7), and the lowest three bits *xxx* denoting the binary value (0~7) of the address level. IPswen addresses of negative extension levels by design are routed only on the first 4 bytes (the IPv4 base address) while the extended bytes are reserved to represent internal private-network hosts. This mechanism is used in this paper as a convenient way to employ the IPswen routing facilities. Thus, IPswen has sufficient addresses to embed the L4 state table entries of up to two layers of stateless address embedding (with both encoding and decoding), which is exactly what CGSAT needs to upgrade CGNAT.

In the following, we firstly design a stateless translation method based on the addressing capability of IPswen's 7-level subspaces, using its routing facilities of negative-level addresses. We then scale it to carrier-grade deployment.

3.2 SATII: Compute, Don't Store

Based on a large enough carrier layer, the core thesis of implementing a new stateless address translation then is straightforward: "*Compute, Don't Store*". The algorithm we adopt for implementing stateless translation is a very simple and efficient reversible address-embedding scheme: we embed the private IPv4 addresses $a_0.b_0.c_0.d_0$ (in part or whole, according to deployment requirements) inside the NAT domain into the public addresses $a_x.b_x.c_x.d_x$ outside the NAT domain as extended addresses of *negative embedding levels*: $a_x.b_x.c_x.d_x.\mathbf{EmbeddingLevel}.a_0.b_0.c_0.d_0$, where the ***EmebeddingLevel*** = -4, -3 or -2. (*-4 is not used by CGSAT.*)

In the following we will call this mechanism *stateless address translation between IPv4 and IPswen*, or **SATII** for short. As discussed later, thanks to strict backward compatibility between IPswen and IPv4, SATII on the IPswen carrier layers can directly interwork, replace or integrate with existing CGNAT routers. This is impossible to achieve by employing IPv6-only cores with stateless translation edges (like 464XLAT), which ends up with stacking stateful NAT on top of dual-stack setups.

3.3 CGSAT with Two-Layer SATII

Based on SATII, a *carrier-grade stateless address translation*, or **CGSAT** for short, then can be designed as a two-layer SATII architecture. The following Figure 1 illustrates the CGSAT architecture. It is isomorphic to CGNAT with two layers of stateful NAT, which is one of the necessary conditions for the two to maintain full compatibility.

CGSAT includes two layers of stateless address translation, namely the inner and outer SATII. The former embeds private IPv4 addresses into the IPswen addresses of the CGSAT domain; the latter converts the CGSAT domain's IPswen address into negative-level IPswen addresses of the IPv4 base address outside the domain, or in the case where CGSAT degrades to CGNAT (as discussed later), the outer SATII converts them into an external public IPv4 address and adds port mapping relationships on both the inner and outer sides.

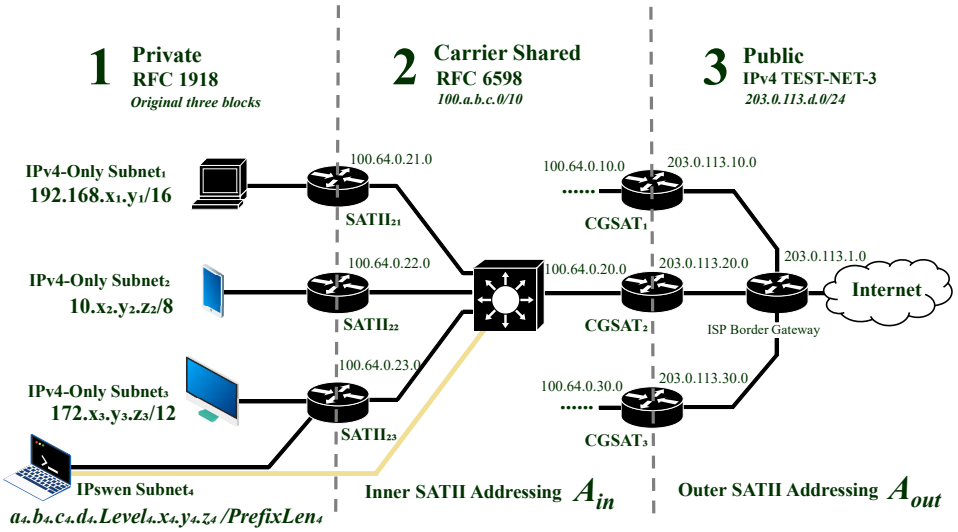


Figure 1: The CGSAT Architecture

The address embedding algorithm of the inner SATII A_{in} and the decoding algorithm A_{out} of the outer layer SATII can be described with the following two formulas:

$$A_{in} = 100.a.b.c.\{\ \mathbf{-2}.x_1.y_1 \mid \mathbf{-3}.x_{2,3}.y_{2,3}.z_{2,3}\ \},$$

$$A_{out} = 203.0.113.d.\{\ \mathbf{-6}.a.b.c.\mathbf{-2}.x_1.y_1 \mid \mathbf{-7}.a.b.c.\mathbf{-3}.x_{2,3}.y_{2,3}.z_{2,3}\ \},$$

where the parameters of dotted decimals are: $a = 64...127$, $x_3 = 16...31$, $x_2 \neq 16...31$, $x_{1,2,4}/y_{1,2,3,4}/z_{2,3,4}/b/c/d/a_4/b_4/c_4/d_4 = 0...255$, and $Level_4 = -7...7$, $PrefixLen_4 = 8...88$.

The two-layer SATII of CGSAT essentially embeds specific types of IPv4-only addresses (such as various private addresses or any designated dedicated IPv4 address block) configured on user-end host nodes located on the edge network into longer IPswen addresses. The edge nodes then achieve interconnectivity within the whole CGSAT domain (between internal subnets of different CGSAT branches) via the IPswen BGP4x routing system [23-25]. For example, the inner $SATII_{21}$ embeds the last two bytes of a *Class B* private address 192.168.x_1.y_1, $SATII_{22}$ embeds the last three bytes of a *Class A* address 10.x_2.y_2.z_2, and $SATII_{23}$ embeds 172.x_3.y_3.z_3 where x_3 = 16~31, which are the IPswen addresses of the external network interfaces of the corresponding SATII routers, which again translate them into IPswen addresses starting with a carrier-grade (shared) private address 100.a.b.c. For instance, the subnetwork 192.168.x_1.y_1/16 is converted into an IPswen address block 100.64.0.21.-**2**.x_1.y_1 using the $SATII_{21}$ router, and the subnetwork 10.x_2.y_2.z_2/8 is converted into an IPswen block 100.64.0.22.-3.x_2.y_2.z_2 using $SATII_{22}$, and so on. A key point here is the utilization of the negative address levels up to -7 of the IPswen, where negative levels indicate that when routing the IP packets, only the first four bytes (IPv4 base address) of the IPswen address are used to forward the IP packet within the CGSAT domain, while the extended addresses following the negative levels is retained as the addresses of the internal private SATII network nodes. Note that every SATII branch (subnetwork) of each of the $CGNAT_{1,2,3}$ domains now has connectivity end-to-end to the other SATII branches in the whole domain.

Thus, based on IPswen, CGSAT supports carrier-grade stateless address translation without mandating IPv6 adoption or requiring modifications to end-user devices, OS or even applications. It is also clear that the inner SATII layers could not be replaced by IPv6 as an overlay network; neither SIIT or stateless NAT46/64 interwork with CGNAT.

3.4 The "Color TV Compatibility"

IPswen supports the "Color TV compatibility", meaning a legacy IPv4 router can pass an IPswen packet (ignoring the extended bytes or treating them as options/padding) while IPswen-aware routers utilize the full extended address information [19], much like the bidirectional compatibility and interworking between color TV and black/white TV.

This relationship largely holds true between CGSAT and CGNAT, where the former maintains strict backward compatibility with legacy CGNAT, allowing for plug-and-play functionality and in-place upgrades/replacements of, or "downgrade" to, existing CGNAT networks.

3.5 Interworking with Legacy CGNAT

It is noteworthy that some legacy NAT and CGNAT deployment configurations may block IPswen packets and cannot be addressed by the edge hosts within CGSAT. Namely, when the external network interworking with a CGSAT router does not support IPswen, e.g., the ISP's public network has not been upgraded to IPswen, and an innermost host node attempts to access a legacy IPv4 host on the external network. For this, CGSAT has two options:

(1) A stateless fallback, such as a Port-Set Identification (PSID) or A+P mode, which is preferrable, where the outer SATII uses a deterministic port-set partitioning to maintain the O(1) static algorithmic mapping $A_{out} \leftrightarrow$ *Src-Port-Range-IDs*, similar to the established softwire principles (e.g., MAP-T [45], Deterministic CGNAT [42]); Or

(2) A stateful fallback, as a downgrade "degradation mode" to operate as normal CGNAT, where the outer layer disables SATII and reverts to stateful NAT, utilizing the L4 transport-layer protocol ports/message-IDs above the IP layer as usual to identify internal host nodes. However, the performance of the inner-layer SATII remains unaffected, and end-to-end direct connectivity across CGSAT domains within the ISP's internal network is consistently preserved. Furthermore, the endpoint {*IP address : L4 port number*} on the inner side of the CGSAT gateway corresponds to a direct connection endpoint of the internal hosts. Since the inner-layer SATII routers all do not require any L4 ports, it becomes possible to initiate inbound connections from the external public network to the innermost hosts within the CGSAT domain, which is a capability that CGNAT lacks.

4 Statelessness as a Game Changer

We now discuss possible consequences of the SATII mechanisms and its CGSAT architecture, trying to understand how CGSAT could be a game changer if large-scale address translation becomes a pure function, where in contrast, CGNAT meets two directions of dead end: it's a dead end in scaling the per-flow translation state, and a dead end in evolving the L3 address space. CGSAT based on IPswen may solve them both.

4.1 Advantages of CGSAT over CGNAT

First of all, CGSAT offers a series of straightforward significant advantages compared to stateful CGNAT:

1. **Greater universality**, restoring end-to-end among hosts within the whole carrier network by not only

interconnecting internal SATII branches but also with interoperability between legacy IPv4-only devices and applications, and newly deployed IPswen hosts and subnets.

2. **Versatility** to support a broader range of upper-layer applications and software that accommodates arbitrary L4 or L5 message protocols.

3. Enabling **port passing** — the ability to address the private IP addresses of the innermost nodes using the external port numbers of the CGSAT gateway router.

4. **Reduced volume** of session-state tables and logging.

5. Overwhelming **operational performance** compared to CGNAT in almost all the NAT indicators: *throughput, forwarding rate, latency and jitter, establishment rate of connection, maximum concurrent sessions, session persistence and recovery rate, port conflict and exhaustion rate, address multiplexing ratio*, and *fault recovery time*.

6. Full **compatibility and interoperability** with CGNAT. CGSAT encompasses all the functionality of CGNAT and when interworking with legacy networks, can be "downgraded" to operate as a standard CGNAT instance.

Furthermore, CGSAT's backward compatibility supports native IPv4 systems on the edge hosts to interconnect end-to-end across multiple CGSAT domains of an AS network.

4.2 Implementation and Early Evaluation

We use XDP (eXpress Data Path) [55] to implement the SATII routers that run at the NIC driver level on commodity Linux servers, which process packets before *sk_buff* allocation. A testbed network of 500 SATII routers and IPv4+IPswen mixed host nodes is configurated as Figure 1, where the carrier network interconnects SATII nodes with Intel E810-CAM2 NICs. An extended MP-BGP [6-8] server using BIRD [12] is used to propagate the new routes of various IPswen embedding addresses of negative levels. Route propagation of each SATII node is done each time when it finishes system initialization. Early evaluation on the testbed network has been highly encouraging. Table 1 is a quick comparison of networking metrics.

Table 1: Comparison of Networking Metrics

Metric	IPv4 (Baseline)	CGNAT (Stateful)	CGSAT (Stateless)
Latency (RTT)			
Idle Load	12.42 ms	12.85 ms	12.48 ms
50% Load	18.30 ms	22.60 ms	18.42 ms
90% Load	25.40 ms	70.10 ms	25.90 ms
Jitter (Variance)			
p99	1.2 ms	45.0 ms	1.8 ms
Throughput			
Idle Load	5.10 Gbps	5.06 Gbps	5.04 Gbps
50% Load	5.07 Gbps	4.30 Gbps	4.69 Gbps
90% Load	4.51 Gbps	2.08 Gbps	3.90 Gbps

Under 90% load, CGSAT sustains 86% of link capacity while CGNAT degrades to 46%, with p99 jitter of 1.8ms versus 45ms. Each SATII node requires zero per-flow memory and port ranges, free of state-explosion and port exhaustion. At each SATII node, memory space remains nearly constant for concurrent TCP connections ranging from 1k to 100 million. By contrast, CGNAT's memory overhead increases from 5 MB to over 50 GB and easily exhausts the 48GB system memory. For each address/port pair mapping, CGNAT needs ~200 bytes per table entry, and additional ~300 bytes to tracking each TCP connection.

To verify the SATII behavior, we implement the CGSAT core feature, the address embedding scheme, in the Vector Packet Processing (VPP) vectorized router packet [56,57]. A VPP CGNAT+CGSAT testbed is deployed using the same configuration, where CGNAT are implemented as plugins attached to the unicast processing pipelines like CGSAT. We reaffirm the state table behavior of both: CGNAT creates one session per flow at each NAT gateway, while CGSAT maintains zero per-flow translation sessions. For each 50k TCP and UDP flows at 5,000 connection requests per second, CGNAT uses approximately 861MB and 738MB tracking memory overheads respectively. This prototype VPP testbed also verifies the design constraints that CGSAT preserves backward full compatibility and interworkability with CGNAT, for the existing CGNAT pipelines remains at each SATII node. Adding efficient implementation of the PSID (or A+P) fallback mechanism is also straightforward.

We also evaluate various combinations of legacy and upgraded (compatible shim enabled) software components (Linux kernel, *libc*, socket API apps) at the edge host's side. Stateless translation achieves near line-rate throughput with negligible CPU overhead, while legacy applications can be transparently interconnected across the whole carrier network with multiple CGSAT domains and subnets.

Additionally, we evaluated the IPswen options that are necessary for embedding addresses both in the testbed ("intranet") and on the global Internet. Internally within the operator's network, CGSAT can be incrementally deployed, where most software (Linux kernel in particular) based routers already forward unknown IPswen options normally. Globally, in contrast to conventional wisdom [14,15], the Internet may have had better acceptance rates of IPswen options than previous [16], possibly due to increasing adoption of cloud-based software CGNAT services using VPP and the like. A large-scale measurement of reachability across approximately 75,000 ASes from several geographically distributed vantage points reveal that IPswen options have 38–64% reachability on today's Internet, with bounded RTT overhead under 32% [21,23,25].

What does this mean to the deployability of CGSAT on today's Internet? In general, on the Internet core layers IPswen options already work relatively well. Interworking CGSAT domains across the core layers using IPswen extension options may have already been a worthy undertaking. On the edges, where legacy NAT centric, the automatic switch of IPswen-in-IPv4 tunnelling [20,23,25] is then an applicable method to cross nodes where direct IPswen reachability is lost. As discussed above, the outer SATII interworking with legacy networks behind CGNAT has two fallback options modes for the switching.

5 Open Questions, New Directions

The feasibility of stateless carrier-grade address translation opens a series of interesting questions. Treating large-scale address sharing as deterministic translation exposes a design space that is previously underexplored.

Firstly, the following questions may lead directions of a new research agenda, including (not limited to): **Mapping Theory:** How should deterministic mappings be designed in a more general way? **Transport Compatibility**: Can long-lived flows survive gateway migration without state? **Hardware Design**: Can SATII's stateless translation be implemented entirely in programmable ASIC pipelines? How do line-rate hardware ASICs handle the embedding IPswen address lookups without dropping performance? **Security**: What are the security implications of stateless translation? How should abuse attribution and traceback work? What are the implications for lawful interception with CGSAT? How should operators rethink logging, compliance, and attribution? What implications in source-address privacy and asymmetric routing? **Deployment:** What incentives or actual benefits does CGSAT provide to the operators most committed to CGNAT, or to those already able to supply IPv6-dependent alternatives?

Secondly, objections are also anticipated, where typical concerns include: Is this architecture generally deployable? Does it scale to the global Internet? What happens when an application embeds a legacy 32-bit IP in its payload? How to handle MTU and fragmentation?

Thirdly, in our exemplary design SATII is based on IPswen, which again is based on the inherent extensibility of IPv4. Thus, other IPv4 extensions may also be applicable for constructing new deterministic NAT versions of the CGSAT design. This line of inquiry leads to IPv4 remedies that has a rich set of related studies in the literature.

Lastly, as CGSAT would also reframe the entire IPv6 deployment narrative and challenge the IPv6 orthodoxy in the post-IPv4 era, we should ask: Why hasn't IPv6 replaced stateful CGNAT completely? How should we interwork IPswen based CGSAT with the increasing IPv6 deployment? Could CGSAT influence future transition from IPv4 to IPv6?

6 Conclusion

Stateful NAT and CGNAT were devised as a stopgap to deal with address scarcity and in reality, the two have essentially solved that problem, but with inherent complexity and increasingly serious flaws. The networking community has spent three decades on managing the complexity of stateful address sharing through larger state tables, faster lookup algorithms, improved synchronization, and more capable hardware. With CGSAT, we explore a different direction: rather than keeping optimizing state management, we ask whether state management is necessary at all. If large-scale address sharing can instead be expressed as deterministic stateless translation, future Internet infrastructure may be much simpler than today's stateful CGNAT-centric deployments.

CGSAT does not actually resolve the whole question, but only suggests a new design space to explore — if we trade per-flow L4 or L5 state for a sufficiently simple L3 address extension at some stateful chokepoint, we can make the whole architecture stateless and overcome many of the flaws inherent to that stateful design.

Lastly, the potential impact of an IPswen based single-stack CGSAT architecture to the long-term evolution of Internet is worth discussing. The ultimate direction of the Internet is inevitably towards a single-stack architecture; it cannot and should not permanently be stuck in a parallel dual-stack mode. This, however, does not necessarily mean an IPv6 single stack, but could also be some new single stack with sufficient backward IPv4 compatibility. As well known, *the major difficulty with IPv6 is that it does not provide a logical and realistic exit strategy for existing IPv4 technology bases.* If IPv6 cannot eliminate/upgrade CGNAT while IPswen can, then it sounds reasonable that we should allow fair and free competition between IPv6 and IPswen and the like technologies. For an Internet that is forever evolving, IP/NG should never be a done deal. Between IPv6 and IPv4+massive NATs, IPswen may become a third path for future Internet infrastructure. Let the market, users, and basic economic rules make the final choice.

ACKNOWLEDGMENTS

The author thanks the shepherd of this paper and the anonymous HotNets reviewers for their valuable comments and suggestions. This work is based on previous results of teamwork, including the BGP4x design of Yu Huang, the ISP4x architecture development by Yi Li, and the VPP prototype implementation of CGSAT by Ruidong Liu and Zhen Chen, who are all members of the *IPswen Stack Team* at Peking University (*https://IPswen.net*).

REFERENCES

[1] S. Akhshabi and C. Dovrolis, 2011. The Evolution of Layered Protocol Stacks Leads to an Hourglass-Shaped Architecture. In *Proceedings of ACM SIGCOMM 2011*, pp 305–310.

[2] James McCauley, Yotam Harchol, Aurojit Panda, Barath Raghavan, and Scott Shenker, 2019. Enabling a Permanent Revolution in Internet Architecture. In *Proceedings of the ACM SIGCOMM 2019*, pp 1–14.

[3] Z. Zhang, Liu, Wang. DIP: Unifying Network Layer Innovations using Shared L3 Core Functions. *HotNets '22*. Austin, TX, USA, 14–15 November 2022.

[4] Guilherme Ribeiro, Luis Pedrosa, Salvatore Signorello, Fernando M. V. Ramos, 2024. Internet Architecture Evolution: Found in Translation. *HotNets '24*. Irvine, CA, USA, 18–19 November 2024. https://doi.org/10.1145/3696348.3696876

[5] Alberto Medina, Mark Allman, and Sally Floyd, 2005. Measuring the Evolution of Transport Protocols in the Internet. *ACM SIGCOMM Computer Communication Review*, 35(2):37–52, 2005.

[6] Y. Rekhter, T. Li, and S. Hares, 2006. A Border Gateway Protocol 4 (BGP-4). *RFC 4271*.

[7] T. Bates, R. Chandra, D. Katz, and Y. Rekhter, 2007. Multiprotocol Extensions for BGP-4. *RFC 4760*.

[8] Jeffrey Haas and John Scudder, 2024. Extending BGP: Reviewing the past and exploring the future. *IETF 119*.

[9] Kevin Hoarau, Pierre Ugo Tournoux, and Tahiry Razafindralambo, 2021. Suitability of Graph Representation for BGP Anomaly Detection. In *2021 IEEE 46th Conference on Local Computer Networks (LCN)*, pp 305–310. IEEE, 2021.

[10] Mengying Xu and Xing Li, 2020. BGP Anomaly Detection Based on Automatic Feature Extraction by Neural Network. In *2020 IEEE 5th Information Technology and Mechatronics Engineering Conference (ITOEC)*, pp 46–50. IEEE, 2020.

[11] S. Hanks, T. Li, D. Farinacci, and P. Traina, 1994. Generic Routing Encapsulation (GRE). *RFC 1701*.

[12] The BIRD Internet Routing Daemon Project. https://bird.network.cz/

[13] Stefan Nilsson and Matti Tikkanen, 1998. Implementing a dynamic compressed trie. In *Workshop on Algorithm Engineering 1998*.

[14] Rodrigo Fonseca, George Porter, Randy Katz, Scott Shenker, and Ion Stoica, 2005. IP options are not an option. *Technical report*. EECS Department, University of California, Berkeley, 2005.

[15] F. Gont, R. Atkinson, and C. Pignataro, 2014. Recommendations on filtering of IPv4 packets containing IPv4 options. *BCP 186*. February 2014.

[16] Brian J Goodchild, Yi-Ching Chiu, Rob Hansen, Haonan Lua, Matt Calder, Matthew Luckie, Wyatt Lloyd, David Choffnes, and Ethan Katz-Bassett, 2017. The record route option is an option! In *Proceedings of the 2017 Internet Measurement Conference (IMC '17)*, pp 311–317, 2017.

[17] Ramesh Govindan and Vern Paxson, 2002. Estimating router ICMP generation delays. In *Passive & Active Measurement (PAM), 2002*.

[18] Bing Swen Sun, 2022. IPswen: A long term evolution approach for the IPv4 Internet architecture. In *Proceeds of IFIP Networking Conference 2022*. Catania, Italy, 13-16 June 022. https://ieeexplore.ieee.org/document/9829762

[19] Bing Swen Sun, 2023. Internet evolution with IPswen: Towards the next generation TCP/IPv4 Internet architecture. In *Proceeds of 2023 International Symposium on Networks, Computers and Communications (ISNCC '23)*. Doha, Qatar, 23-26 Oct.2023. https://ieeexplore.ieee.org/document/10323919

[20] Bing Swen Sun, Yu Huang, and Yi Li, 2025. Towards a Viable Design of a Global IPswen Testbed. In *Proceedings of the 9th Asia-Pacific Workshop on Networking (APNet 2025)*. Shanghai, China, 7-8 August 2025. https://doi.org/10.1145/3735358.3737758

[21] Yu Huang, Yi Li, Run Dong Liu, and Bing Swen Sun, 2025. Exploring the Potential of Unknown IPv4 Options. In *Proceedings of the 9th Asia-Pacific Workshop on Networking (APNet 2025)*. Shanghai, China, 7-8 August 2025. https://doi.org/10.1145/3735358.3737766

[22] Yi Li, Yu Huang, Run Dong Liu, and Bing Swen Sun, 2025. Is Reverse Traceroute Reliable?. In *Proceedings of the 9th Asia-Pacific Workshop on Networking (APNet 2025)*. Shanghai, China, 7-8 August 2025. https://doi.org/10.1145/3735358.3737765

[23] Yi Li, Rundong Liu, Zhen Chen, and Bing Swen Sun, 2026. Towards Option-Based IPv4 Address Extension: A Measurement Study and Testbed Deployment. In *Proceedings of the 10th Asia-Pacific Workshop on Networking (APNet 2026)*. Singapore, 6-7 August 2026. https://doi.org/10.1145/3820441.3820454

[24] Yi Li, Rundong Liu, Zhen Chen, and Bing Swen Sun, 2026. ISP4x: Restoring IPv4 End-to-End Connectivity via Stateless Address Extension. In *Proceedings of the 10th Asia-Pacific Workshop on Networking (APNet 2026)*. Singapore, 6-7 August 2026. https://doi.org/10.1145/3820441.3820467

[25] Bing Swen Sun, 2026. Design and Implementation of a Testbed Network for IPswen Development. In *Proceeds of 2026 International Symposium on Networks, Computers and Communications (ISNCC '26)*. Bristol, UK, 8-10 Sept. 2026. https://ieeexplore.ieee.org/document/11693825

[26] S. Deering and R. Hinden, 2017. Internet Protocol, Version 6 (IPv6) specification. STD 86, RFC 8200. July 2017.

[27] Ravi Kumar CV and Hrithik Goyal, 2019. IPv4 to IPv6 Migration and Performance Analysis using GNS3 and Wireshark. In *2019 International Conference on Vision Towards Emerging Trends in Communication and Networking (ViTECoN)*, pp 1–6. IEEE, 2019.

[28] R. Hinden and J. Postel, 1996. IPv6 testing address allocation. *RFC 1897*.

[29] R. Fink and R. Hinden, 2004. 6bone (IPv6 Testing Address Allocation) phaseout. *RFC 3701*.

[30] Geoff Huston, 2017. Opinion: In defence of NATs. *APNIC Blog*. https://blog.apnic.net/2017/09/06/opinion-defence-nats/

[31] Geoff Huston, 2024. IPv6 deployment: 30 Years Later. *APNIC IPv6 Deployment, APNIC 57*. Bangkok, 28 February 2024.

[32] Geoff Huston, 2024. Is the DNS ready for IPv6? Not yet. *V6OPS Working Group Meeting, IETF 119*. Brisbane, 20 March 2024.

[33] Geoff Huston, 2016. Is IPv6 really faster? *NANOG 66 Hacking conference*.

[34] GK Ordabayeva, M Othman, B Kirgizbayeva, Zh D Iztaev, and A Bayegizova, 2020. A Systematic Review of Transition from IPv4 To IPv6. In *Proceedings of the 6th International Conference on Engineering & MIS 2020*, pp 1–15.

[35] Amit Jain, Manpreet Singh, and Pankaj Bhambri, 2021. Performance Evaluation of IPv4-IPv6 Tunneling Procedure using IoT. In *Journal of Physics: Conference Series, volume 1950*. IOP Publishing.
[36] D. Schinazi and T. Pauly, 2017. Happy eyeballs version 2: Better connectivity using concurrency. *RFC 8305*.
[37] D. Wing and A. Yourtchenko, 2012. Happy eyeballs: Success with dual-stack hosts. *RFC 6555*.
[38] M. Ford et al., 2011. Issues with IP Address Sharing. *RFC 6269*.
[39] R. Penno et al., 2013. Common Requirements for Carrier-Grade NATs (CGNs). *RFC 6888*.
[40] C. Donley et al., 2013. Assessment of the Impact of High-Scale NAT (NAT444) on Network Applications. RFC *7021*.
[41] Y. Shirasaki et al., 2012. NAT444. *IETF Internet-Draft*, 2012.
[42] C. Donley et al., 2014. Deterministic Address Mapping to Reduce Logging in Carrier-Grade NAT Deployments. *RFC 7422*.
[43] Richter, P. et al., 2016. A Multi-perspective Analysis of Carrier-Grade NAT Deployment. In *Proc. ACM IMC 2016*. (arXiv:1605.05606)
[44] O. Troan et al., 2016. Mapping of Address and Port with Encapsulation (MAP-E). *RFC 7597*.
[45] X. Li et al., 2015, Mapping of Address and Port using Translation (MAP-T). RFC *7599*.
[46] Y. Cui et al., 2015. Lightweight 4over6: An Extension to the DS-Lite Architecture. *RFC 7596*.
[47] I. Farrer et al., 2015. DHCPv6 Options for Configuration of Softwire Address and Port-Mapped Clients. *RFC 7598*.
[48] X. Li, C. Bao, F. Baker, 2011. IP/ICMP Translation Algorithm. *RFC 6145*.
[49] M. Bagnulo, P. Matthews, I. van Beijnum, 2011. Stateful NAT64. *RFC 6146*.
[50] A. Durand et al., 2011. Dual-Stack Lite Broadband Deployments Following IPv4 Exhaustion. *RFC 6333*.
[51] H. Jiang et al., 2011. An Incremental Carrier-Grade NAT (CGN) for IPv6 Transition. *RFC 6264*.
[52] A. Keranen, C. Holmberg, J. Rosenberg, 2018. Interactive Connectivity Establishment (ICE). *RFC 8445*.
[53] M. Petit-Huguenin et al., 2020. Session Traversal Utilities for NAT (STUN). *RFC 8489*.
[54] T. Reddy et al., 2020. Traversal Using Relays around NAT (TURN). *RFC 8656*.
[55] Toke Høiland-Jørgensen, Jesper Dangaard Brouer, Daniel Borkmann, John Fastabend, Tom Herbert, David Ahern, and David Miller. 2018. The eXpress Data Path: Fast Programmable Packet Processing in the Operating System Kernel. In *Proceedings of the 14th International Conference on Emerging Networking EXperiments and Technologies (CoNEXT '18)*, pp 54–66. doi:10.1145/3281411.3281443
[56] The Fast Data Project (FD.io) Project. https://fd.io/
[57] Leonardo Linguaglossa, Dario Rossi, Salvatore Pontarelli, Dave Barach, Damjan Marjon, and Pierre Pfister, 2019. High-speed data plane and network functions virtualization by vectorizing packet processing. *Computer Networks, 149:187–199, 2019.*