NDN Impact on Tactical Application Development

Jeff Burke
UCLA
jburke@remap.ucla.edu

Alex Afanasyev
FIU
aa@cs.fiu.edu

Tamer Refaei
The MITRE Corporation †
mrefaei@mitre.org

Lixia Zhang
UCLA
lixia@cs.ucla.edu

***Abstract*—Named Data Networking (NDN) is a network architecture that forwards data directly based on application-defined names. This paper describes how NDN enables software design patterns that have been successful in a variety of Internet applications to be used in battlefield scenarios. These scenarios involve highly dynamic and disrupted network conditions where implementations over the TCP/IP architecture have struggled. The six patterns include: host-independent abstractions, multicast communication, pervasive network-accessible storage, opportunistic communication, namespace synchronization as transport, and data-centric security. The patterns are motivated by previous research into applications using NDN, which is briefly summarized.**

***Index Terms*—Named Data Networking, Software architecture**

I. Introduction

Named Data Networking (NDN) [1] is a proposed network architecture that enables efficient and secure communication under highly dynamic and disrupted network conditions [2], such as those encountered in battlefield scenarios. This makes NDN a particularly well-suited architecture for tactical applications that face multiple challenges from the existing TCP/IP architecture. Through application of NDN software design patterns—host-independent abstractions, multicast communication, pervasive network-accessible storage, opportunistic communication, namespace synchronization as transport, and data-centric security—tactical application designs can be simplified while hardening security and robustness under challenging network conditions.

NDN forwards data based on application-defined names. As a replacement for the TCP/IP architecture,[1] it provides request-response semantics at the network layer that are similar to web semantics, but at packet granularity. It does this without requiring host addressing or name-to-address mappings, such as those provided by the Domain Name System (DNS). Each data packet is cryptographically bound to its name by a cryptographic signature or similar mechanism. Stateful forwarding is used to route packets through the network without source and destination addresses [3].

This material is based upon work supported by the National Science Foundation under award CNS-1719403.

†Author's affiliation with The MITRE Corporation is provided for identification purposes only, and is not intended to convey or imply MITRE's concurrence with, or support for, the positions, opinions or viewpoints expressed by the author. It is approved for Public Release; Distribution Unlimited. Case Number 18-2377.

[1]NDN can also run as an overlay on top of TCP/IP networks and, in fact, on top of any medium that can carry bits.

NDN makes named data the new "thin waist", or common interoperability layer. This enables application development and deployment patterns that successfully use named data at upper layers (e.g., HTTP-based approaches) but which depend on continuous connectivity or complex network management to be applied in tactical networks. The assumptions of connectivity always being available or being able to rely on many-layered network provisioning and management approaches do not hold under 1) challenging and dynamic communication environments (e.g., battlefield scenarios) and 2) resource-limited devices (e.g., IoT). Removing the need for applications to manage the binding between data and endpoint addresses can significantly reduce complexity while preserving the robustness of a data-centric request-response paradigm as provided by web protocols, including HTTP and variants such as CoAP [4].

To summarize: NDN communicates using application-level naming, which leads to systems that are semantically interconnected at the network layer. Using names directly for packet delivery provides Web-like request-response semantics at the network layer. With NDN, the common stack used by all nodes provides capabilities normally left to middleware or higher-level frameworks, thus 1) simplifying application design, network management, and 2) reducing the attack surface of deployed networks dramatically, as we elaborate in Section III. Further, NDN secures data directly at its production, decoupling security from both middleboxes (e.g., CDNs) and the channels over which data are delivered.

Our objective in this paper is twofold. First, we intend to summarize related NDN application research from different disciplines that may be of interest to the MILCOM community. Second, we introduce several application design patterns that are well-supported by NDN even in challenging communication conditions, and describe the architectural primitives discussed in companion papers [2], [5]–[7].

II. Related Work

Early research by the NDN team on the architecture's use in military applications includes an assessment of how NDN could provide an alternative to IP-based communication in the Army's Warfighter Information Network-Tactical (WIN-T) and the Navy's Automated Digital Networking System (ADNS) [8] and a related emulation analysis [9]. Two later papers on NDN for tactical communication environments [10], [11] evaluated its properties in environments with frequent disruptions, intermittent connectivity, and low bandwidth, and

suggested that NDN could provide significant gains in network performance over IP under such conditions.

More recently, researchers from DARPA and Tactical Blue Laboratories indicated that NDN has "demonstrable utility" to meet the security and performance requirements of tactical command and control. They conclude:

> While many of the advantages NDN exploits for its performance gains are available in some form to IP networks, such as multicast and content delivery networks, the configuration, maintenance, and long term administration overhead of those capabilities in IP networks are a significant burden for dynamic remotely deployed networks in hostile situations. NDN provides the capability of taking advantage of those resources out of the box, significantly reducing the administrative burden. [12]

The prior work, especially this baseline assessment of value for tactical networks, provides a motivation for pursuing the use of NDN in battlefield applications but does not provide much information on how NDN may impact the application design and deployment process itself. To do so, we consider salient research on civilian applications of NDN as well. The NDN project team, along with other researchers in the NDN community, have designed, simulated, and evaluated a variety of relevant applications:

- Real-time communications of audio, video, and other signals have been a regular area of study, from early work in secure voice communication [13]–[15] and instant messaging [16] to more comprehensive frameworks [17] applied to video and audio conferencing [18]. Many of these applications' designs employ host-independent communication patterns (Section III-A), and rely on NDN's inherent multicast behavior (Section III-B). They also use synchronization of shared namespaces for user/channel rendezvous and multiparty communications, avoiding the abstraction of connections between participants (see Section III-E).
- Mobile health was explored as an application domain in [19]. This work motivated research on name-based access control, and provides a complete prototype implementation of mobile data collection, secure storage, composable processing, and visualization. The application serves as a worked example of data-centric security, discussed in Section III-F. Further examples of secure communications over NDN include industrial control and building management systems [20]–[22].
- NDN's benefits for the Internet of Things (IoT) are introduced in [23]; in [24], the authors discuss in more detail how NDN can support local rendezvous and trust without reliance on cloud connectivity, which is important to battlefield scenarios. A recent vision paper by authors from Intel [25] addresses similar topics. These examples rely on directly secured data and opportunistic connectivity (Section III-D). More rapid mobility and larger scale mobile compute capability are considered in NDN vehicular networking research such as [26], [27], with results including approaches for interest forwarding via geolocation [28] and data naming strategies [29]. Vehicular applications provide relevant examples of non-unicast behavior, as discussed in Section III-B.
- Efficient content distribution was one of NDN's initial driver applications. File sharing has been an area of continuous exploration, from Dropbox-style secure sharing [30] to BitTorrent-style communication [31]. These applications illustrate end-user interfaces to pervasive storage provided through basic network mechanisms, as discussed in Section III-C.
- As an integrative, user-facing application, augmented reality (AR) can significantly benefit from NDN's capabilities. It provides examples of how NDN supports information fusion that combines real-time media sources with data from IoT devices. In [32], we propose a conceptualization of an AR ecosystem of secure, granular content and services dynamically retrieved and rendered based on user context; it illustrates how many of the design patterns described in this application can be applied holistically to offer a fundamentally new approach.

As the AR application described above is quite recent work, and has a number of facets in common with the tactical application scenario presented below—including intermittent connectivity, the use of location-specific context, processing at the edge, real-time and historical data, and granular security requirements—we will use it as our primary civilian application example in the subsequent sections. Readers are invited to review [32] or visit our related project website[2] for more details.

III. Data-centric Design Patterns

Figure 1 shows an example battlefield scenario. The distributed entities need to communicate with each other through intermittent connectivity. The nodes in the network generate chat messages in a distributed manner, and the nodes need to synchronize with each other regarding the chat messages generated.

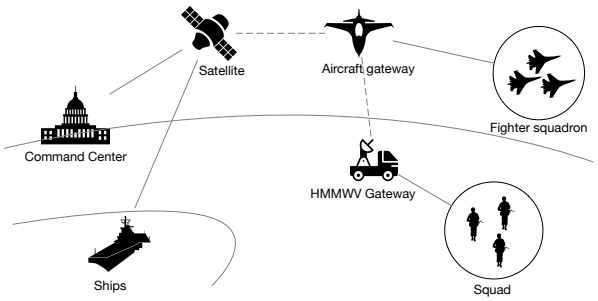

Fig. 1. Example Scenario

What is implicit in Figure 1 is actually critical. Each line of communication between entities is implemented as

[2]http://ice-ar.named.data.net

communication between computing devices (very often more than one) on each entity. Moreover, each device may have more than one radio. In the IP architecture, each interface on each node on each entity has an address, and these addresses must be managed. Thus, there is a fundamental disconnect between the notion of node interface addresses and upper layer semantics: What is actually important is the *data* produced by, observed by, or carried by the entities, which often has little to do with the compute nodes themselves.

Without NDN, relating relevant data to the addresses of individual interfaces on individual compute nodes requires one or more mapping systems. These mapping systems must translate application data semantics to numerical addresses. They must also take into account the network connectivity—distance, available resources, quality of connection, and, in a battlefield, traffic priorities and risk factors. In a battlefield, both the semantic mapping and network connectivity can change rapidly. Mapping services increase not just network complexity but the attack surface of the system, as these services become a target of focused attacks given their critical importance. NDN's name-based forwarding removes the indirection of resolving application names to network addresses, and enables network routing to work directly with application namespaces to forward requests to best available resources. Thus, any communication link available to the nodes can be used transparently to the application.

The shift in paradigm for networking from a node-centric to a data-centric approach has significant implications for application design and development. The affordances of the network and their implications—for example, the required effort in developing, provisioning, securing and maintaining the mapping systems described above—shape application designs, often preventing them from successful application of design patterns that work well in, for example, civilian cloud environments. In the following section, we suggest several design patterns common at the application layer that are applicable in tactical scenarios, difficult to achieve over IP networks, and inherently enabled by the NDN architecture.

A. Create Host-Independent Behavior

The success of the cloud paradigm for many applications demonstrates the value of host-independent design. In today's Internet applications, successfully scalable and reliable cloud applications typically abstract *capabilities* away from the individual *nodes* which provide those capabilities. This is achieved through infrastructure that provides data-to-host mappings at various layers. From DNS-based solutions for load-balancing to virtualization techniques that deploy containers as needed to any provisioned node, application designers and deployers draw from a variety of techniques to create host-independent applications that embody service-centric and/or data-centric architectures. Thus, an increasing number of design patterns are available that avoid reliance on any single host.

Unfortunately, cloud-based approaches are either unavailable or work poorly in tactical networks using the TCP/IP protocol stack. They are typically built above layer three, relying on an underlying ability to assign and manage IP addresses easily, provide name-to-address resolution, offer application-level capability-to-node mappings, and communicate within secure perimeters or channels, often established with TLS or IPSec, which in turn rely on certificate authorities living in the tactical cloud. Challenges to applying these approaches in battlefield scenarios are already stated in the related work.

Nonetheless, device-independent designs are critical, as battlefield applications can involve very large numbers of devices (sensors, etc.), which can be mobile and individually fragile. Through the mechanisms outlined in NDN literature, NDN enables nodes with limited capabilities and intermittent connectivity to forward named and secured data to each other successfully, without host-specific addressing (unless built into the data by the application).

NDN enables host-independent application designs in a wider variety of network contexts than TCP/IP, including environments with resource constrained devices, limited bandwidth, and contested channels. For our example scenario, host-independent designs enable more than one aircraft, vehicle, and warfighter to provide the same data or the same capabilities transparently to a network-wide application for redundancy or just-in-time tasking. At another level, it allows multiple nodes available within a given entity (e.g., more than one mobile device carried at the squad level) to do the same.

The impact on tactical applications is significant. Host-independence enables redundancy, disruption tolerance, and rapid tasking. With NDN, to enable commands of a certain type to reach one or multiple nodes, or for certain types of data to be retrieved from them, the team deploying or provisioning each node needs to configure it with the namespaces to listen to and provide it with the necessary cryptographic keys to sign/verify and encrypt/decrypt relevant data. It can then inform networks it is able or allowed to associate with (at lower layers) of what data it is interested in and what namespaces it would like to receive interests for. When more than one node can provide the same data, e.g., for redundant sensors, they can all inform the network, and no conflict resolution is necessary. The node's capability is inherently independent of any host address, and does not require access to any global name resolution.

The AR browser application introduced in Section II is designed for host independence with respect to both user data (context) and content. The context streams available for a user's browsing session (e.g., their location in physical space, past navigation choices) are associated with one or more user identities and published independently of the device used for browsing. For example, a tourist walking through a city square might have an Instagram-based identity and separate identity for the tourism app that they have received from the city; both identities might be associated with both their phone and smart watch. All four identity-device pairs might provide location context, while only the Instagram-phone pair might provide camera context. Each application would receive keys to decrypt the available context.

NDN names for context can be device-independent, so

authorized and interested services can retrieve it over more than one network (e.g., WiFi *and* LTE) transparently, and use standard NDN network services to handle publisher mobility, rather than requiring application-specific mappings between data and host addresses.

Location-specific AR *content* can be named with a prefix based on its geography (e.g., a USNG grid location) and fetched directly over the network layer, using NDN forwarding to direct interests to the nodes at the specified location.

Host-independent application design, where capabilities or data are provided and consumed by application processes without knowledge of endpoint addresses (e.g., reading steps from a smart watch and odometry from a phone in a device-independent fashion) is a boon to application developers, allowing them to reduce middleware reliance and operate using potentially simpler protocols.

NDN is also beneficial for those who must *deploy and support* applications, who must manage not only network address allocation and other network configuration to provide applications running on a node with the IP connectivity expected by that node. When nodes have more than one way to communicate, e.g., through multiple radios, this management becomes even more complicated because IP addresses are interface-specific, affecting both the application developer, who must manage multiple addresses and endpoints, and deployer, who must determine how to indicate to the stack which radio to use when, and for what kind of traffic.[3]

B. Embrace Multicast

Host independence is a natural application design strategy for tactical applications, as it enables resilience to failure and efficient resource tasking. With it, redundant capabilities are straightforward to implement. It incorporates the idea of multihoming—that an application instance should reach the network transparently, through whatever means their hosting nodes have available, whether over one interface or several. Having multiple data sources and multiple consumers reveals a crucial aspect of many modern applications that runs counter to a key affordance of IP as typically deployed, especially on the global Internet—the norm of unicast behavior.

Though it is the easiest approach over IP, unicast behavior is not the norm in many human and electronic communication scenarios. Humans rely on receiving a variety of environmental cues that are multicast on an ongoing basis—i.e., available to any observer in range—in addition to directly sent messages. Barnlund's *transactional model of communication* [33] shares the Shannon-Weaver concept of communication reducing uncertainty, but notes that "[i]t requires that the organism be open to all available cues and that it be willing to alter meanings until a coherent and adequate picture emerges." This observation, made about interpersonal interaction, is relevant to tactical environments as well, in which decisions should involve information gathered on an ongoing basis from many sources, behavior that is straightforward to imagine at the application level (e.g., gathering data from a variety of sensors, each may be available intermittently, transmitted over any available means using secured data) but poorly matched to unicast communication behavior on IP networks.

IP multicast is sometimes realized in tactical networks as a last resort data dissemination mechanism when unicast routing is not converging, typically due to rapid network dynamics or link instability. It is also realized in some applications that use broadcast and multicast behavior e.g., publish/subscribe message busses. These patterns are usually implemented above the network layer, either because matching features are not widely available (as in the case of multicast), network segments may differ (in the case of broadcast), or connection-oriented security is incompatible (see Section III-F). Instead, the patterns are implemented at the application layer as middleware-supported overlays, for example. A significant role of middleware and application-specific libraries is to support this behavior over underlying unicast transport, which is often connection-based and reliant on knowledge of gateway, broker, or other key node addresses. Unfortunately, common uses of TCP/IP often lead application designers to start from assumptions of unicast or connection-oriented behavior at the network layers.

NDN's intrinsic multicast behavior[4] enables application developers to be more confident that application-level strategies can be implemented at the network layer without brittleness, complexity, or inefficiencies. Further, NDN suggests an alternative design principle is possible: that unicast is not the norm, and that data should be retrievable and usable by any application authorized to use it, from anywhere it is available. NDN offers the opportunity to implement other, more application-appropriate behavior at the network layer. On wireless media, the inherent broadcast nature of the media, when exposed by underlying layers, can provide this behavior very efficiently over local networks. The impact on application design is significant. Not only can host-independence be implemented with relative ease, but an application's communication strategy can start from the concept of efficient, application named, receiver-driven multicast rather than worrying about how to support many unicast streams.

In the AR browser application introduced above, as well as the tactical network scenario described in [11], this has important practical implications. For example, the power-constrained mobile device used to browse AR content only needs to supply real-time context packets (e.g., location or even its camera feed) once to serve many simultaneous[5] requests, due to NDN's multicast behavior. This greatly reduces the power and computational burden on the mobile device. In the AR browser application, we take advantage of this to publish the

[3] By eliminating endpoint addressing and making forwarding strategy explicitly configurable based on application-defined names, NDN addresses this challenge by architecture design, though exactly how to present forwarding strategy configurations to users remains an open area of research.

[4] As described in the companion papers, a feature of the architecture is that one Data packet can satisfy many outstanding Interests. Additionally, NDN can take advantage of the multicast nature of wireless media directly: Any data "heard" over a wireless channel can be used by a node because each packet can be verified independently of how it is obtained. See Section III-F.

[5] The more cache storage available on the network, the more the constraint of simultaneity can be relaxed.

real-time camera feed from a phone only once, using data-centric security, described in Section III-F, and consume it at multiple edge services with no additional burden on the phone. For example, in the example of AR tourism from Section III-A, if the user authorizes an additional application to use the camera, in addition to the future Instagram app, it receives a decryption key for the camera context stream. The second application issues NDN Interests for the camera data and decrypts with that key. The device's camera Data is only published once, and the NDN architecture ensures an Interest from both applications is fulfilled by that one packet. So there is no power consumption or bandwidth impact caused by adding the second consumer of the camera data.

C. Enable Storage Everywhere

As described in the related work and companion papers, by naming data instead of their containers, NDN enables applications to secure *data* directly by letting them cryptographically sign each data packet to bind its name with the content; they can also encrypt the data whenever data confidentiality is needed [6], [34]. Each data object is also uniquely named and immutable. Named, secured, and immutable data can be stored, and fetched, from anywhere.

Applications designed to run over NDN can thus embrace the use of network storage in many scenarios. For example, in real-time media applications we have developed over NDN, including the AR browser, applications write new content to be published (e.g., video and audio from a videoconference or the camera stream from a mobile client) to local in-memory (or on-disk) storage and then "forget" about it, allowing that storage to handle incoming Interests at any time. With well-named data, this transparently allows the application to provide both real-time and historical data with no additional design effort, subject to the scaling limits of the storage mechanism.

By promoting the combination of 1) application-level framing [35], in which applications, rather than the network, decide how to packetize their data and 2) application-defined, network-forwarded naming of those data frames, NDN enable storage to be implemented consistently on any network-attached node by simply storing secured, application-named and -framed packets and responding to Interests for that data when appropriate. Network operators may tune the retention policies of individual caches or persistent storage for performance or security reasons. The result are applications that are inherently scalable to large numbers of content consumers, are delay tolerant, and support historical access to data generated by low-capability devices–all vital for tactical scenarios.

D. Communicate Aggressively and Opportunistically

The notion of *best-effort delivery* is an important part of the Internet's success, as it requires upper layers to determine when and how to retransmit, reconnect, etc. We propose that a broader notion of best-effort communication is possible to achieve using NDN: the idea of moving bits from one node to another by *any available means* without requiring applications to manage detailed decisions about those various means.

NDN enables tactical applications to implement this broader notion with relative ease. Though a variety of communication patterns can be implemented, an aggressive, opportunistic approach is one that we believe is promising for tactical applications in disrupted environments: Data publishers create named, signed, and encrypted data packets when new information is available or upon request. These packets are stored in their own combination of in-memory and persistent storage with an application-specific retention policy. Whenever they have connectivity, this storage subsystem responds to requests for interests received over any media. Requests not answered by storage are demultiplexed to client applications. Consumers of data issue Interests whenever they have connectivity over any media. Many, if not all, devices act as caches and forwarders of secured Data objects, holding data they receive, whether explicitly requested or overhead in a relevant namespace, and passing on Interests if they are connected on multiple interfaces. Though the best forwarding and caching strategies to employ in various scenarios is an active area of research, the general approach is promising for tactical applications.

This approach is based also on our experience with using applications over existing NDN research platforms. Early NDN forwarder prototypes provided many options for explicit configuration of forwarding between nodes over a variety of communication, from ethernet and bluetooth to TCP and UDP tunnels. But they all shared the requirement of being manually configured, though an autoconfiguration / hub discovery toolset was later prototyped. While useful for exploring different network topologies and constructing fairly stable testbeds, for application developers, we have learned a valuable lesson in trying to use similar approaches under more dynamic network conditions. Forwarders to be used in dynamic (or hostile) environments should instead be configured to create NDN routes aggressively and opportunistically to any nearby node that speaks NDN, over any media. When applications, including those for discovery/rendezvous, are issuing interests that go unanswered, these forwarding approaches can try other paths. Such self-learning techniques can be used to discover what data is available from what path, and forwarding strategies can tune how interests are forwarded based on interface cost and performance for getting answers from certain namespaces. Over this connectivity, nodes can also inform the network of namespaces in which they are interested in publishing. Forwarding using self-learning is discussed in [36]; we plan to apply it in our AR browser application, to find paths to local content and services.

A network layer that prioritizes finding and maintaining reachability to desired data provides applications with inherent disruption tolerance and data muling. Applications can potentially request from the underlying stack resilient behavior, allowing implementation consistency across applications.

E. Share Namespaces, Not Connections

The notion of a point-to-point *connection* (and to a lesser extent, a *session*) between two parties, is a fundamental abstraction in many applications that is encouraged by the

prevalence of unicast behavior in practical deployments of the IP architecture along with the success of TCP-based protocols for reliable communications. However, as discussed above, these underlying affordances are not particularly beneficial to applications in tactical scenarios–they are a subset of a much broader range of communications strategies.

Efficient set reconciliation techniques, such as those discussed in [37], enable dataset synchronization across multiple collaborating nodes that provide higher-level communication abstractions for multiparty communication. This provides a fundamental shift in how applications can conceive of, and implement, multiparty communication such as the chat messages in the example scenario we described in Section III: Distributed applications publish and consume messages in a shared dataset. A dataset synchronization protocol then takes responsibility of disseminating application data and maintaining a consistent state of the shared dataset across all participants in the system.

The data-centric nature of the NDN architecture provides a foundation to design such a data synchronization protocol (Sync) [5], [38], and build applications that leverage its connectionless, multiparty behavior. To share new data, a data producer injects the names of that data into its version of the dataset. Thanks to NDN's unique, secured binding between name and Data, Sync needs only to synchronize the revised namespace of the shared dataset among a group of distributed entities. After learning the new names, consuming applications decide whether and when to fetch the new data according to their own needs and available resources.

This use of NDN sync to provide reliable data-centric communication differs from data retrieval via a TCP connection in three fundamental ways. First, it naturally supports data retrieval among multiple parties, while TCP supports data exchange between two parties only. Second, it does not require all communicating parties to be interconnected at the same time as TCP does. Third, it does not care from where the data is returned since the security is attached to the data instead of its container or communication channel. Each of these facets addresses the requirements of tactical scenarios discussed in previous sections of this paper, as well as an increasing number of civilian applications.

Sync is used to share information on what names are available, and applications can choose how and when to fetch that data, potentially using specially named metadata objects to specify the data fetching and processing patterns to use. For example, in the AR browser application, we are exploring how to use Sync-style communication to gather information on data prefixes available for a given geographic location prefix, from many different publishers, and then use application-specific metadata, or other indexing techniques, within those prefixes, to decide what other data to fetch.

F. Secure the Data First

To implement all the patterns described above, data-centric security is required. Battlefield reality makes today's solutions of securing communication channels between a pair of communicating nodes infeasible, due to the nature of the nodes' heterogeneity and dynamics. They may move, any of them can fail at any time, and one cannot establish a secure channel between nodes of different colors. Although numerous strategies for describing and securing data in IoT systems have been proposed, almost all exist only at the application layer and run in data centers, severely limiting the ability of infrastructure to support distributed analytics in highly heterogeneous networks and provide robust enclave borders.

Applications that require group communication are common in tactical networks, but it is uncommon for these applications to generate multicast traffic natively. This is primarily due to the lack of well-defined secure transport for multicast data. As a result, it is common to observe multicast streams broken down into independent unicast streams over secure point-to-point channels like TLS, which is complex and inefficient.

The concept, design, and implementation of per-packet verification, schematized trust (leveraging name relationships to manage trust), and name-based access control are discussed in [39], [40], and [6], [34] respectively. These approaches require (and promote) application developer engagement with security concerns. The requirement to secure the data (rather than only the channel) in order to achieve many NDN communication benefits promotes designs that compartmentalize access to cryptographic keys used to encrypt or generate data, and makes it a first order part of the networking layer of all applications.[6] NDN's data-centric security unifies security across protocol stack, reduces dependency on the security of intermediaries, and is thus well-suited to hostile environments. This concept works over IoT as well.[7]

IV. Conclusion

In summary, NDN provides tactical application designers with the opportunity to apply established design patterns that are useful for battlefield scenarios but challenging to implement and deploy over IP. Such patterns include: 1) **Create host-independent behavior**. Host-independence is a key aspect of providing resilience to failure and rapid taskability; NDN makes this viable with minimal infrastructure. 2) **Embrace multicast**. Point-to-point communication is neither intrinsic to many applications nor a property of many underlying media; NDN enables application designers and providers of lower-layer communications capability to address these semantic mismatches, and provide more efficient and robust behavior both above and below the network layer. 3) **Put storage everywhere**. The combination of NDN's application-level framing, data-centric security, and application-defined names enable consistent, transparent mechanisms for storage across all platforms, making delay tolerance a much simpler

[6]This is in contrast to a reliance on transport-layer security that provides no protection outside of the communication channel, as well as to data-centric security strategies that have no network-layer support and may differ significantly in their implementation for different device types and enclaves.

[7]Low-end and handheld devices may not have sufficient computing power for even AES encryption, let alone running RSA/ECDSA for signatures. However, these devices can use weak authentication mechanisms (e.g., MACs) to interact with nearby powerful nodes to offload these computations.

concern for applications and broadly supporting distributed memory in tactical networks. 4) **Communicate aggressively and opportunistically**. NDN networking stack can be configured to communicate aggressively and opportunistically over a variety of media, reducing the coupling of mechanisms and configuration between network deployment and application behavior that is currently necessary to achieve this behavior. This enables developers to consider a more robust meaning for "best effort delivery." 5) **Share namespaces, not connections**. Efficient set reconciliation techniques enable data namespace synchronization to be a viable transport mechanism for multiparty communication, enabling a shift away from connection-oriented thinking that offers increased robustness for application design. Each of the above patterns is enabled by data-centric security, leading to the final pattern (or principle): 6) **Secure the data first**.

REFERENCES

[1] L. Zhang, A. Afanasyev, J. Burke, V. Jacobson, kc claffy, P. Crowley, C. Papadopoulos, L. Wang, and B. Zhang, "Named Data Networking," *ACM Computer Communication Review*, July 2014.

[2] A. Afanasyev, T. Refaei, L. Wang, and L. Zhang, "A brief introduction to Named Data Networking," in *IEEE MILCOM 2018*.

[3] C. Yi, A. Afanasyev, I. Moiseenko, L. Wang, B. Zhang, and L. Zhang, "A Case for Stateful Forwarding Plane," *Computer Communications: ICN Special Issue*, vol. 36, no. 7, pp. 779–791, April 2013.

[4] Z. Shelby, K. Hartke, and C. Bormann, "The constrained application protocol (CoAP)," RFC 7959, June 2014.

[5] T. Li, W. Shang, A. Afanasyev, L. Wang, and L. Zhang, "A Brief Introduction to NDN Dataset Synchronization (NDN Sync)," in *IEEE MILCOM 2018*.

[6] Z. Zhang, Y. Yu, S. K. Ramani, A. Afanasyev, and L. Zhang, "NAC: Automating access control via Named Data," in *IEEE MILCOM 2018*.

[7] T. Refaei and A. Afanasyev, "Enabling a Data-Centric Battlefield Through Information Access Gateways," in *IEEE MILCOM 2018*.

[8] B. Etefia and L. Zhang, "Named Data Networking for Military Communication Systems," in *Aerospace Conference*. IEEE, 2012.

[9] B. Etefia, M. Gerla, and L. Zhang, "Supporting Military Communications with Named Data Networking: An Emulation Analysis," in *IEEE MILCOM 2012*.

[10] M. T. Refaei, S. Ha, Z. Cavallero, and C. Hager, "Named data networking for tactical communication environments," in *2016 IEEE 15th International Symposium on Network Computing and Applications (NCA)*, 2016.

[11] C. Gibson, P. Bermell-Garcia, K. Chan, B. Ko, A. Afanasyev, and L. Zhang, "Opportunities and Challenges for Named Data Networking to Increase the Agility of Military Coalitions," in *Proceedings of Workshop on Distributed Analytics InfraStructure and Algorithms for Multi-Organization Federations (DAIS)*, 2017.

[12] J. B. Evans, S. G. Pennington, and B. J. Ewy, "Named data networking protocols for tactical command and control," in *Open Architecture/Open Business Model Net-Centric Systems and Defense Transformation 2018*, vol. 10651, 2018.

[13] V. Jacobson, D. K. Smetters, N. H. Briggs, M. F. Plass, P. Stewart, J. D. Thornton, and R. L. Braynard, "Voccn: voice-over content-centric networks," in *Proceedings of the 2009 workshop on Re-architecting the internet*. ACM, 2009, pp. 1–6.

[14] Z. Zhu, S. Wang, X. Yang, V. Jacobson, and L. Zhang, "ACT: audio conference tool over named data networking," in *Proceedings of the ACM SIGCOMM workshop on Information-centric networking*, 2011.

[15] Z. Zhu, J. Burke, L. Zhang, P. Gasti, Y. Lu, and V. Jacobson, "A new approach to securing audio conference tools," in *Proceedings of the 7th Asian Internet Engineering Conference*, 2011.

[16] Z. Zhu, A. Afanasyev, Y. Yu, and L. Zhang, "ChronoChat: a Server-less Multi-User Instant Message Application Over NDN (poster)," in *NDN Community Meeting*, Los Angeles, CA, September 2014.

[17] P. Gusev, Z. Wang, J. Burke, L. Zhang, E. Muramoto, R. Ohnishi, and T. Yoneda, "Real-time streaming data delivery over Named Data Networking," *IEICE Transactions*, May 2016.

[18] P. Gusev and J. Burke, "Ndn-rtc: Real-time videoconferencing over named data networking," in *Proceedings of the 2Nd International Conference on Information-Centric Networking*, ser. ICN '15. New York, NY, USA: ACM, 2015.

[19] H. Zhang, Z. Wang, C. Scherb, C. Marxer, J. Burke, L. Zhang, and C. Tschudin, "Sharing mhealth data via named data networking," in *Proc. of ACM ICN*, 2016.

[20] W. Shang, Q. Ding, A. Marianantoni, J. Burke, and L. Zhang, "Securing building management systems using named data networking," *IEEE Network*, vol. 28, no. 3, pp. 50–56, 2014.

[21] J. Burke, P. Gasti, N. Nathan, and G. Tsudik, "Securing instrumented environments over Content-Centric Networking: the case of lighting control," in *Proc. of IEEE INFOCOM 2013 NOMEN Workshop*, 2013.

[22] V. Perez, M. T. Garip, S. Lam, and L. Zhang, "Security evaluation of a control system using named data networking," in *Proceedings of Eighth Workshop on Secure Network Protocols (NPSec)*, October 2013.

[23] W. Shang, A. Bannis, T. Liang, Z. Wang, Y. Yu, A. Afanasyev, J. Thompson, J. Burke, B. Zhang, and L. Zhang, "Named data networking of things," in *Internet-of-Things Design and Implementation (IoTDI), 2016 IEEE First International Conference on*. IEEE, 2016, pp. 117–128.

[24] W. Shang, Z. Wang, A. Afanasyev, J. Burke, and L. Zhang, "Breaking out of the cloud: Local trust management and rendezvous in Named Data Networking of Things," in *Proc. of ACM/IEEE IoTDI*, 2017.

[25] E. M. Schooler, D. Zage, J. Sedayao, H. Moustafa, A. Brown, and M. Ambrosin, "An architectural vision for a data-centric iot: Rethinking things, trust and clouds," in *Distributed Computing Systems (ICDCS), 2017 IEEE 37th International Conference on*. IEEE, 2017.

[26] G. Grassi, D. Pesavento, L. Wang, G. Pau, R. Vuyyuru, R. Wakikawa, and L. Zhang, "Vehicular inter-networking via named data," in *ACM HotMobile 2013 (Poster)*, 2013.

[27] G. Grassi, D. Pesavento, G. Pau, R. Vuyyuru, R. Wakikawa, and L. Zhang, "Vanet via named data networking," in *Computer Communications Workshops (INFOCOM WKSHPS), 2014 IEEE Conference on*. IEEE, 2014, pp. 410–415.

[28] G. Grassi, D. Pesavento, G. Pau, L. Zhang, and S. Fdida, "Navigo: Interest forwarding by geolocations in vehicular named data networking," in *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2015 IEEE 16th International Symposium on a*. IEEE, 2015, pp. 1–10.

[29] J. Wang, R. Wakikawa, R. Kuntz, R. Vuyyuru, and L. Zhang, "Data naming in vehicle-to-vehicle communications," in *Proceedings of INFOCOM 2012 NOM Workshop*, 2012.

[30] A. Afanasyev, Z. Zhu, Y. Yu, L. Wang, and L. Zhang, "The Story of ChronoShare, or How NDN Brought Distributed Secure File Sharing Back," in *Proceedings of IEEE MASS 2015 Workshop on Content-Centric Networks*, 2015.

[31] S. Mastorakis, A. Afanasyev, Y. Yu, M. Sweatt, and L. Zhang, "nTorrent: BitTorrent in Named Data Networking," in *26th International Conference on Computer Communication and Networks (ICCCN)*, July 2017.

[32] J. Burke, "Browsing an Augmented Reality with Named Data Networking," in *26th International Conference on Computer Communication and Networks (ICCCN)*, July 2017.

[33] D. C. Barnlund, "A Transactional Model of Communication," in *Language Behavior: A Book of Readings in Communication.*, J. Akin, A. Goldberg, G. Myers, and J. Steward, Eds. Moulton, 1970.

[34] Y. Yu, A. Afanasyev, and L. Zhang, "Name-Based Access Control," NDN, Technical Report NDN-0034, Jan. 2016.

[35] D. Clark and D. Tennenhouse, "Architectural Considerations for a New Generation of Protocols," in *Proc. of SIGCOMM*, 1990.

[36] J. Shi, E. Newberry, and B. Zhang, "On Broadcast-based Self-Learning in Named Data Networking," in *Proc. of IFIP Networking*, 2017.

[37] D. Eppstein, M. T. Goodrich, F. Uyeda, and G. Varghese, "What's the Difference? Efficient Set Reconciliation without Prior Context," in *ACM SIGCOMM Computer Communication Review*, 2011.

[38] W. Shang, Y. Yu, L. Wang, A. Afanasyev, and L. Zhang, "A survey of distributed dataset synchronization in Named Data Networking," NDN, Technical Report NDN-0053, May 2017.

[39] V. J. D. Smetters, "Securing Network Content," PARC, Tech Report, October 2009.

[40] Y. Yu, A. Afanasyev, D. Clark, kc claffy, V. Jacobson, and L. Zhang, "Schematizing Trust in Named Data Networking," in *Proc. of ACM ICN*, 2015.